Cognitive Self-Organizing Network Management for Automated Configuration of Self-Optimization SON Functions

Dissertation

for the degree of

Doctor of Natural Sciences (Dr. rer. nat.)

Simon Lohmüller

University of Augsburg

Department of Computer Science

Software Methodologies for Distributed Systems

February 2019

Cognitive Self-Organizing Network Management for Automated Configuration of Self-Optimization SON Functions

Supervisor:	**Prof. Dr. Bernhard Bauer**, Department of Computer Science University of Augsburg, Germany
Advisors:	**Prof. Dr. Jörg Hähner**, Department of Computer Science, University of Augsburg, Germany
	Prof. Dr. Jörg Müller, Department of Informatics, Clausthal University of Technology, Germany
Thesis Defense:	July 16th, 2019

Abstract

As a reply to the increasing demand for fast mobile network connections, the concept of Self-Organizing Networks (SONs) has been developed, reducing the need for humans to execute Operation, Administration and Maintenance (OAM) tasks for mobile networks. A SON consists of several autonomously operating closed control loops, so-called SON functions, which can influence the behavior of the mobile network by adapting their input parameters. With SON management, a multitude of simultaneously operating SON functions can be configured according to network context-specific and weighted targets for Key Performance Indicators (KPIs), denominated as technical objectives, by using different input models. These are the operator-defined context and objective model, and the SON function manufacturer-provided effect model. SON management facilitates the SON-enabled system to work optimally regarding the achievement of defined KPI targets. Since Mobile Network Operators (MNOs) have to fulfill rising mobile network performance demands while reducing costs at the same time, it is crucial for SON management to gain an understanding of the network behavior to allow a cost-neutral performance improvement while simultaneously reducing the risk of network misconfiguration and service disturbance.

This thesis introduces four different SON management approaches, all using the same type of input models and allowing to automatically configure SON functions according to these input models. These approaches, namely Policy-based SON Management (PBSM), Objective-driven SON Management (ODSM), Adaptive SON Management (ASM) and Cognitive SON Management (CSM), thereby represents different stages of development, i.e., they build on one another and each of them overcomes the disadvantages of its predecessor.

The PBSM approach is presented which first enables the management of a system at a high level of abstraction and, at the same time, reduces manual effort. Operator and SON function manufacturer knowledge is represented in a structured and automatically processable form for the first time and an objective manager is presented that performs a reasoning process to map this knowledge to an optimal parameter configuration for each individual SON function.

However, the simplicity of the models in PBSM and the computational complexity may limit the applicability of the concept, the reason why PBSM is further developed to ODSM. In this approach, more expressive input models are used and SON functions which usually influence each other, are considered as a union, allowing for a trade-off between them.

In both approaches, the manufacturer-provided effect models are static and do not adapt to the actual network environment. This may lead to non-optimal operation of the SON system and hence to non-optimal network performance. In ASM, the actual influence of the current combined SON function configuration

on the network performance is determined by analyzing KPI measurements from the network, and the effect models are enhanced in such way that the contribution of the corresponding SON functions towards achieving the technical objectives is improved.

In the most sophisticated approach, the CSM, ASM is equipped with machine learning capabilities. The behavior of SON functions in the network is analyzed using four different algorithms in the field of supervised learning in order to predict their effects under untested parameter configurations. Also, performance data of network cells are analyzed for similarities using techniques in the field of unsupervised learning. That is, machine learning is applied to complement the sketchy effect models, giving the CSM system a wider range of possible configurations.

The four different stages are evaluated in a realistic mobile network simulator to show the value of SON management in general and the performance improvement to previous stages of development. While these approaches provide an increasing level of maturity from PBSM to CSM, they all are designed in a way that the MNO has full control over the mobile network at any time and that he or she can interrupt automated actions at any time.

Acknowledgments

First of all, a big thank you goes to my supervisor, Prof. Dr. Bernhard Bauer, for providing me the opportunity to write my Ph.D. thesis at his professorship. Thank you for all the helpful discussions, for the permanent support, even when I decided to change the topic of my thesis after more than two years. There were several moments during the past six years when I was faced with doubts regarding my thesis, but you could always eliminate these doubts and motivate me again. I am deeply grateful for the enjoyable working atmosphere within this time and for the (still ongoing) support, especially in the past year, that was probably one of the most decisive years in my life.

Special thanks go to my colleagues at Nokia Bell Labs, first and foremost to Lars Christoph Schmelz and Dr. Henning Sanneck, for the countless discussions as well as the positive and, more important, the critical feedback that was pushing me even more. Also, thanks for the opportunity to write my dissertation in collaboration with one of the big players in the field of mobile networks and for the great time we had at journeys to conferences and during the SEMAFOUR project.

I would like to thank all my colleagues and former colleagues at the Software Methodologies for Distributed Systems Lab. When colleagues become friends, working conditions can not be any better. I was really appreciating the funny conversations we had during coffee breaks, that made it much more easier to go back to work in stress situations. I want to name and thank Christoph Frenzel who was supporting me during the first years of my research, and Andrea Fendt for her support during the last years. I am also thankful that you, Sonja, were relieving me from so many organizational tasks so that I could fully set my focus on this thesis.

A hearty thank goes to my family: My parents, who were always supporting me in all my decisions, who supported me during my studies, thereby making this dissertation possible at all and who were encouraging and motivating me at any time; my brother Philipp who was advising and encouraging me during the past years and beyond, whether it be at work or at any other time; finally, my girlfriend Nathalie for her incessant encouragement when I was having doubts about my thesis and for suffering my bad mood in times of stress.

Last but not least, I want to thank all my friends who directly or indirectly supported me in writing this thesis. Especially in the last year, I was sometimes too focused and forgot to relax and rest. You always accomplished to distract me from work and hence, you also have a big stake in this thesis.

Contents

Part I.

Introduction and Basics

1 Introduction

With the rise of smart phones came a huge increase in demand for mobile internet connections. In addition to an increasing number of data intensive services such as mobile video streaming, the Internet of Things (IoT) is creating a whole new category of devices demanding reliable and capable Radio Access Technologys (RATs) [HSS11]. In fact, Hämäläinen et al. estimate "that we will see a hundredfold increase in network traffic in the near future" [HSS11]. This assumption is endorsed by Cisco who estimate the global mobile data traffic to be increased sevenfold within only six years and the number of mobile-connected devices to be increased to 11.6 billion by 2021 [CWP17] (cf. Figure 1.1).

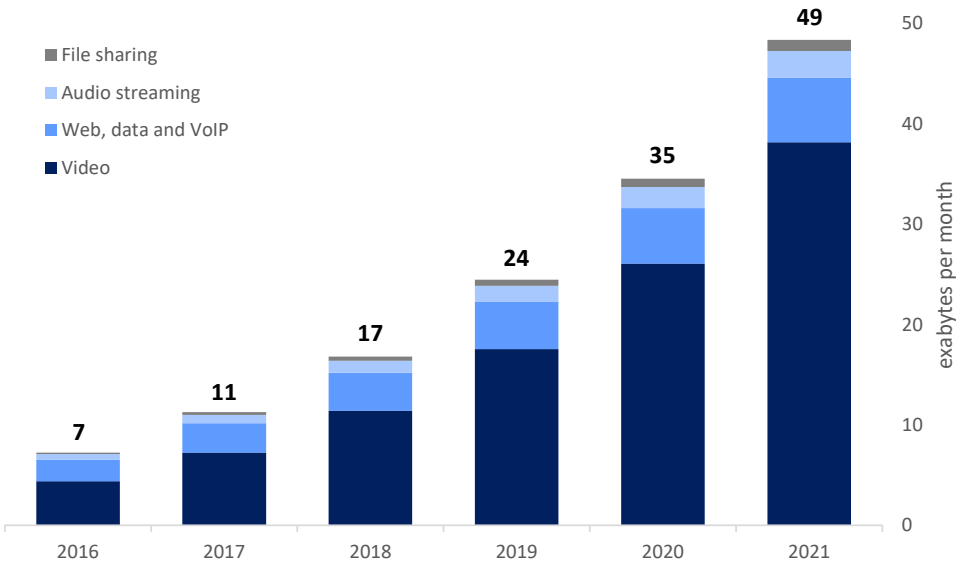


Figure 1.1.: Global mobile data traffic by type and year in exabytes(EB) per month (adapted from [CWP17])

However, prices per megabyte are actually going down due to fierce competition among Mobile Network Operators (MNOs). This puts high pressure on MNOs to reduce expenses in order to remain profitable. Meeting the expected demand, MNOs are forced to invest into their infrastructure and will inevitably be faced with high Capital Expenditures (CAPEX). Currently, the operation of a network is still largely based on a centralized MNO approach, executed by human operators. Therefore, the focus lies on limiting Operational Expenditures (OPEX) [Fre16]

which can be achieved by automating the time-consuming, expensive and error-prone tasks manually executed by human MNOs [HSS11].

Today's mobile cellular Radio Access Networks (RANs) are highly complex heterogeneous systems where different RATs such as Enhanced Data Rates for GSM Evolution (EDGE), Universal Mobile Telecommunication System (UMTS), Long Term Evolution (LTE) or WiFi are deployed together, and multiple cellular layers within each of these RATs, such as macro, micro and pico cells, are implemented. The environment of these cells differs in multiple domains such as their geographical layout or the motion profile of each user. This complexity makes the Operation, Administration and Maintenance (OAM) of these networks a demanding task for the MNO since manual configuration, optimization and failure recovery becomes increasingly difficult, not mentioning the experience an operator would need to do so. For example, there may be different target values for Key Performance Indicators (KPIs) such as the Dropped-Call Rate (DCR) or Hand-over Success Rate (HOSR) within dedicated cells or cell groups of the network. To enable a satisfactory user experience it is necessary to configure cells in accordance to requirements posed by the environment. To achieve these targets, different settings for Network Configuration Parameter (NCP) (e.g., increasing the transmission power or changing the tilt of an antenna) are required which is barely possible to achieve through a manual approach. Furthermore, certain aspects of the environment can change over time, e.g., commuters leaving the down town area of a city in the evening and thereby producing a multiple of traffic compared to the rest of the day. This fact impedes the manual adaptation of the RAN even further and requires a constant readjustment of the parameters, immensely increasing the workload for an MNO. [HSS11] Based on these needs the concept of SON was developed in 2007 by the Next Generation Mobile Networks (NGMN) alliance [NGM07] and 3rd Generation Partnership Project (3GPP) [3GP18b].

SON leverages self-management methods and techniques, with the aim to automate dedicated day-to-day OAM tasks. Thereby, SON is separated into self-configuration, self-optimization and self-healing tasks where each of these tasks is implemented through a set of autonomously operating closed control loops, called SON functions, that gather information from the network in terms of Performance Management (PM), Configuration Management (CM) as well as Failure Management (FM) data and compute new NCP settings which are then deployed to the cells. By modifying the input parameters of a SON function, the so-called SON Function Configuration Parameters (SCPs), its behavior can be influenced regarding its impact on NCPs and in turn the network KPIs [Hah+14]. For self-optimization SON functions which are in the focus of this thesis, such as Coverage and Capacity Optimization (CCO), Mobility Robustness Optimization (MRO) and Mobility Load Balancing (MLB), usually only one single or a small set of KPIs are improved by the NCP changes. For this reason, several independent SON functions need to run in parallel to improve the overall network performance and to achieve the MNO's goals in terms of handling network complexity and

OPEX reduction.

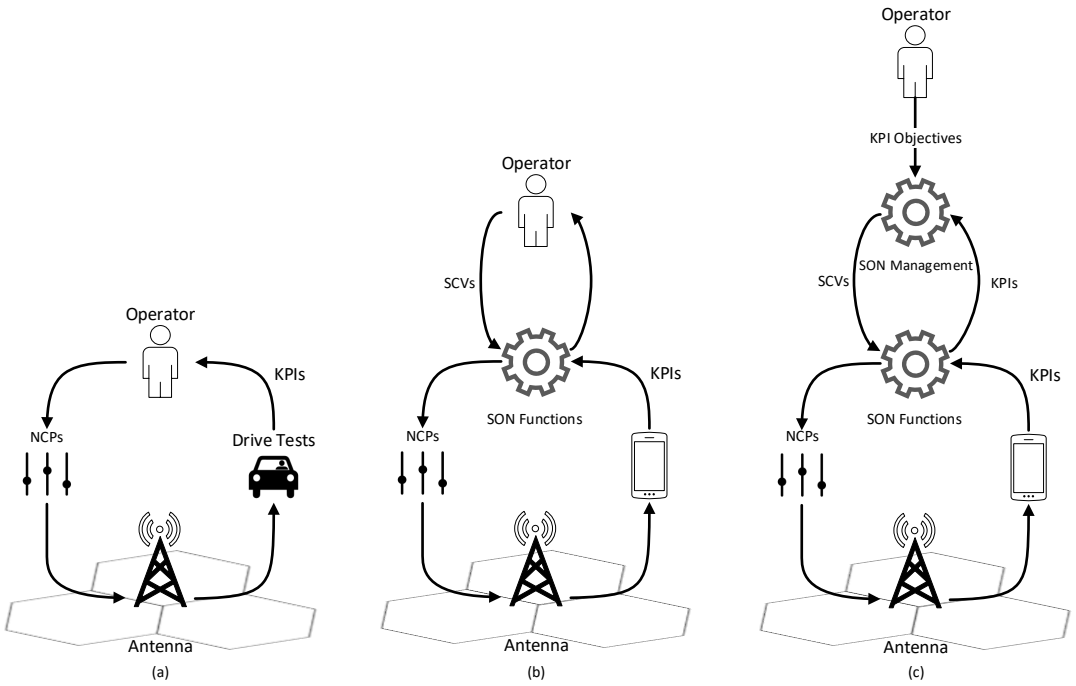


Figure 1.2.: Evolution from a network not using SON (a) to a SON-enabled network using default configurations (b) to a fully managed SON (c)

With the introduction of SON the problem of manual NCP adaptation depicted in Figure 1.2 (a) has been overcome. However, in current SON-enabled systems the input SCPs still need to be adjusted manually by human MNOs. Due to the required effort, usually a default configuration, provided by the SON function manufacturer, is applied to each SON function (cf. Figure 1.2 (b)). However, often only one default configuration for a SON function is used, and this default configuration remains unchanged during network operation, i.e., the SON function instances deployed on different cells are usually uniformly configured. This may lead to a non-optimal operation of the SON since default configurations do not adapt to changing operational context. Furthermore, as a SON function in general is delivered as a black-box, the determination of dedicated configurations for a set of SON functions is non-trivial for an MNO since it is not possible to accurately estimate the behavior for certain configurations.

The goal of the MNO is to run the network in such a way that it works optimally according to dedicated technical objectives, i.e., target values that have been defined for KPIs. A KPI target may furthermore be dependent on context information such as the time of the day or a certain cell type. Changes to a KPI target or to the context requires adjusting the configurations for the SON functions in order to adapt their behavior in such a way that they contribute to the changed KPI target. In case only default configurations are used, and no adjustment is performed, the SON-enabled network may not operate optimally. If an adjustment of SON function configurations shall be performed, considerable manual

intervention by the human operator is required in today's mobile networks. In order to relief human MNOs from these complex manual tasks, SON management (cf. Figure 1.2 (c)) performs an automated transformation of KPI targets into so-called SON Function Configuration Parameter Value (SCV) sets facilitating objective-driven control of the SON functions' behavior.

Starting with a SON management approach tackling the challenge to automate the manual process of configuring SON functions, this thesis presents in total four different SON management approaches where each of them enhances and gradually improves its predecessor. All these approaches use three types of models which the transformation process is based on: An objective model specifying KPI targets an MNO wants to achieve, a context model defining context parameters and values of the mobile networks' environment and an effect model giving an indication how certain SCV sets influence network behavior. On the way to the fourth and most sophisticated stage of development, the *Cognitive SON Management*, these input models were successively refined and brought closer to reality from manually defined models to automatically generated models using techniques in the field of machine learning.

1.1. Problems and Challenges

This section describes concrete problems in the management of SONs. Most of these problems are the result of discussions with the industrial project partners from Nokia Bell Labs and within the European Union (EU) FP7-funded project Self-Management for Unified Heterogeneous Radio Access Networks (SEMAFOUR) [SEM12]. Tackling these problems, different challenges are faced when aiming at optimizing the process of operating a mobile network.

1.1.1. Automated Management of Self-Organizing Networks

The primary aim in the management of mobile radio networks is not the optimization of dedicated single performance indicators at the cell or base station level, but the achievement of dedicated KPI targets, i.e., target values being defined for the KPIs. KPI targets may furthermore be dependent on context information which is a set of cell parameters and environment parameters where cells operate in. Whenever either objectives or the operational context changes, the SON functions' behavior needs to be adapted by adjusting their SCPs such that they account for an optimal fulfillment of KPI targets. Since only default configurations are used in today's mobile networks, the SON functions' behavior must be affected manually to enable an optimal network performance. This opens up a *manual gap* in the automated operation of a SON-enabled mobile radio network which is illustrated in Figure 1.3.

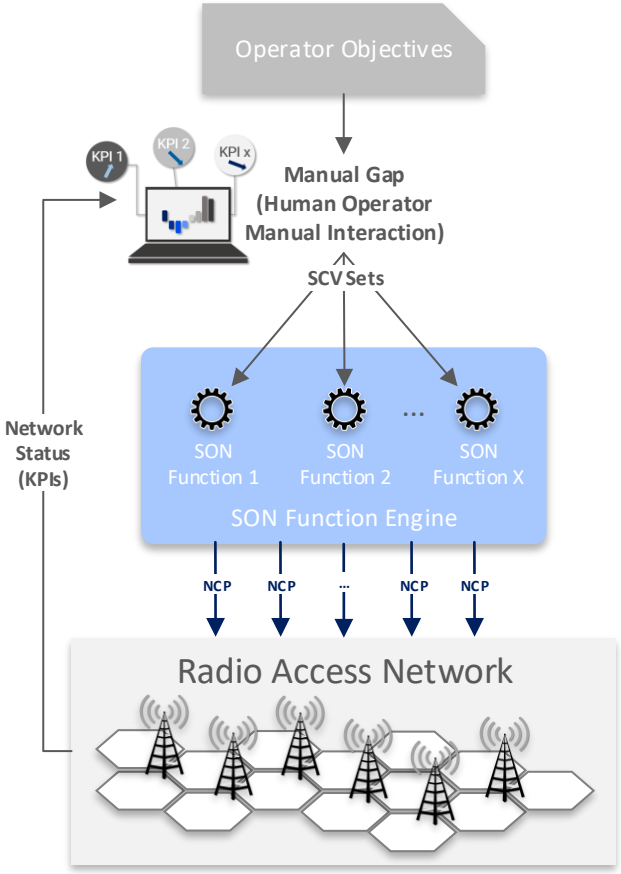


Figure 1.3.: Manual gap between operator objectives and SCV sets

Problems
The management of SONs requires a structured definition of operator objectives and knowledge about the SON-enabled network. In addition, a description of the behavior of SON functions deployed in the network in order to overcome the manual gap. The manual gap is divided into three major problems for which no solutions exist in current systems:

Automation Gap In current systems there is neither an entity available to manage the linking between operator objectives and SCV sets nor there are methods available to perform this mapping in an automated way.

Dynamics Gap The SON-enabled network and thus, the operational and network context, may be subject to frequent changes which requires a dynamic adaptation of SCV sets.

Knowledge Gap SON functions are usually delivered as black-boxes by the manufacturer, i.e., the MNO has no insight into the actual SON function algorithm which even complicates the manual adaptation of SCPs. Furthermore, operator knowledge is completely based on experience and does not exist in a way that it can be automatically processed.

Challenge 1 *Knowledge representation about operator objectives, network context and SON functions in a structured way and provisioning of an entity that is able to perform an automated transformation of KPI targets into SCV sets whenever a reconfiguration of the network is necessary.*

1.1.2. Design of Realistic Input Models

Several input models serve as a starting point to enable a mapping process between operator objectives and SCV sets. The objective model contains targets in terms of KPIs which are defined by human MNOs. A manually constructed context model provided also by MNOs describes cell parameters and the environment where cells operate in. Effect models are coming from the manufacturers of SON functions and provide a description of the respective SON function given a certain configuration. An overview of these models and their origin is depicted in Figure 1.4. The following problems need to be tackled in order to design these models in a realistic way.

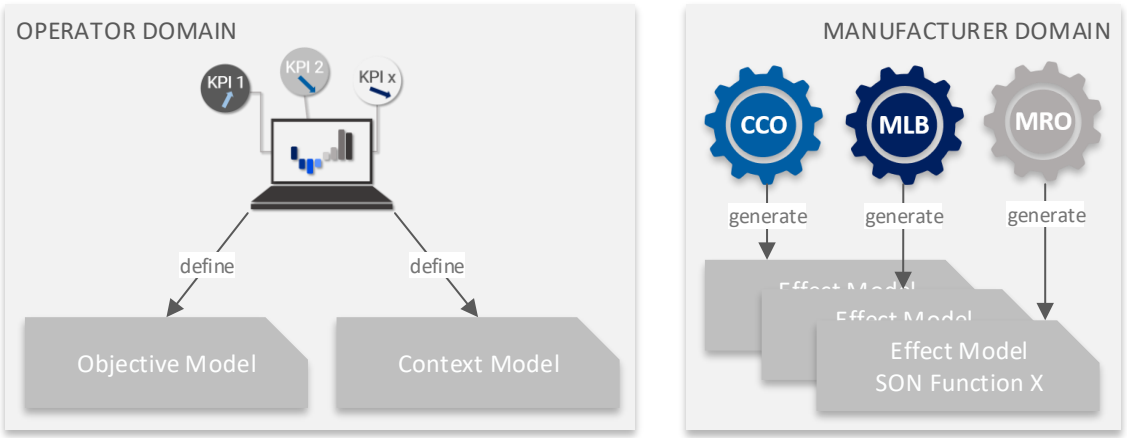


Figure 1.4.: Overview of input models defined in different domains

Problems
Manual OAM of a mobile network is a very labor intensive and error-prone task which requires a lot of experience. Initially provided input models are either manually defined or provided by third parties, it is highly probable that they do not prove true in a real network environment which may lead to an undesired behavior.

- A mobile network consists of hundreds and thousands of base stations with a multitude of different context parameters and values they can take on. Capturing all these parameters and combining them leads to an exponentially growing design space and consequently, an unacceptable runtime of the optimal SCV set calculations.

- Defining KPI targets for each state in the context space requires excessive manual effort from a human MNO.
- Manufacturer-provided effect models are generated for each single SON function individually as each of them aims at the optimization of a dedicated single network KPI. However, in order to fulfill a set of operator objectives, different SON functions which may influence each other, have to run concurrently.
- Effect models are generated by SON function manufacturers in a different environment than the network where they are deployed in the end. Hence, their validity must be questioned.

Challenge 2 *Develop methods to gradually improve the manually created, unrealistic input models provided by network operators and SON function manufacturers and continuously adapt them to a real network environment.*

1.1.3. Cognition in SON Management

Due to the dimension and complexity of modern cellular networks, even experienced operators and SON function manufacturers can not capture all network properties and estimate the performance of SCV sets in a satisfactory way. This results in sketchy input models which can not be complemented even during the operation of the network since a trial and error testing of undocumented inputs could lead to an unsatisfactory Quality of Service (QoS) in terms of, e.g., network coverage, voice quality or data rates, which each MNO wants to avoid under any circumstances. This requires for a mechanism to fill in the blanks in these models while at the same time minimizing the risk of an undesired network behavior. Machine learning uses statistical techniques to equip a system with artificial intelligence which can be used to approximate the incomplete input models to reality and to complement them.

Problems
Since each manufacturer of a SON function derives an effect model from experiments in his or her own environment, there is little guarantee that the predictions hold true in real world applications where the SON functions might face a different environment and work alongside other SON functions. A situation may occur where none of the tested SCV sets does well in achieving given operator objectives. However, there may be untested SCV sets possibly doing better in fulfilling KPI targets. These facts motivate the following problems:

- Predictions in the effect models are dependent on context which has been manually defined by MNOs and which is sketchy and faulty with the utmost probability.

- Mappings from SCV sets to network KPIs are inaccurate and sketchy with the utmost probability.
- Testing all SCV sets in the real network is simply not feasible. However, untested configurations may lead to a better objective fulfillment.

Challenge 3 *Gain a better understanding of the network behavior by introducing a method to enhance SON management models with machine learning capabilities in order to allow a cost-neutral performance improvement while simultaneously reducing the risk of network misconfiguration and service disturbance.*

1.1.4. Trust in SON Management

During the discussions with MNOs in the SEMAFOUR project [SEM12] it appeared that on the one hand a reduction of the manual effort is necessary to cope with the increasing complexity of OAM tasks in mobile networks and the increasing OPEX involved. On the other hand relieving MNOs from manual tasks does not mean to dispossess them of control over the SON-enabled network.

Problems
The thin line between increasing the degree of automation as much as possible and leaving the human MNO the last entity to make crucial decisions implicates a set of problems which are listed below:

- An MNO always wants a guarantee that the network will result in an acceptable state when using machine-made models that have been inferred automatically.
- An MNO always wants to be in a position to interrupt machine-aided and learned actions.

Challenge 4 *Even though automation is one of the key challenges in the development of a cognitive SON management system, possibilities for the exercise of influence have to be provided for MNOs in order to let them decide about the trustworthiness of machine-made models.*

1.2. Objectives, Approach and Contributions

This section identifies objectives to deal with the problems and challenges in the management of SON-enabled systems, as outlined in Section 1.1. Therefore, a high-level overview over involved domains and their models is given in Figure 1.5 where objectives are assigned to the respective domain or models. Subsequently, the main contributions of this thesis with respect to the objectives are listed.

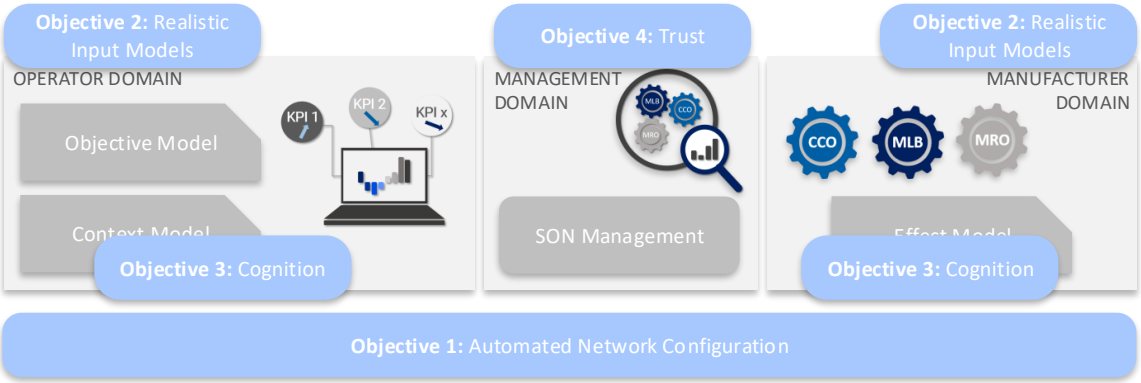


Figure 1.5.: Overview of main objectives tackled in this thesis

1.2.1. Automated Management of Self-Organizing Networks

Figure 1.5 depicts the three domains which are involved into the design of a SON management, i.e., the operator domain, manufacturer domain and the network architect domain. In order to overcome *Challenge 1* defined in Section 1.1.1, an appropriate design of models on operator and manufacturer side as well as of the SON management component itself is necessary.

Objective 1 *Automate the process of finding optimal SCV sets and close the manual gap between operator objectives and SCV sets.*

Contributions
In order to overcome the manual gap by enabling an automated network configuration, the following components are developed:

- An *objective manager* is invented as core component of the SON management performing an automated mapping process between models from the operator and manufacturer domain. The resulting SON management approach is designed according to established software engineering techniques in order to guarantee extendibility, re-usability and compatibility for further stages of development.

- A *policy system* has been integrated into the SON management approach enabling to dynamically react on changes in the network at any time. Therefore, the whole SON management system is designed in a form that it enables the usage of policy-based techniques.

- Knowledge from the operator and manufacturer side is provided in an *automatically processable form*. Objective model, context model as well as effect model are designed as a set of rules such that they can be easily used by the SON management.

1.2.2. Design of Realistic Input Models

The second objective covers the three types of input models within the operator and manufacturer domain. In order to overcome *Challenge 2* defined in Section 1.1.2, the initially provided models generated by MNOs and SON function manufacturers are gradually refined and improved to bring them closer to reality.

Objective 2 *Automatically generate realistic and complete input models which are continuously updated according to the current network state such that they optimally support the SON management system in finding the best possible network configuration.*

Contributions
To enable an optimal configuration of the mobile network, initially provided input models are enhanced or replaced by the following more realistic models:

- A *context model* is created that combines network states in which cells behave similarly such that the SON management only has to calculate SCV sets for a severely reduced set of contexts.
- An *objective model* is created that uses context model information in order to reduce the number of operator objectives and hence the manual effort for an MNO.
- A comprehensive data analysis is performed for each SON function in order to understand their behavior under certain configurations. The data analysis results are used to generate a *combined effect model* reflecting the impact of running several SON functions in parallel.
- *Network measurements* are used to generate an additional effect model over time expressing the impact of combined SCV sets in a real network environment.

1.2.3. Cognition in SON Management

The third objective covers the context model as well as the effect model. In order to overcome *Challenge 3* defined in Section 1.1.3 an extensive data analysis is performed and data mining techniques are applied to accomplish a network performance improvement while simultaneously reducing the risk of misconfiguration.

Objective 3 *Reliably estimate the performance of untested SCV sets dependent on automatically derived context information which has been reduced to a manageable level.*

Contributions
To enable a better understanding of the impact of (probably untested) SCV sets, the following models are developed:

- *Supervised learning techniques* are adopted to automatically create a context model based on an analysis of the cells' behavior in the network.
- Techniques in the field of *unsupervised learning* are adopted to improve estimations of SCV sets.
- *Different algorithms* in the field of unsupervised learning are used to estimate the performance of untested SCV sets and filling the blanks in sketchy effect models.

1.2.4. Trust in SON Management

The fourth and last objective covers the management domain. In order to overcome *Challenge 4* defined in Section 1.1.4, the SON management system is designed in a way such that the MNO can influence the usage of machine-made models at any time.

Objective 4 *Develop a fully automated SON management system which allows MNOs to comprehend, restrict and influence automated actions at any time.*

Contributions
To establish a trustworthy SON management, the whole system is developed following the subsequent principles:

- *Metrics* are defined that indicate the trustworthiness of machine-made models such that it is on the MNO's authority to decide whether calculated models are good enough to be used by the SON management or not.
- *Possibilities* are provided to the MNO to decide about which models shall be used for the calculation of optimal SCV sets.

1.2.5. Approach

The overall objective of this thesis is to develop a SON management approach that overcomes all challenges tackled on the way to fulfill *Objectives 1-4*. Therefore, a simple SON management is invented serving as a starting point which introduces the general architecture of the whole system. This thesis then aims at gradually refining the initial approach with manually created and unrealistic models into a cognitive SON management that performs a very complex analysis of the SON functions' behavior by applying machine learning techniques to reliably estimate network performance. Thereby, MNOs' requirements are always

considered as hard side constraints while evolving different stages of SON management.

1.3. Outline

An overview of all chapters in this thesis is illustrated in Figure 1.6, whereby arrows indicate dependencies between chapters. Readers which are familiar with the concepts of SON and machine learning, may skip Chapter 2 or parts of it. Chapter 3 represents the basic chapter for Chapter 4 - Chapter 7 and should be read for a full understanding of the overall approach. Chapter 4 - Chapter 7 are strongly related to each other whereby every chapter only describes the delta to previous chapters in the main section, meaning they can not be read separately. Chapter 8 shows the results of evaluating the main section. The thesis is summarized and an outlook is provided in Chapter 9.

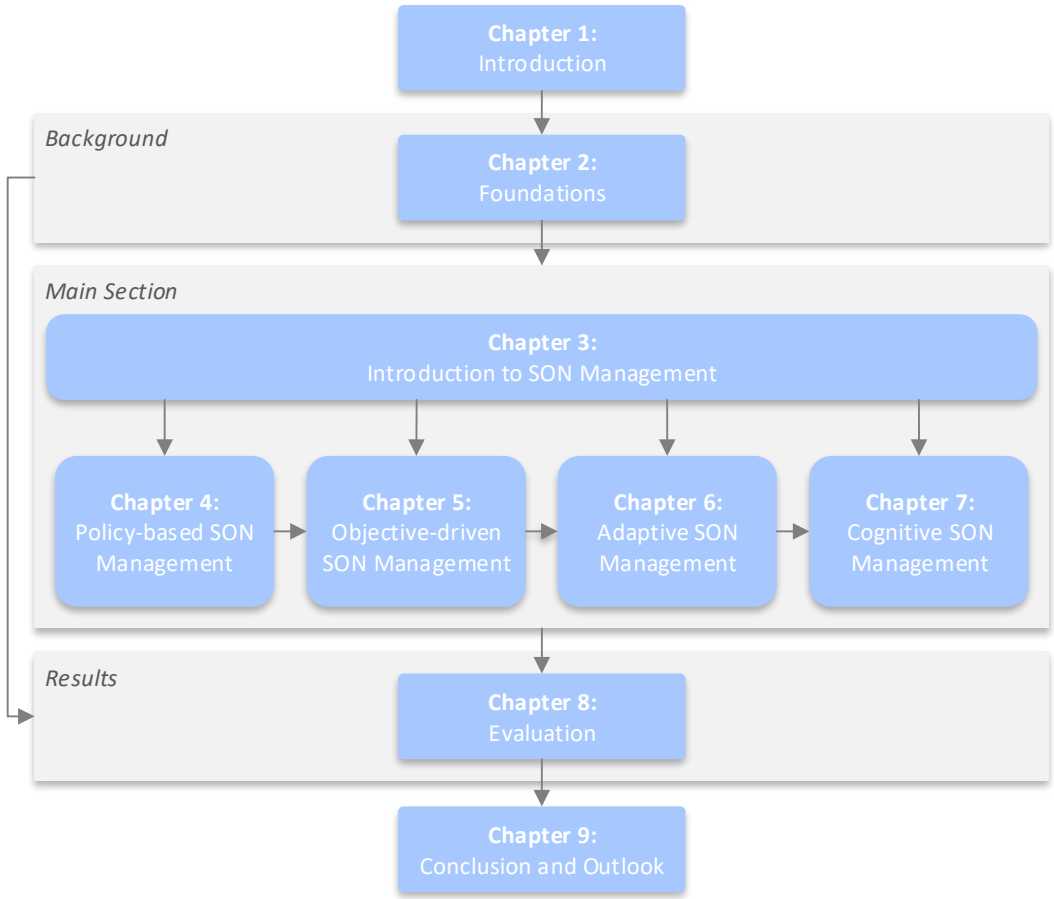


Figure 1.6.: Overview of chapters in this thesis

Chapter 1 Introduction motivates the research that has been done for this thesis. It illustrates why SON management is an inevitable part of mobile network automation. Problems and challenges are identified when developing a SON management system. Based on that, the main objectives and resulting contributions are listed. Finally, an overview of the chapter is given, followed by a list of the author's publications.

Chapter 2 Foundations describes the underlying basics that are necessary for a full understanding of this thesis. A general introduction to the concept of SON is followed by a description about policy-based network management. Subsequently, applied machine learning techniques, more precisely, supervised and unsupervised learning methods, are presented in more detail.

Chapter 3 Introduction to SON Management gives a general overview about the architecture of SON management as well as the operator and manufacturer input models. This chapter serves as a basis for Chapter 4 - Chapter 7, i.e., the presented architecture and models can generally be applied for all of these chapters.

Chapter 4 Policy-based SON Management presents the first approach on SON management. A structured description for the several input models is provided and a method is introduced to perform a mapping process between the operator and manufacturer domain.

Chapter 5 Objective-driven SON Management builds the second step in the evolution of SON management. Initially provided input models are rendered more precisely and a new methodology to transform operator objectives into combined SCV sets is presented.

Chapter 6 Adaptive SON Management introduces an approach that uses feedback from the network to enhance existent input models with a more realistic effect model. Furthermore, the calculation of optimal SCV sets is improved by a more sophisticated method.

Chapter 7 Cognitive SON Management presents the final SON management approach in which artificial intelligence is used to further enhance and improve the context model and effect model. This again requires for a more complex methodology with respect to SCV set selection.

Chapter 8 Evaluation details the realization and implementation of the proposed approaches. The general setup is presented followed by an explanation of how to derive used input models. Finally, the different SON management approaches are evaluated and compared in different ways.

Chapter 9 Conclusion and Outlook summarizes the results by referring to objectives and contributions presented in Section 1.1. An outlook on future research topics related to SON management complements the thesis.

1.4. Publications

Parts of this thesis have already been presented in other publications. In the following, a list of all of the author's publications is provided, together with a short summary and the relevance for this thesis. Besides scientific publications, patent applications and project deliverables are considered as well. Concepts and results of the author that have already been published and which are applied in this thesis are not additionally cited.

1.4.1. Scientific Publications

1. Christoph Frenzel, Simon Lohmüller and Lars Christoph Schmelz. "Dynamic, context-specific SON management driven by operator objectives". In: *2014 IEEE Network Operations and Management Symposium (NOMS)* (May 2014), pp. 1-8 [FLS14a]

   The author of this thesis, together with two fellow researchers, presents a SON management approach which transforms operator objectives into SCV sets. The ideas and concepts as well as the publication itself have been completely developed and written in cooperation with the other authors making the whole work the intellectual property of all involved authors in equal parts. This work is predominantly integrated into Chapter 3 and Chapter 4.

2. Lars Christoph Schmelz et al. "SON management demonstrator". In: *2014 IEEE Network Operations and Management Symposium (NOMS)* (May 2014), pp. 1-2 [Sch+14c]

   The author of this thesis, together with several fellow researchers, presents a demonstrator based on the concepts and ideas presented in [FLS14a]. While this work is not directly integrated into this thesis, the ideas and concepts are relevant for the development of a Policy-based SON Management (PBSM) approach as described in Chapter 4. The author of this thesis has been significantly involved in developing the ideas and concepts as well as the implementation of mentioned approach.

3. Christoph Frenzel, Simon Lohmüller and Lars Christoph Schmelz. "SON management based on weighted objectives and combined SON Function models". In: *2014 11th International Symposium on Wireless Communications Systems (ISWCS)* (Aug. 2014), pp. 149-153 [FLS14b]

   The author of this thesis, together with two fellow researchers, presents an enhancement of the SON management approach presented in [FLS14a]. The

ideas and concepts as well as the publication itself have been completely developed and written in cooperation with the other authors making the whole work the intellectual property of all involved authors in equal parts. This work is predominantly integrated into Chapter 5.

4. Lars Christoph Schmelz et al. "Demonstrator for objective driven SON operation". In: *2014 11th International Symposium on Wireless Communications Systems (ISWCS)* (Aug. 2014), pp. 506-507 [Sch+14a]

   The author of this thesis, together with several fellow researchers, presents a demonstrator based on the concepts and ideas presented in [FLS14b]. While this work is not directly integrated into this thesis, the ideas and concepts are relevant for the development of a Objective-driven SON Management (ODSM) approach as described in Chapter 5. The author of this thesis has been significantly involved in developing the ideas and concepts as well as the implementation of mentioned approach.

5. Sören Hahn et al. "Classification of Cells Based on Mobile Network Context Information for the Management of SON Systems". In: *2015 IEEE 81st Vehicular Technology Conference (VTC Spring)* (May 2015), pp. 1-5 [Hah+15a] The author of this thesis, together with several fellow researchers, describes different applications for cell classification in mobile networks. The author's contribution to this work comprises the definition SON management in section I B. and the definition of context and classes in section II A. This work is predominantly integrated into Chapter 6.

6. Simon Lohmüller et al. "Policy-Based SON Management Demonstrator". In: *2015 IEEE 81st Vehicular Technology Conference (VTC Spring)* (May 2015), pp. 1-2 [Loh+15]

   The author of this thesis, together with several fellow researchers, presents an extension of the demonstrator presented in [Sch+14a]. While this work is not directly integrated into this thesis, the ideas and concepts are relevant for the development of a ODSM approach as described in Chapter 5. The author of this thesis has been significantly involved in developing the ideas and concepts as well as the implementation of mentioned approach. Furthermore, the author of this thesis is mainly responsible for the publication itself.

7. Simon Lohmüller, Lars Christoph Schmelz and Sören Hahn. "Adaptive SON management using KPI measurements". In: *NOMS 2016 - 2016 IEEE/IFIP Network Operations and Management Symposium* (Apr. 2016), pp. 625-631 [LSH16]

   The author of this thesis, together with two fellow researchers, presents an enhancement of the SON management approach presented in [FLS14b]. All ideas and concepts have been developed together with mentioned authors. The publication itself and the evaluation have been accomplished by the author of this thesis. This work is predominantly integrated into Chapter 6.

8. Christian Mannweiler et al. "Cross-domain 5G Network Management for Seamless Industrial Communications". In: *NOMS 2016 - 2016 IEEE/IFIP Network Operations and Management Symposium* (Apr. 2016), pp. 868-872 [Man+16]

   The author of this thesis, together with several fellow researchers, presents a concept to enable a cognitive, joint management of mobile industrial and cellular networks. The author of this thesis has supported the other authors by means of discussions and by reviewing the publication. This work is only integrated into Chapter 9 of this thesis on an abstract level.

9. Lars Christoph Schmelz et al. "Demonstrator for adaptive SON management". In: *NOMS 2016 - 2016 IEEE/IFIP Network Operations and Management Symposium* (Apr. 2016), pp. 991-992 [Sch+16]

   The author of this thesis, together with several fellow researchers, presents a demonstrator based on the concepts and ideas presented in [LSH16]. While this work is not directly integrated into this thesis, the ideas and concepts are relevant for the development of a Adaptive SON Management (ASM) approach as described in Chapter 6. The author of this thesis has been significantly involved in developing the ideas and concepts as well as the implementation of mentioned approach.

10. Christoph Frenzel et al. "Demonstrator for utility-based SON management". In: *2016 IEEE 27th International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)* (Sep. 2016), pp. 1-2 [Fre+16]

    The author of this thesis, together with several fellow researchers, presents an extension of the demonstrator presented in [Sch+14a]. While this work is not directly integrated into this thesis, the ideas and concepts are relevant for the development of a ODSM approach as described in Chapter 5. The author of this thesis has been significantly involved in developing the ideas and concepts as well as the implementation of mentioned approach.

11. Simon Lohmüller et al. "SON Function Performance Prediction in a Cognitive SON Management System". In: *2018 IEEE Wireless Communications and Networking Conference Workshops (WCNCW)* (Apr. 2018), pp. 13-18 [Loh+18]

    The author of this thesis, together with several fellow researchers, presents a concept to extend SON management with learning capabilities. The concepts and ideas have been developed in cooperation with a master student within the scope of a master thesis. The master student has been supervised by the author of this thesis. The publication itself has been written by the author of this thesis. This work is predominantly integrated into Chapter 7 and Chapter 8.

12. Andrea Fendt et al. "A Network Slice Resource Allocation Process in 5G Mobile Networks". In: *2018 12th International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS)* (Jul. 2018), pp. 695-704 [Fen+18c]

The author of this thesis, together with several fellow researchers, presents a vision of an end-to-end network slice resource allocation process. The author of this thesis has supported the other authors by means of discussions and by reviewing the publication. This work is only integrated into Chapter 9 of this thesis on an abstract level.

13. Andrea Fendt et al. "A Network Slice Resource Allocation and Optimization Model for End-to-End Mobile Networks". In: *2018 IEEE 1st 5G World Forum (5GWF'18)* (Jul. 2018) [Fen+18b]

    The author of this thesis, together with several fellow researchers, presents an Integer Linear Program for off-line mobile network slice embedding. The author of this thesis has supported the other authors by means of discussions and by reviewing the publication. This work is only integrated into Chapter 9 of this thesis on an abstract level.

1.4.2. Patent Applications

1. Lars Christoph Schmelz, Christoph Frenzel and Simon Lohmüller. "Network Entity and Method for Controlling a SON-Function." WIPO Pub. No. WO 2014/191469 A1. Dec. 2014 [SFL14]

   The author of this thesis, together with two fellow researchers, presents an approach to overcome the manual gap (cf. Section 1.1.1) by introducing a component to combine operator objectives and SCV sets. The ideas and concepts as well as the publication itself have been completely developed and written in cooperation with the other authors making the whole work the intellectual property of all involved authors in equal parts. This work is predominantly integrated into Chapter 3 and Chapter 4.

1.4.3. Project Deliverables

The following project deliverables have been produced in the context of the SEMAFOUR project [SEM12].

1. Mehdi Amirijoo et al. *Demonstration Scenarios (updated version)*. Deliverable D3.4. SEMAFOUR Project, May 2015 [Ami+15]

   The author of this thesis, together with several fellow researchers, presents objectives and an approach for the design and implementation of a demonstrator based on use cases within the SEMAFOUR project. While this work is not directly integrated into this thesis, the ideas and concepts are relevant for the development of a SON management approach as described in Chapter 5 and Chapter 6. The author of this thesis has been significantly involved in writing chapter 7 of the deliverable, developing the ideas and concepts as well as the implementation of mentioned approaches.

2. Sana Ben Jemaa et al. *Integrated SON Management Requirements and Basic Concept*. Deliverable D5.1. SEMAFOUR Project, Dec. 2013 [Jem+13]

   The author of this thesis, together with several fellow researchers, describes the general architecture of an integrated SON management system. The author of this thesis has been significantly involved in writing chapter 2 of the deliverable. Presented concepts and ideas are mainly based on the authors patent application [SFL14]. This work is predominantly integrated into Chapter 3.

3. Lars Christoph Schmelz et al. *Integrated SON Management - Policy Transformation and Operational SON Coordination (first results)*. Deliverable D5.2. SEMAFOUR Project, Jun. 2014 [Sch+14b]

   The author of this thesis, together with several fellow researchers, describes a first approach to overcome the gap between operator objectives and SCV sets as well as a concept to coordinate SON functions. The author of this thesis has been significantly involved in writing chapter 2 of the deliverable. Presented concepts and ideas are mainly based on the authors publication [FLS14a]. This work is predominantly integrated into Chapter 4 and Chapter 5.

4. Dario Götz et al. *Integrated SON Management - Policy Transformation and Operational SON Coordination (first results)*. Deliverable D5.3. SEMAFOUR Project, Feb. 2015 [Göt+15]

   The author of this thesis, together with several fellow researchers, describes different approaches to overcome the gap between operator objectives and SCV sets as well as a concept to coordinate SON functions. The author of this thesis has been significantly involved in writing chapter 2 of the deliverable. Presented concepts and ideas are mainly based on the authors publication [FLS14b]. This work is predominantly integrated into Chapter 5 and Chapter 6.

5. Luis Campoy et al. *Integrated SON Management Implementation Recommendations*. Deliverable D5.4. SEMAFOUR Project, Aug. 2015 [Cam+15]

   The author of this thesis, together with several fellow researchers, describes implementation concepts and guidelines for an integrated SON management system. The author of this thesis has been significantly involved in writing section 2.1 and chapter 5 of the deliverable. Presented concepts and ideas are mainly based on the authors publications [FLS14b], [Sch+14a] and [Loh+15]. This work is predominantly integrated into Chapter 3.

6. Sören Hahn et al. *Final report on a unified self-management system for heterogeneous radio access networks*. Deliverable D6.6. SEMAFOUR Project, Aug. 2015 [Hah+15b]

   The author of this thesis, together with several fellow researchers, presents the final outcome of the SEMAFOUR project. The author of this thesis has been significantly involved in writing section 4.2 and subsection 5.3.5 of the

deliverable. Presented concepts and ideas are mainly based on the authors publications [FLS14b], [Sch+14a] and [Loh+15]. This work is predominantly integrated into Chapter 5 and Chapter 6.

2
Foundations

In this chapter technologies and theories will be presented playing a role for the further chapters of this thesis. Thereby, the aim of this chapter is not to provide an all-over summary of relevant technologies or theories but it focuses on those parts only which are necessary for a full understanding of the following chapters. In Section 2.1 the concept of SON in the context of mobile networks and associated mechanisms are presented. A technique which serves as a basis for Chapter 4 - Chapter 7 is Policy-based Management (PBM) further explained in Section 2.2. Finally, machine learning in general and a couple of supervised and unsupervised algorithms are presented which serve as a basis for Chapter 7 and Chapter 8.

2.1. Self-Organization in Mobile Networks

Radio networks, and mobile communication systems in particular, typically consist of a large number of elements interacting with one another. Their structure and the associated communication effort lead to a high system complexity. The goal of a mobile network is to connect User Equipments (UEs) to the internet via RATs. Therefore, base stations are placed in the environment which are connected via land-line to the bigger network. Commonly, each base station consists of three antennas, splitting the area around it into three cells inside which UEs can connect to. Cells are limited in physical size and the amount of UEs which can simultaneously connect to them. UEs can switch the cells they are connected to either because they move, or because it is within an area covered by two cells and these agree to perform a hand-over (HO). A customer desires a high bandwidth as well as no dropped calls or delays of text messages. This QoS in each cell is measured in KPIs and used to be collected through so-called drive tests, where sensor equipped vehicles took measurements of the network. However, nowadays these values are sampled directly at each UE and reported to the MNO in intervals, the so-called Granularity Period (GP). [Fre16]

These KPIs largely depend on the UE and the environment it is operating in, as well as the configuration of the antenna [Fre16]. Tall buildings or landscape features can block or weaken the connection [TOH14], and UEs close to the boundary between two cells might experience frequent unnecessary HOs [Fre16]. The antennas have NCPs such as base station transmission power, HO hysteresis or

remote electrical tilt [Fre+15]. These are the parameters which MNO personnel used to configure during the installation of a base station. However, as mentioned in Chapter 1, these tasks are very labor and knowledge intensive, and automating them is the goal of SON.

As also mentioned in Chapter 1, MNOs have to reduce costs in order to remain profitable on the contested market of mobile networks. Thereby, costs are subdivided into [HSS11]:

Operational Expenditures (OPEX) This sums up all costs which arise during the operation of a base station, e.g. expenditures for rent, electricity or maintenance.

Capital Expenditures (CAPEX) Expenses referring to material and equipment, e.g., cell towers and cables, are called CAPEX.

Implementational Expenditures (IMPEX) Costs for the installation and the initial operation are covered by the term Implementational Expenditures (IMPEX).

The most labor intensive task in current mobile networks is the manual OAM, which is why this area is ideal for automation and hence, ideal for a reduction of costs, i.e., OPEX.

SON Architectures

The traditional approach in dealing with radio networks is centralized OAM. A central OAM system is responsible for configuration and optimization of the network. Usually this OAM system is semiautomated and relies heavily on human expertise for planning and optimization, resulting in high costs, error-proneness and slow procedures [HSS11]. The goal of SON is to shift OAM tasks from human experts to the network itself leading to lower operating costs and reduced human errors. The network is provided with high-level guidelines and has to be able to achieve their realization in a self-organized manner. However, the human operator must be able to understand reconfigurations made by SON and if necessary revert them [HSS11].

As illustrated in Figure 2.1, SON architectures can be categorized into three different types, namely centralized, distributed and hybrid SON [ØG12].

Centralized SON In centralized SON architectures, requests, commands and parameter settings data is sent from the Network Management (NM) level to the Network Elements (NEs). Measurements and reports are passed from the NEs to the NM level. In the centralized approach SON algorithms operate on the NM layer, enabling them to utilize information from all the NEs. This approach usually results in more globally optimized solutions for the individual SON functions. Also, SON functions with conflicting objectives tend to be less of an issue. However, a centralized SON has longer response times, higher backbone traffic and poses a single point of failure.

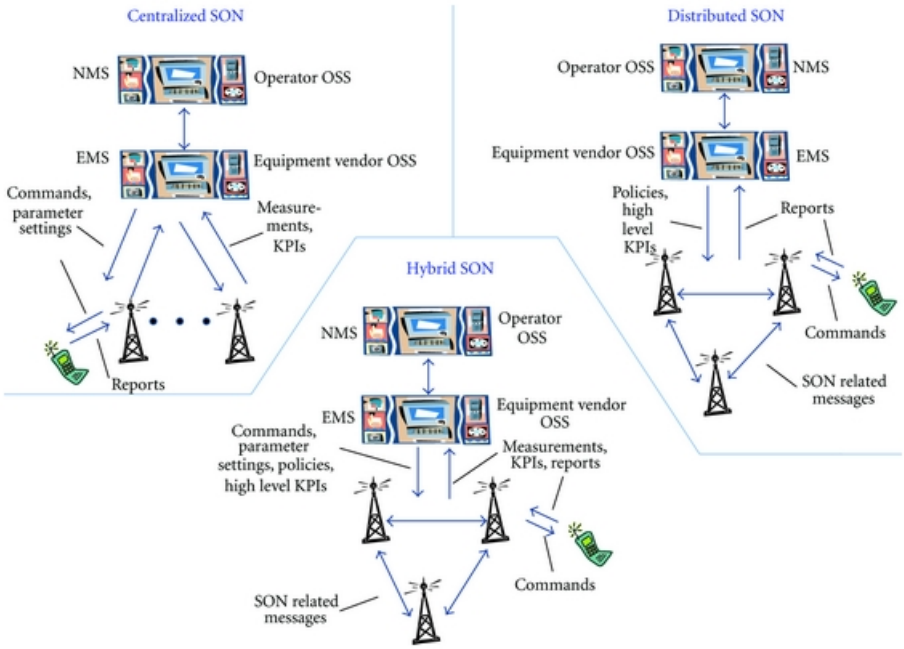


Figure 2.1.: Three types of SON architectures [ØG12]

Distributed SON The distributed SON architecture differs from the centralized one by applying SON algorithms directly on the network nodes, which communicate SON related messages among themselves. This results in quicker reaction times to changed conditions and better scaling capabilities. The main drawbacks of the distributed approach are, that optimizations done by network nodes do not necessarily optimize global network behavior and that the implementations of SON algorithms are manufacturer-specific, making third party solutions difficult. However, even in distributed architectures the NM system tends to be able to influence the behavior of SON functions, e.g., via the selection of optimization criteria.

Hybrid SON As the name suggests, hybrid SON architectures utilize the centralized and distributed approaches alike, meaning that SON algorithms run on the NM level as well as on NEs. Hybrid architecture approaches inherit the drawbacks of the centralized and distributed approaches alike while only benefiting from some of their advantages.

While all types of SON architectures have their own advantages and disadvantages, in this thesis only a centralized approach is relevant. This is due to the fact that first, one of the main objectives of this thesis is to find a global optimum in terms of SCV sets which fulfill operator objectives to the highest possible degree (cf. Section 1.2.1). Secondly, the MNO should keep control over the network which can be hardly achieved by a distributed or hybrid approach where SON functions act on the network nodes themselves (cf. Section 1.2.4).

2.1.1. Self-X Properties

The concept of SON can be split into three separate functional areas: self-configuration, self-optimization and self-healing, covered by the term self-X properties. All SON use cases, i.e., SON functions, are covered by and can be clearly assigned to one of these three areas. Figure 2.2 shows these three central attributes of SON.

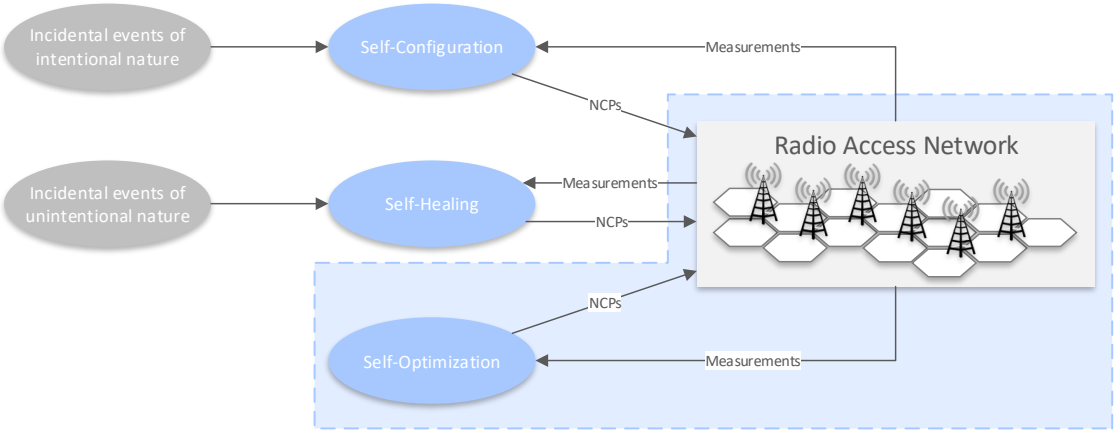


Figure 2.2.: Self-X properties in SON (adapted from [Mar+10])

Thereby, the dashed area marks the part where this thesis aims at: The optimization of the RAN by adjusting NCPs based on measurements from the network.

2.1.1.1. Self-Configuration

Self-configuration is the process that reduces human interventions in the field of deployment. Whenever a new NE is added to a system, it is necessary to configure this element by automatic installation procedures. This process is considered an "incidental event of intentional nature" [Mar+10]. Modern mobile SONs supporting a variety of different technologies such as EDGE, UMTS and LTE are networks with a constantly increasing rate of NEs making its installation and configuration an enormous factor of expense. The increasing rate is due to the fact that the trend is to decrease the cell size. Thus, a higher number of cells is needed [HSS11]. Therefore it is obvious that self-configuration can help to reduce CAPEX.

When having a closer look at the process of self-configuration, it consists of three basic phases. It starts with the auto-connectivity & security setup, establishing a secure connection between the NE and the underlying OAM system. The next step is auto-commissioning, consisting of the automated obtaining and testing of the involved software and NE configuration data. Finally, dynamic radio configuration builds the last step, whereat the state of the network deployment is needed to identify relevant configuration parameters and adapt them to the current state. [HSS11]

Typical SON use cases in this field are, amongst others [ØG12]:

- Physical Cell ID (PCI) Configuration
- Automatic Neighbor Relationship Setup (ANR)

2.1.1.2. Self-Optimization

Even though self-configuration provides an initial configuration of NEs, a network needs to be continuously improved. This is due to the fact that the traffic behavior changes and new traffic concentrations occur. Furthermore, the immediate use of new NEs after their deployment could negatively influence existing NEs since in some cases, it is not possible to integrate an NE without reconfiguring others. Therefore, self-optimization in SONs aims at autonomously maintaining and improving network quality and performance. Monitoring and analysis of performance data provides the basis for the optimization. Through self-optimization, the network is able to deal with volatile traffic conditions and thus improve the user experience.

While self-configuration and self-healing algorithms are usually waiting for a certain event as a trigger and only become active when such an event occurs, self-optimization algorithms are running at any time, observing the network in terms of KPI measurements and aiming for a steady improvement. NCPs are checked according to these measurements, i.e., KPIs, and SCV sets are adapted accordingly. Self-optimization algorithms thereby act in different time frames, i.e., it depends on the use case how often the corresponding SON function needs to be triggered.

Self-optimization use cases being identified by the 3GPP [3GP12] and by [HSS11] are, amongst others:

- MRO
- MLB and Traffic Steering
- Energy Saving (ES)
- CCO
- Random Access Channel (RACH) Optimization

2.1.1.3. Self-Healing

The larger a system, the more often failures occur independent of the degree of automation. A failure thereby mostly refers to the outage of a certain cell. If possible, failures should be diagnosed and handled automatically such that the network is able to remain in a working state without any human interventions. This process is called self-healing. Unlike self-configuration, self-healing is triggered

by "incidental events of unintentional nature" [Mar+10]. Sometimes a root cause analysis has proven to be difficult since by the majority of cases, the detection system was damaged such that the fault detection message could not be transferred. Hence, these so-called sleeping cells are then located by evaluating performance data requiring for a permanent interpretation [Nak+06]. A poor network performance would reduce the availability, reliability and QoS of a system, and, as a result, an MNO would have to accept financial penalties [HSS11]. Hence, it is necessary to react to failures as fast as possible which can be hardly achieved by human operators and legitimates the existence of self-healing algorithms.

SON use cases being identified by the 3GPP in this field are [3GP18c]:

- Self-Recovery of NE Software
- Self-Healing of board faults
- Self-Healing of Cell Outage, split into Cell Outage Detection (COD) and Cell Outage Compensation (COC)

2.1.2. Key Performance Indicators

A KPI is a measurement or a simple metric that quantifies the network performance in terms of a certain criterion. For instance, DCR indicates the number of calls that were not interrupted by end-users but due to a bad connection, compared to the total number of calls. There are various different KPIs, but in this thesis only the three KPIs described subsequently are of interest. The definitions are based on [HSS11] and [Fre16]. While Hand-over Ping-Pong Rate (PiPo) and Physical Cell Load (CL) KPIs are percentage values and hence, have a value interval of [0, 1] by nature, Channel Quality Indicator (CQI) has a domain of 0,...,5.5547 with 15 different values according to the 64QAM modulation technique [3GP18a].

Channel Quality Indicator (CQI) CQI is a value representing the connection quality to all of the UEs in a cell. It is influenced, among other factors, by obstacles in the environment. A higher value indicates a better signal quality.

Hand-over Ping-Pong Rate (PiPo) PiPo describes the maximum amount of ping-pongs between a cell and its neighbors. A ping-pong describes an unnecessary HO where a UE repeatedly gets assigned to the neighboring cell. These HOs are resource intensive and have detrimental effects on the overall QoS which is why a lower value is preferred.

Physical Cell Load (CL) CL serves as an indicator of how much workload the cell currently has and, therefore, how much workload this cell could further handle. Thus, it represents the quota of currently connected UEs in relation to the maximum possible connected UEs. If there are too many devices, the value does not increase past 1.0 (in contrast to another KPI, the virtual cell load, which takes unconnected UEs into account and therefore can have higher values). A lower value is preferred for CL.

2.1.3. SON Functions

> "Self organizing networks, SON, can be defined as a set of use cases that govern a network including the planning, set up and maintenance activities." [RAD18]

This definition describes, that the SON paradigm is a composition of different use cases in the three areas self-configuration, self-optimization and self-healing. Most of these use cases have been identified by the 3GPP, e.g., in [3GP18c] and [3GP12], but also research projects such as the Self-Optimisation and Self-Configuration in Wireless Networks (SOCRATES) project have investigated in this topic [Scu+08]. A SON use case describes a particular functionality that should be enabled in a SON by using self-organizational techniques. Enabling thereby means that each use case aims at achieving a certain objective.

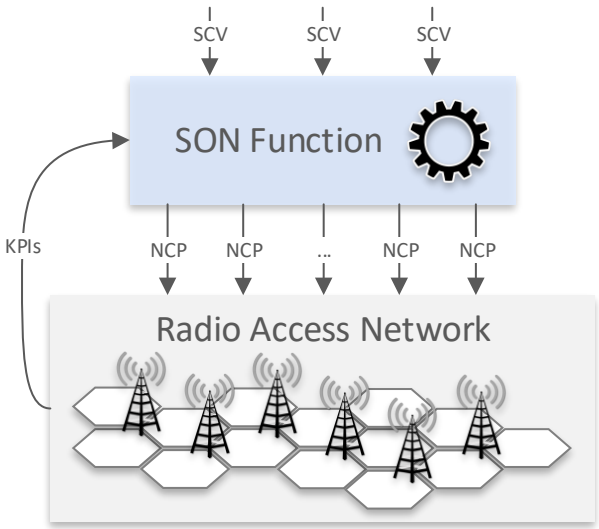


Figure 2.3.: Configuration of a RAN by means of SON functions

Self-optimization use cases are realized and implemented as SON functions which can be seen as a closed-loop control system, installed on each antenna and using feedback from the network in terms of KPIs [Sch+08]. SON functions are designed to act on cell level, meaning they can be configured for every single cell

independently and react to the cell's needs. Thereby, each SON function instance usually optimizes only for a single KPI, but may have an effect on a variety of KPIs. SON functions can be implemented in different ways, however, what the all have in common, is, that they can be configured via a set of input parameters, the so-called SCP. A value is assigned to each of the SCPs, the so-called SCV and hence, the sum of all parameter values for a SON function is an SCV set. The CCO, MRO and MLB functions as used in this thesis use a threshold and a step-size SCPs, represented through the respective SCV. At the end of each GP the respective KPI gets reported to the SON function and once it reaches the threshold, the SON function becomes active and changes an NCP based on the stepsize. This in turn affects the connection between the antenna and the UEs, therefore changing the KPI. The whole process is depicted in Figure 2.3.

Matching the three KPIs presented in Section 2.1.2, this thesis focuses on the three self-optimization SON functions CCO, MRO and MLB.

2.1.3.1. Coverage and Capacity Optimization

CCO has the goal to "provide sufficient coverage and capacity in the whole network area with minimal radio resources" [HSS11]. Thereby, coverage and capacity are adjusted according to changing environmental conditions like rush hours or temporary physical obstacles interfering with radio range [HSS11]. To this end, the two predominantly affected KPIs are the CQI reflecting signal quality and CL indicating the available capacity. CCO can be realized by tuning the NCPs antenna tilt and throughput power [BSR11]. If the CQI misses a predefined threshold, a higher signal quality may be achieved by changing the antenna tilt, meaning it changes the angle of the antenna in relation to the ground. If instead the load value exceeds a certain limit, CCO aims at activating small cells that can take over some UEs and thus reduce the load of the original cell [Fre16].

2.1.3.2. Mobility Robustness Optimization

MRO has the goal to ensure mobile UEs switch seamlessly from one cell to another whilst minimizing unnecessary ping-pongs, i.e., it aims at strengthening the network in terms of mobility. Poor HO parameter settings may lead to bad utilization of network resources which can cause various problems like HO failures, Radio Link Failures (RLFs) or ping-pongs [ØG12]. RLFs usually cause dropped calls, i.e., UEs loose connection, e.g., when moving out of a cells range. A HO ping-pong is the effect of UEs getting handed over to another cell and quickly afterwards handed back to the original cell. Hence, to accomplish a better performance and end-user quality, the MRO function observes the KPIs PiPo as well as DCR to monitor the cell's current state in order to effect an optimization if the indicators fall below a certain threshold. Therefore, if necessary, it adapts the NCP Cell Individual Offset (CIO), "which controls the selection of the serving cell by

the UE" [Fre16]. That is, the SON function tries to avoid unnecessary HOs by changing the virtual cell borders of specific cells. Thereby, MRO takes trade-offs with other SON functions (in particular MLB) into account. [HSS11]

2.1.3.3. Mobility Load Balancing

MLB is used to distribute traffic across the system's radio resources, i.e, its main goal is to push UEs from highly loaded cells to neighboring cells with more unused resources, provided that the UE is in reach of the neighboring antenna. It ensures a better performance and Quality of Experience (QoE) for end-users by balancing the load across cells in the network and thereby achieving a higher throughput for all UEs. Moving load between cells is triggered when the KPI CL is higher than a predefined threshold. In that case, the function tries to virtually shrink the size of the network cell by adapting the NCP CIO. Consequently, UEs outside of the now reduced virtual range are handed over to a neighboring cell and thus reducing the load of the former serving cell. This, however, also has an impact on the CL of the now serving neighboring cell. Of course this can only be reasonably applied when neighboring cells have capacities left. [Fre16] [HSS11] This process influences the HO performance and therefore coordination between MLB and MRO functions is required [ØG12].

2.1.4. SON Coordination

The more SON functions are deployed in the network, the more often conflicts between two or more SON functions may occur. Thereby, a conflict is a negative influence on one KPI or a set of KPIs. [Ban13] identified three different types of conflicts between SON functions:

Configuration Conflicts are the most obvious type of conflicts happening in case that different SON functions trigger contradictory actions in terms of NCPs, i.e., when more than one SON function want to adapt a specific NCP but in different ways. An increasing number of applied SON functions thereby increases the chance of conflicts. Possible configuration conflicts between the most common SON functions are depicted in Figure 2.4.

Measurement Conflicts describe a class of conflicts that occur when SON functions use performance measurements such as KPIs as input parameters. A situation where SON function actions influence input parameters that are used by the monitoring part of a SON function may lead to a partially or totally disjoint set of performance measurements.

Characteristic Conflicts appear when two SON functions both influence a specific cell characteristic, e.g., the cell size, but get different measurements as input and adapt different NCPs. This will inevitably lead to undesired

changes in KPIs that actually should not be touched by a certain SON function.

SON coordination tries to prevent or resolve SON function conflicts via low-level coordination of autonomously operating SON functions. SON coordination has to be distinguished from SON function co-design and SON function harmonization. All approaches essentially tackle the same problem by trying to prevent conflicts, but co-design tries to do so during design-time while harmonization is an approach for runtime conflict resolution, i.e., when conflicts have already appeared. Primarily, SON function co-design focuses on creating SON functions with disjoint output NCP sets [HSS11]. Thereby, the set of KPIs that is influenced by the SON functions should be as disjoint as possible. SON function coordination is required if conflicts occur during runtime and cannot be ruled out by co-design. It thereby provides a centralized component acting at runtime but, in contrast to SON function harmonization, is aiming at steering clear of conflicts in order to avoid a negative impact on the network performance beforehand [Ban13].

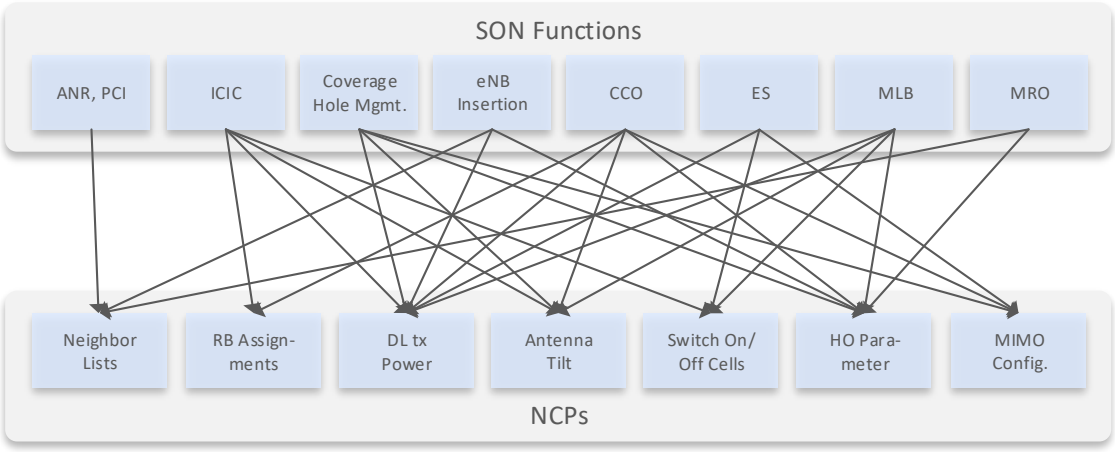


Figure 2.4.: Potential NCP conflicts between different SON functions (adapted from [Ban+11])

Whenever a SON function wants to perform a configuration change, it has to be checked first by the SON coordinator, that this will not lead to a conflict. SON coordination evaluates the current network state and decides whether a SON function is allowed to perform the NCP change or not. When a conflict is detected, it is up to the SON coordinator to approve the change request. This decision is based on a variety of guidelines. First, an MNO can define high-level requirements that influence the behavior of the coordinator via, e.g., thresholds for certain KPIs. Secondly, SON functions may be prioritized such that a configuration change is executed due to a higher priority in spite of possible conflicts. This way, SON coordination has full control over all configuration changes done by SON functions and is able to intervene at any time a conflict appears. [Ban13]

2.2. Policy-based Management

Mobile SONs need a technology that allows them to express directives on different levels. A common way to design and control networks are policies. This section first introduces the term policy in general and a special kind of policies, so-called Event Condition Action (ECA) policies. Subsequently, PBM is characterized and finally, the term policy continuum is defined in this context.

2.2.1. The Concept of Policies

Policies can be described as guidelines and constraints to system management. Thus, it is possible to transfer a system into a new state and to change its behavior. With these constraints, a target entity can be restricted without infringing given directives. Every entity has a set of attributes and values that belongs to those attributes. The constraints limit the value margin so as to leave the system in a reasonable state.

Along with constraints to system management, policies can also represent service requirements such as availability, response time, throughput or security. In this context, policies can be understood as constraints to system measurements. It is important to observe these so-called Service Level Agreements (SLAs) in order to avoid expensive SLA breaches. [USS07]

There are different ways to describe policies. The first option is to define a Domain-Specific Language (DSL) which is beneficial because a DSL focuses on relevant aspects and therefore uses a specific syntax. So concerning DSLs, a specific policy language can be created that reflects the requirements needed. Source code will then be generated out of those DSL models.

Another common way is to form policies subject to a predefined model such as the ECA model which has the following general form:

```
1 ON event IF condition THEN action
```

Listing 2.1: General form of an ECA rule

In that case, a policy obviously consists of three parts, the event, condition and action part. The event part specifies the signal that triggers the invocation of a rule. When such a signal appears, the conditions is checked. A condition is the logical part that can be composed of different sub-conditions. It decides whether it is essential to adapt the system behavior or not. Finally, when all sub-conditions are satisfied, the execution, represented by actions, is started. Any action is correlated to one or more parameters of an entity. By changing the parameter values, actions are the virtual part that puts the system into a new state. [RBS11] Furthermore, it is a benefit of ECA policies that, since they are widely used, they form the basis of most policy interpreters like Ponder2 [PON13] or [DRO18].

2.2.2. The Management Concept

Strassner [Str03] defines policy management as "the usage of rules to accomplish decisions". In [Ban+04], the importance of policy based approaches is justified by the fact that "they allow the separation of the rules that govern the behaviour of a system from the functionality provided by that system". Bringing together these two statements, PBM can be seen as a management paradigm that combines different views of abstraction in systems consisting of different layers, and that uses policies to enforce decision making in a system. In most cases, the highest layer is a business layer and the lowest layer is a more technical one. The number of layers thereby depends on its complexity. For simple systems, merely a technical and a business layer might be adequate. However, for larger systems it is necessary to have more than the two basic layers in order to represent relevant information about a system for a multitude of different target groups [RBS11].

Using this management approach it is possible to adapt a system's behavior at a high-level of abstraction by defining high-level policies. By refining these policies into low-level policies a more technical view on a system is provided. Low-level policies can then also be used to generate source code, preventing the labor-intensive and error-prone task of code adaptation. To enable a refinement, mappings are used to transform high-level elements into more technical low-level elements, thereby facilitating the management of the underlying system. However, monitoring is still necessary in order to align a system's behavior with sought goals. [Rom12]

The goals of policy refinement can be summarized as follows [MS93]:

- Identify resources demanded by the policy.
- Transform high-level business policies into low-level policies that can be automatically processed.
- Ensure that after the transformation, low-level policies still fulfill requirements defined for high-level policies.

For the realization of a PBM system, some requirements have to be met. First of all, a business policy language and an according policy meta model, for instance the above described ECA model, need to be selected or specified. Furthermore, expertise knowledge for every single layer is necessary in order to bridge the semantic gap between the different layers of abstraction. Above all, completeness is an important factor. Hence, administrators of particular layers need to have expertise knowledge of other layers in order to avoid fragmentary or wrong refinement processes between neighboring layers. [Rom12] In order to avoid the completeness problem, automatically generated mappings are a reasonable solution. Therefore, policies do only have to be specified at a high level of abstraction and are further refined into lower layer policies via generated mappings.

There is a variety of approaches concerned with the refinement of policies such as a model-driven approach [Rom12], a goal-based approach [Ban+04], a classification-based approach [USS07], an expertise knowledge-based approach [Eck07] or an ontology-based approach [Gue+06]. How far these approaches are relevant for this thesis, is discussed in Chapter 3. However, a characteristic that all these approaches have in common, is the fact that they describe policies on different layers of abstraction. Section 2.2.3 deals with the specific abstraction layers of the refinement process as a unity according to the definition of [Str+06], the so-called policy continuum.

2.2.3. Policy Continuum

As outlined above, different constituencies require different perspectives on a system, more precisely, one perspective for each constituency is needed. This demands a consistent approach to PBM in order to enable deployment at any time. Three prerequisites must be met in order to guarantee this [Str+06]:

- A policy language must be determined for the definition of policies on all abstraction layers.
- High-level policies are stepwise refined into lower-level ones down to the point of automatically processable policies.
- Network nodes are configured according to policies defined on the highest abstraction layer.

The challenge is to solve these problems as a collective in order to achieve a consistent approach. The result is a paradigm, where every layer uses its own grammar and terminologies to define policies at a particular abstraction layer. However, these policies can be linked with each other through model mappings. Figure 2.5 shows the layers identified by the authors of [Str+06], all together forming the policy continuum. The highest layer represents a business view containing

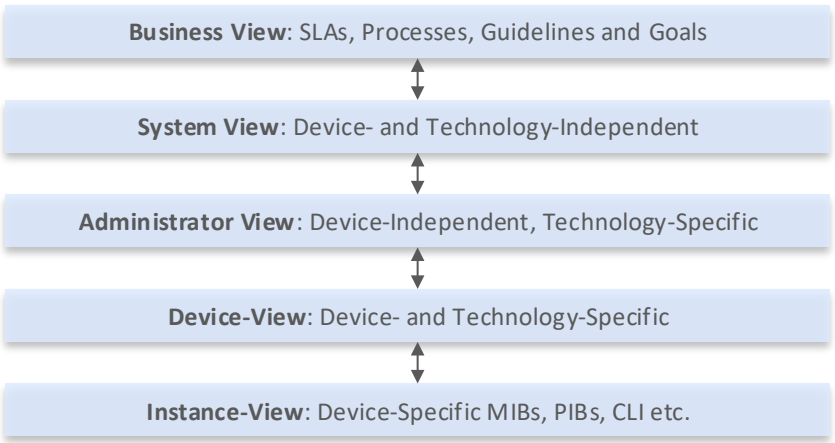


Figure 2.5.: The policy continuum (adapted from [Str+06])

high-level SLA information, processes, general guidelines and goals. A user interested in the business view does not need low-level information. Correspondingly, a user focusing on an instance view is not interested in high-level SLAs. The instance view represents the lowest layer which is more technical and closest to the source code. Between these two layers, three other layers exist representing policies for system, administrator and device view users.

In Chapter 3 it is investigated which layers are relevant in the course of this thesis and the proposed solution is related to the policy continuum.

2.3. Machine Learning

In [Mur12], machine learning is defined as an "automated method for data analysis" that "detects pattern in data" and uses this information to "predict future data or other outcomes of interest". That means, the general goal of machine learning is to build models of systems which allow the computer to perform tasks it was not explicitly programmed for. To that end, data is first collected from those systems and then analyzed. According to [Jam+13] any system can be captured through the term:

$$Y = f(X) + \epsilon \tag{2.1}$$

Thereby, Y refers to the outcome or reaction of a system (the dependent variable), produced by a function $f()$ with an input or state vector X (the independent variable) containing a set of features $(X_1, X_2, ..., X_p)$. ϵ denotes the random noise which has to be independent of X and has an average of zero. By using methods in the field of supervised, unsupervised and reinforcement learning, predictions can be made by building a model of the form

$$\hat{Y} = \hat{f}(X) \tag{2.2}$$

where $\hat{Y}$ represents the outcome or prediction when $\hat{f}()$ is the estimate of $f()$ and X the given input vector. Such models can be used to predict the behavior of a system under unseen circumstances or to gain a better understanding of a system by finding patterns in its behavior.

In [BHH05], the authors state that:

> "The most that can be expected from any model is that it can supply a useful approximation to reality: All models are wrong; some models are useful."

This means that the predicted output $\hat{Y}$ will never exactly match the expected output Y for all inputs X. However, enabling a perfect prediction is not the aim of machine learning. Instead, machine learning shall alleviate to generate models which allow data mining to identify relations between inputs and outputs and thereby potentially reducing unknown risks and insecurities.

Generally, the field of machine learning is divided into three areas where each of them faces a different problem setting [HSS11]: While supervised learning gets annotated data as input and tries to find the best fitting model for these data, unsupervised learning tries to find structures in unannotated data. The last type of machine learning, reinforcement learning, learns from experience in an interactive environment. Subsequently, a selection of supervised and unsupervised learning algorithms that are applied in the context of this thesis, are described in more detail. Even though there are several approaches using reinforcement learning techniques in SONs (cf. Section 7.4), these algorithms are not in the focus of this thesis and hence, are not described in further detail. This is due to the fact that the management system and all its actions should remain transparent for an MNO such that he or she can interrupt them at any time which is not able any more when using reinforcement learning algorithms.

2.3.1. Supervised Learning

Supervised learning needs labeled data, meaning it needs to know the true output for given inputs. Referencing back to Equation 2.2, supervised models are trained on datasets of the form $\{(X_1, Y_1), (X_2, Y_2), ...\}$, with X_i representing the input and Y_i the associated output or label. The quality of a model built through supervised learning can then be checked through comparing the predicted $\hat{Y}$ to the actual Y (hence supervised). Once a satisfying model is built, it can then be used to predict the behavior of the actual system under unseen inputs. Depending on the type of labels, there are two sub-types [Jam+13]:

Regression Learning is used for labels with continuous values where the algorithm is asked to predict continuous results, such as, e.g., housing prices depending on the amount of rooms. Hence, the quality of a model can be measured by error metrics indicating the deviation from the actual values.

Classification Learning is used for labels with discrete values (classes) where the algorithm is asked to classify samples, such as, e.g., the contents of a picture. Here, the quality of a model is measured by its accuracy, meaning there is always either a wrong or a right classification.

In this thesis the focus is on regression learning since KPI values shall be predicted which are continuous values.

Quality of Fit

For a comparison of models, it is necessary to evaluate the fit of a model to the available data [Jam+13]. There are various different metrics to accomplish this, each with their own advantages and drawbacks. Nevertheless, it will be focused on the most widespread practices. How well a model describes the data can

be measured in residuals (ϵ_i), the difference between a known output y_i and a predicted output $\hat{y}_i$ for given inputs:

$$\epsilon_i = y_i - \hat{y}_i \tag{2.3}$$

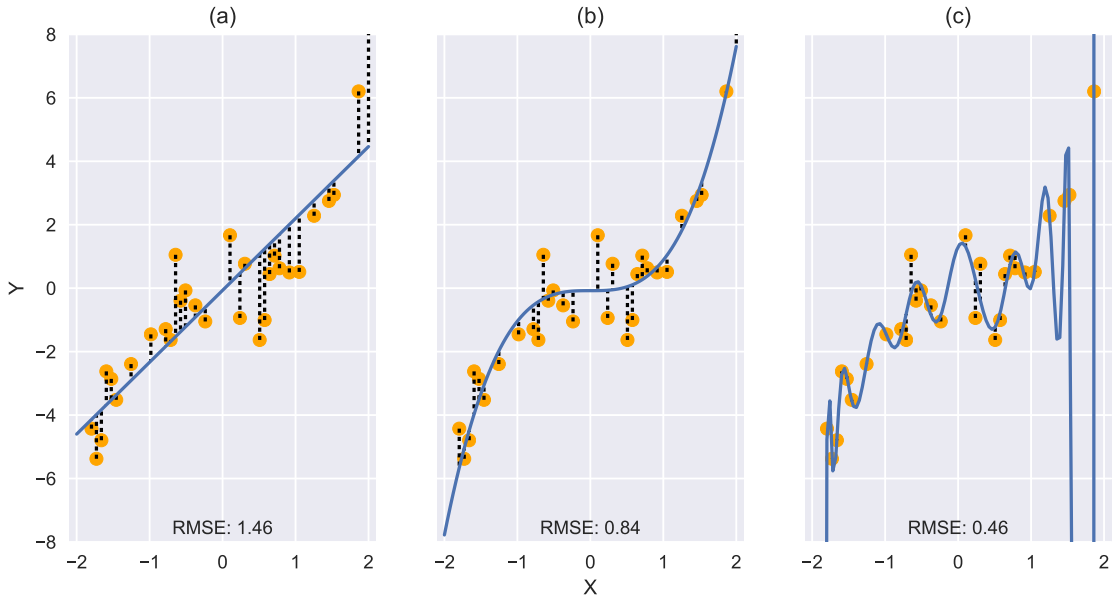


Figure 2.6.: Fitting a linear (a), a cubic (b) and a 22nd degree polynomial (c) to noisy data and reporting the Root Mean Squared Error (RMSE) for each model

In Figure 2.6 three different linear regression models are fit to the same dataset, with the dotted vertical lines representing the residual for each sample. The following ways are the most commonly used ones to measure the overall error of a model:

Mean Absolute Error (MAE) The simplest error measure is to average over all differences between measurement and prediction, with the result being in the units of the system output, and therefore interpretable [WM05].

$$MAE = \frac{1}{n}\sum_{i=1}^{n}(|y_i - \hat{f}(x_i)|) \tag{2.4}$$

Mean Squared Error (MSE) Compared to the Mean Absolute Error (MAE), the Mean Squared Error (MSE) has the advantage of penalizing bigger residuals more, therefore providing a better distinction between models that are completely off and models that are fairly close. However, the value does not have the same units as the system outputs, and therefore is less interpretable by the user [WB09].

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{f}(x_i))^2 \tag{2.5}$$

Root Mean Squared Error (RMSE) Compared to the MSE, the RMSE is interpretable in the actual units of the output, thus being easier to understand while still retaining the high penalty for larger residuals [WM05].

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{f}(x_i))^2} \tag{2.6}$$

After an error metric is selected, different models with different parameters and different degrees of flexibility can be compared and tuned in order to reduce the error, and consequently lead to a model which better captures the behavior of the underlying system.

Nonetheless, searching for a model with zero error is not the way to generate successful predictions, which is shown in the following section. This is due to the fact that all measurements Y are tainted by the error ϵ when measured: $Y = f(X) + \epsilon$. Since ϵ is random, it can not be included in the model $\hat{f}()$. Hence, it is logical to split the residuals into

$$|Y - \hat{Y}| = \epsilon_{reducible} + \epsilon_{irreducible} \tag{2.7}$$

with $\epsilon_{reducible}$ referring to the reducible error induced through wrong models and $\epsilon_{irreducible}$ referring to errors from the random noise in the system [Jam+13].

For instance, panel (b) in Figure 2.6 actually shows the perfect RMSE for this system. Both the model and the system are $f(X) = x^3 = \hat{f}(X)$, any remaining residuals stem from the noise pollution.

Bias vs. Variance

When selecting a model, one has to balance the bias and variance it provides. According to [Jam+13], "bias refers to the error that is introduced by approximating a real-life problem, which may be extremely complicated, by a much simpler model". Panel (a) in Figure 2.6 shows an already fairly biased model with only two degrees of flexibility. In contrast, variance describes how much a model changes when fit to a different dataset [Jam+13]. Panel (c) in Figure 2.6 shows a model with a high variance, one can easily imagine how the graph would have a very different form if it were initially fitted to another set of points sampled from the same system. In fact, when applying the model (c) to a new set of samples, as seen in Figure 2.7, the residuals to these new data points are very large, significantly larger than for the other two models. This effect is called over-fitting which is further explained in Section 2.3.1.

In summary, both a too biased model (horizontal line) and a too flexible model (degrees of freedom = amount of samples) have shown to have flaws when trying to build a reliable model.

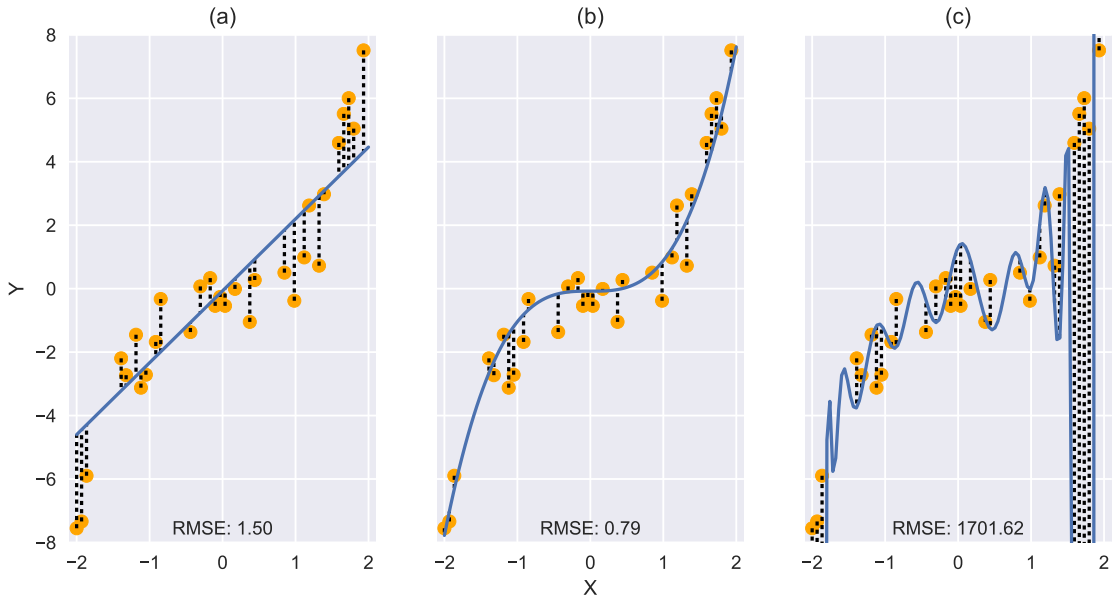


Figure 2.7.: Testing the models from Figure 2.6 on unseen data and reporting the RMSE

Training and Testing

In the paragraphs above it is explained how a model fitting the data perfectly is not desirable compared to a model with more bias, and how to compare models to each other based on their error metrics. Since it is impossible to know how much of an error is due to $\epsilon_{reducible}$ and how much due to $\epsilon_{irreducible}$ without explicitly knowing $f()$, an extra step is necessary to evaluate the true performance of any given model regarding its original goal: Predicting outputs of unseen inputs and comparing these to the measured outputs [Jam+13].

In Figure 2.7 the models fitted in Figure 2.6 are tested on data samples which the models had not been exposed to during fitting, therefore each model is asked to predict the output of unseen inputs. Here is where the drawbacks of the high polynomial model become apparent: Polynomials of high degrees are able to capture the training data very well, but show an increased amount of jitter in areas without data points. These high variations in $\hat{y}$ lead to big residuals when tested on new data, an effect called over-fitting: The model describes the data presented during fitting so well, that it incorporates the $\epsilon_{irreducible}$ in its predictions and fails to capture the underlying $f()$.

But how does one detect that a model is over-fitted before actually applying the model in production? The solution is to not fit the model on all available data from the start. Instead, the dataset is split into training data and testing data [Bis06]. The model is then trained by fitting it to the training data, before it is tested against the remaining testing data. All models are then compared on the basis of their performance on the test dataset. Over-fitted models will have small residuals on the training data, but are too sensitive to the noise and fail on the

testing data. This allows the user to select the best model which then can be used in production and trained on all available data.

Resampling Methods
In order to gain information about the quality of a certain model, it has to be consistently validated by sampling the training data. This process is called resampling. The two most commonly used methods are K-fold cross-validation and bootstrapping [Jam+13] while only K-fold cross-validation is relevant for this thesis.

K-fold Cross-Validation Thereby, the training data is divided into K evenly sized groups. Subsequently, the model is trained with data of $K-1$ groups and the excluded group G_0 is used for the validation. Afterwards, $f(x)$ and $\hat{f}(x)$ of every instance $x \in G_o$ are compared with each other and the error of the estimation is measured. This procedure needs to be done for every group and so, the overall error E can be calculated.

[Han+11] Leave One Out (LOO) cross-validation is a special case of cross-validation where K equals the number of observations. That means, the model is trained on $K-1$ observations and asked to predict the value of the Kth sample leading to K residuals. Hence, the overall error rate CV_K can be determined by averaging over the error rates of all K groups (cf. Equation 2.8) [Jam+13].

$$CV_K = \frac{1}{K}\sum_{i=1}^{K} E_i \tag{2.8}$$

Bootstrapping "uses the computer to "resample" an original sample extensively, inductively arriving at an estimate of a statistic's sampling distribution" [MD93]. That means, bootstrapping calculates statistics by repetitively drawing a sample and thereby estimating the distribution of samples in a dataset. This approach is useful to make inferences about a dataset with an unknown distribution of the samples' characteristics. An advantage of this approach lies in the fact that no strong assumptions about the sample distribution in the dataset under investigation have to be made beforehand. [MD93]

In the following, four supervised learning algorithms are presented in more detail. In general, these algorithms can be separated into parametric and non-parametric approaches. While in non-parametric models the number of parameters is not fixed, meaning it grows with the amount of training data and thereby is more flexible, parametric models have a fixed number of parameters leading to less flexible however also computationally less complex models [Mur12]. Linear Regression (LR) can be clearly sorted in parametric methods while k-nearest Neighbors Regression (KNN) as well as Gaussian Process Regression (GPR) are denoted as non-parametric approaches. Artificial Neural Networks (ANNs) are

classified somewhere in between: Usually, they are denominated as parametric, however, there are also approaches for non-parametric neural networks available [PC17].

2.3.1.1. Linear Regression

LR takes an input vector X with n features $\{x_1, x_2, ..., x_n\}$. The model built by LR then has the following form:

$$\hat{y} = \beta_0 + \beta_1 * x_1 + \beta_2 * x_2 + ... + \beta_n * x_n = \hat{f}(X) \tag{2.9}$$

Fitting a LR model equals estimating the coefficients $\{\beta_0, ..., \beta_n\}$. In order to do so, there needs to be a labeled dataset with k observations of the form $\{(X_1, Y_1), (X_2, Y_2), ...\}$, with $k \geq n$, otherwise Equation 2.9 can not be solved. If $k = n$, then there is precisely one solution for each β. If $k > n$, then it is in most cases not possible to find βs such that each sample can be described with zero residual [Jam+13].

To fit $\hat{f}()$ to the dataset, the least squares method is employed in order to reduce the residuals [Jam+13]. Since the model for LR $\hat{f}()$ was defined in Equation 2.9, it is now possible to merge it with Equation 2.3 into the following:

$$\begin{aligned} \epsilon_i &= y_i - \hat{f}(X_i) \\ \epsilon_i &= y_i - (\beta_0 - \beta_1 * x_{1,i} - \beta_2 * x_{2,i} - ... - \beta_n * x_{n,i}) \end{aligned} \tag{2.10}$$

The goal is to determine the set of $\beta_0, \beta_1, ...$ which have the minimal Residual Sum of Squares (RSS) which is defined as the sum over all squares of each available residual:

$$RSS = \epsilon_1^2 + \epsilon_2^2 + ... + \epsilon_k^2 \tag{2.11}$$

Inserting Equation 2.10 into Equation 2.11 leads to:

$$RSS = \sum_{i=1}^{k} (y_i - (\beta_0 - \beta_1 * x_{1,i} - \beta_2 * x_{2,i} - ... - \beta_n * x_{n,i}))^2 \tag{2.12}$$

Since there are more data points k than unknown coefficients β, the equation has more than one solution. Through derivation and some calculus the optimal set of βs can be calculated [Jam+13].

Increasing Flexibility
So far only linear combinations of the different features $\{x_1, x_2, ...\}$ of X have been considered. However, in Figure 2.6 and 2.7 (panel (b) and (c)) polynomials with higher degrees than 1 as model have already been shown. According to [Bis06] this is still considered to be LR, since the polynomials can just be considered as additional features, where X_{poly} now consists of $\{x_1, x_1^2, x_1^3, ..., x_2, x_2^2, ...\}$. This transformation of X enables more complicated models while still retaining the same procedure as with classical LR.

In the same vein as adding polynomials of features, it is quite reasonable to expect that certain inputs have synergies with each other and influence the output to a different extent than each of them does individually. An example would be the growth rate of a plant g based on the amounts of rain r and sunshine s. r and s individually have measurable effects on the growth rate of a plant, but when combined the actual growth rate of the plant can be described much more accurately: $g = \beta_0 + \beta_1 r + \beta_2 s + \beta_3 * r \times s$. These so-called interaction terms (and their polynomials) can then be used to extend the input vector X and as a result lead to an even more flexible model [Jam+13]. Advantages and drawbacks of an increasingly more flexible model have already been discussed in Section 2.3.1.

2.3.1.2. k-nearest Neighbors Regression

The basic idea of the KNN algorithm is that for every instance x, averaging $f(y)$, whereby y belongs to the k closest instances to x, approximates $f(x)$. This algorithm is an example of a non-parametric regression [Mur12]. According to [Mit97], it is the most basic instance-based method. Hence, unlike batch learning, the instances of the training data are considered one after another. Therefore, in instance-based approaches, different approximations to the target function can be constructed for every distinct instance [Mit97]. Especially, for complex target functions mapped by a collection of easier local functions, this is a huge advantage. On the other hand, these numerous computations consequently are accompanied by high costs. Furthermore, KNN is not robust towards unneeded computations if a new instance is considered that is quite similar to a previous instance [Mit97]. These facts get clearer by the explanation according to [Mit97]:

$$x \equiv \langle a_1(x), a_2(x), ...a_n(x) \rangle \tag{2.13}$$

x represents an instance as a vector of attributes whereas $a_r(x)$ $(r, n \in \mathbb{N} \wedge 1 \leq r \leq n)$ is the r^{th} attribute of the instance x. Since the algorithm needs to find the closest or rather most similar instances $Y_{x,k}$ out of all distinct instances I for a given instance x, the difference d between two instances needs to be defined. There are many ways to measure the similarity of instances. One of the most commonly

used ones is the euclidean distance

$$d_{euc}(u,v) = \sqrt{\sum_{r=1}^{n} (a_r(u) - a_r(v))^2} \tag{2.14}$$

where u and v are instances and a is the value of a feature r, hence, n the number of features.

Approximating $\hat{f}(x)$ for $f(x)$ comes along by computing $Y_{x,k}$. The calculation of $d(x, x_i)$ for every $x_i \in I$ determines $Y_{x,k}$.

$$\hat{f}(x) = \frac{\sum_{y \in Y_{x,k}} f(y)}{k} \tag{2.15}$$

$$Y_{x,k} = \{y \in I : |\{(a,x) \in I^2 : d(a,x) \leq d(y,x)\}| \leq k\} \tag{2.16}$$

Furthermore, a refinement for KNN is depicted in [Mit97]: Within the distance-weighted nearest neighbor estimator $\hat{f}_{dwg}$, also known as locally weighted regression, the distance of each of the k-nearest instances is taken into account. Thus, instances that are closer to x have a greater impact than instances that are further away from x:

$$\hat{f}_{dwg}(x) = \frac{\sum_{y \in Y_{x,k}} f(y) w(x,y)}{\sum_{y \in Y_{x,k}} w(x,y)} \tag{2.17}$$

Thereby, the impact is included by the function $w : I^2 \to \mathbb{R}^{\geq 0}$ taking the distance d between two instances into account.

$$w(u,v) = \frac{1}{d(u,v)^2} \tag{2.18}$$

As the name of the algorithm indicates, the parameter k is a crucial factor for this method [Han+11]. In [KNN10] the author attributes a high influence to noise if k is a small number whereas assigning k with a larger value increases the computational cost as well as disagreeing with the basic idea of KNN due to the diminishing difference to the mean value. There are several ways to determine k. A very simple approach is to define k by the number of training instances N: $k = \sqrt{N}$ [KNN10]. Another one is the K-fold Cross-Validation (cf. Section 2.3.1) building the model for all the potential values of k and determining k by the minimal global error E.

2.3.1.3. Gaussian Process Regression

The GPR, also known as kriging, is like an interpolation of a function $f : \mathbb{R}^n \to \mathbb{R}^z$. Wilson and Adams define a Gaussian process in [WA13] as "a collection of random variables, any finite number of which have a joint Gaussian distribution".

$$f(x) \sim GP(m(x), \kappa(x, x')) \tag{2.19}$$

Thus, the basis of this algorithm is the assumption that functions $f(x)$ comes from a Gaussian process parametrized with the mean function $m(x)$ and covariance kernel $\kappa(x, x')$. Due to the usual assignment of $m(x)$ to null that is proposed in [RW06] any collection of function values has the following joint Gaussian distribution [WA13]:

$$\begin{bmatrix} f(x_1) \\ \vdots \\ f(x_m) \end{bmatrix} = \begin{bmatrix} y_1 \\ \vdots \\ y_m \end{bmatrix} = y \sim \mathcal{N}(0, K) \tag{2.20}$$

This multivariate normal distribution described in Equation 2.20 is defined by null as m-dimensional mean vector and K as $m \times m$ covariance matrix [Mur12].

$$K = \begin{bmatrix} \kappa_{11} & \kappa_{12} & \cdots & \kappa_{1m} \\ \vdots & \vdots & \vdots & \vdots \\ \kappa_{m1} & \kappa_{m2} & \cdots & \kappa_{mm} \end{bmatrix} \tag{2.21}$$

Therein, $\kappa_{ij} \equiv \kappa(x_i, x_j)$ maps a kernel function measuring the similarity of two instances x_i and x_j [RW06]. The choice of the kernel function significantly influences the results predicted by the GPR and hence, is a crucial step for a good quality of the results. There are many different kernel functions which are applicable in different use cases. In this thesis, only kernel functions able to be used for real-valued data [AY14] are relevant and presented subsequently:

Radial Basis Function Kernel One simple example is the Radial Basis Function (RBF) kernel κ_{RBF} with the free parameter λ as bandwidth [Mur12]:

$$\kappa_{RBF}(x_i, x_j) = exp\left(-\frac{||x_i - x_j||^2}{2\lambda}\right) \tag{2.22}$$

Polynomial Kernel Another widely used kernel is the polynomial kernel function κ_{poly}. When d is set to 1 in Equation 2.23, one can get the linear kernel function. k thereby is the constant. [AY14]

$$\kappa_{poly}(x_i, x_j) = (x_i.x_j + k)^d \tag{2.23}$$

Pearson VII Universal Kernel A kernel function which is mostly used in the area of support vector machines, is the Pearson VII Universal Kernel (PUK) function. This functions offers a high degree of flexibility by changing parameters λ and ω. [AY14]

$$\kappa_{PUK}(x_i, x_j) = 1/\left[1 + \left(\frac{2\sqrt{\|x_i - x_j\|^2}\sqrt{2^{(1/\omega)} - 1}}{\lambda}\right)^2\right]^{\omega} \tag{2.24}$$

Following the principal of the Gaussian process it enables to predict $f(x_*) = y_*$ for a x_* that is not in the data by assuming $f(x_*)$ is a sample of $\mathcal{N}(0, \kappa_{**})$ [RW06].

$$\begin{bmatrix} y \\ y_* \end{bmatrix} \sim \mathcal{N}\left(0, \begin{bmatrix} K & K_{i*} \\ K_{*i} & \kappa_{**} \end{bmatrix}\right) = \mathcal{N}\left(0, \begin{bmatrix} K & \begin{bmatrix} \kappa_{1*} \\ \kappa_{2*} \\ \vdots \\ \kappa_{m*} \end{bmatrix} \\ \begin{bmatrix} \kappa_{*1} & \kappa_{*2} & \cdots & \kappa_{*m} \end{bmatrix} & \kappa_{**} \end{bmatrix}\right) \tag{2.25}$$

Based on this, the Cholesky theorem (cf. [Pre+07]) allows to calculate the expected value μ_* including the confidence interval c_* for $f(x_*) = y_*$ [RW06].

$$\mu_* = E(f(x_*)) = K_{i*}^T K^{-1} y \tag{2.26}$$

$$c_* = \kappa_{**} - K_{i*}^T K^{-1} K_{i*} \tag{2.27}$$

For a better understanding, Figure 2.8 illustrates GPR. Whereas Figure 2.8 (a) only depicts a certain example for the prediction of x_* that is between x_2 and x_3, Figure 2.8 (b) is extended to a differentiable graph. Thereby, the dashed line visualizes the target function's true course only represented by four samples marked as bullets. Additionally, the filled area illustrates the confidence interval for the expected values of the prediction portrayed as the continuous line. Analogously, the triangle indicator surrounded by two bows in Figure 2.8 has the same meaning.

2.3.1.4. Artificial Neural Networks

According to [Aga+09], "a neural network is defined as a powerful data modeling tool that is able to capture and represent complex input/output relationships". Thereby, in contrast to, e.g., LR, ANNs are a good way to model non-linear relationships in a dataset. The idea of ANNs is to copy the functional principles of biological neural nets as part of the nervous system. Biological neural nets basically consist of linked cells processing signals and communicating with each

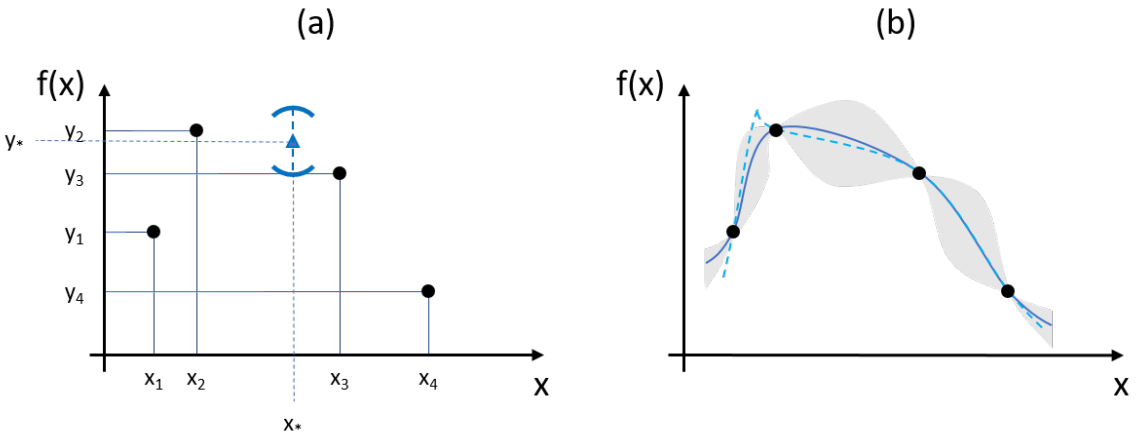


Figure 2.8.: Predicting an exemplary point in (a), illustration of a GPR prediction graph in (b)

other. While biological neurons are inferior to logic gates in terms of computation time, the human brain can still deal with problems that are unsolvable for modern computers. This is mainly due to the massively layered networking happening in biological neural nets. Modeling this complex system as an artificial net is exactly what ANNs are aiming for. [Roj96]

The Perceptron

The perceptron replicates the functionality of the smallest unit in biological neural networks, the neuron. It consists of a number of input values $X = \{x_1, x_2, ..., x_n\} \in \{0,1\}$, weights $w = \{w_1, w_2, ..., w_n\}$ and a threshold t. It generates an output (or activation) $a \in \{0,1\}$ by multiplying every input value with its respective weight and determining whether the result is greater than the threshold value.

$$a = \begin{cases} 1, \text{if } \sum_{j=1}^{n} w_j x_j > t \\ 0, \text{if } \sum_{j=1}^{n} w_j x_j \leq t \end{cases} \tag{2.28}$$

Using the bias $b = -t$ and the dot product $\sum_j w_j x_j = w \cdot x$ for vectors w and x, this can also be written as:

$$a = \begin{cases} 1, \text{if } w \cdot x + b > 0 \\ 0, \text{if } w \cdot x + b \leq 0 \end{cases} \tag{2.29}$$

Perceptrons can be used for decision making or to calculate logic operations. However, they are limited to linear problems. Networks can be constructed by using multiple perceptrons in different layers, taking the output values of one layer as input values for the next one. They can be adjusted to match a desired behavior by changing the weights of the perceptrons. This process of modifying weights is what is considered as learning in the context of ANNs. [Nie15]

The Sigmoid Neuron
With their input and output limited to 0 or 1, perceptrons might not be ideal for learning a model. A small change of weights can completely flip the output. Ideally, a slight adjustment of weights should cause an accordingly small change of the output.

The sigmoid neuron is designed to overcome this weakness. Just like the perceptron, the sigmoid neuron has weights and a threshold or bias. The difference is, that the input values can be any real value between 0 and 1. Thus, the weights are $w = \{w_1, w_2, ..., w_n\}$, the input values $X = \{x_1, x_2, ..., x_n\} \in [0,1]$, the threshold t, the bias $b = -t$ and the output $a \in [0,1]$. The output is calculated by using an activation function. Three popular activation functions are [UFLDL]:

Sigmoid Function This activation function is defined as:

$$\sigma(z) = \frac{1}{1+e^{-z}} \tag{2.30}$$

Hyperbolic Tangent Function This activation function is defined as:

$$tanh(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}} \tag{2.31}$$

Rectified Linear Function This activation function is defined as:

$$f(z) = max(0, z) \tag{2.32}$$

While the hyperbolic tangent and rectified linear function are not part of the sigmoid neuron model, their resemblance to the sigmoid function gets apparent in Figure 2.9. The hyperbolic tangent looks very similar to the sigmoid function, however differs in range. The rectified linear activation function has proven to be the best choice for deep architectures in the majority of cases [Aga+09].

Using the sigmoid function allows to match the desired behavior. For small changes Δw_j and Δb of weights and bias, a small change Δy of the output is received. The following approximation holds [Nie15]:

$$\Delta a \approx \sum_{j \in W} \frac{\partial a}{\partial w_j} \Delta w_j + \frac{\partial a}{\partial b} \Delta b \tag{2.33}$$

The sigmoid can be seen as an improved version of the perceptron, that generally acts like a perceptron, but is easier to handle in terms of choosing and adjusting weights and bias. The perceptron's outputs can simply be treated as 'yes' or 'no'. Since the sigmoid does not just output a 0 or 1, results have to be dealt with in a different way. The easiest and probably most obvious way is to create some sort of scoring model. For instance, if the score is higher than 0.8 then output 'yes' otherwise 'no'. In some cases the real value of the output can come in handy, for instance when a percentage value is required, rather than a boolean [Nie15].

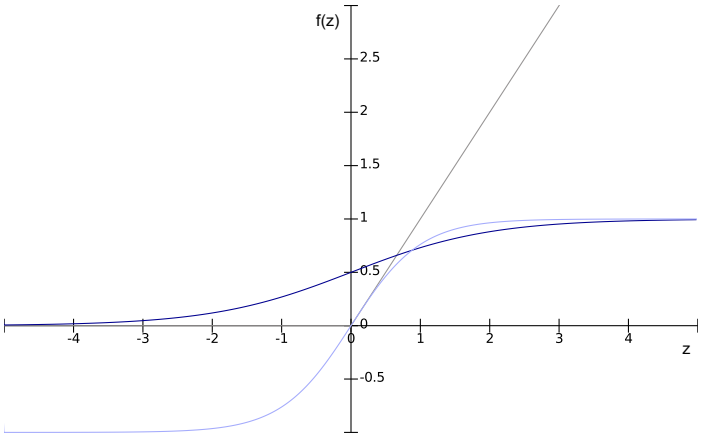


Figure 2.9.: The sigmoid function (dark blue), the hyperbolic function (light blue) and the rectified linear function (gray)

Multilayer Neural Networks

Multilayer Neural Network (MNN) are made out of several layers of artificial neurons. As shown in Figure 2.10, the first layer of neurons in an MNN is called the input layer, accordingly the last one is named the output layer. All the layers in between are called hidden layers. The signals are forwarded through the network layer by layer until the output layer is reached. The results of the output layer are then used to make the final prediction.

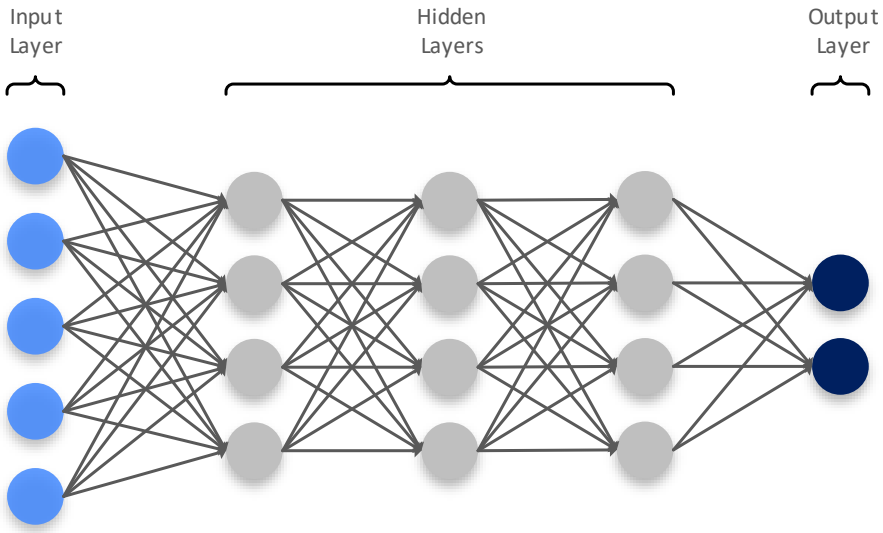


Figure 2.10.: An abstract exemplary MNN architecture containing three hidden layers with four neurons each

A simple version of an MNN is the Multilayer Perceptron (MLP), that, as the name suggests, uses perceptrons as building blocks for the network. MLPs are designed for supervised learning, using a set of labeled training data to solve challenging, non-linear problems. This is accomplished by using the back-propagation algorithm. However, they can be used for unsupervised learning as well,

by training them with a data set that has the same inputs and outputs. Note that an MLP with at least three layers is needed, because the hidden layers determine the outputs. Training an MLP in this manner is affiliated to high computational cost and high training times [LH00].

A more formal definition of MNNs, in accordance to [Ben09], is given by the following equations. Each layer l of an MNN computes a respective output vector a^l using the output of its predecessor a^{l-1}. This sequence starts with the input vector $x = a^0$. The outputs for each layer are generated using the equation

$$a^l = f(b^l + w^l a^{l-1}) \tag{2.34}$$

with an offset vector b^l, a weight matrix w^l and a suitable activation function f (e.g., the sigmoid function or tangens hyperbolicus, see Section 2.3.1.4). A prediction is made on the basis of the results of the output layer a^k, for k being the output layer of the network. Afterwards the output of the training set y and the computed output are combined into a loss function $L(a^k, y)$, usually convex in $b^k + w^k a^{k-1}$. Contrary to the other layers, the output layer might not have linearity. An example for non-linearity is the softmax, a generalization of the sigmoid function

$$a_i^k = \frac{e^{b_i^k + w_i^k a^{k-1}}}{\sum_j e^{b_j^k + w_j^k a^{k-1}}} \tag{2.35}$$

with w_i^k as the ith row of w^k, a_i^k positive and $\sum_i a_i^k = 1$. In that case an approximate value of $P(Y = i|x)$ is given by the output a_i^k, meaning that Y is the class related to the input values x. Afterwards minimization of tuples (x, y) using negative conditional log-likelihood $L(a^k, y) = -\log P(Y = y|x) = -\log a_y^k$ is a common approach.

However, the crucial part in configuring an MNN is to find a suitable number of hidden layers and hidden neurons within these layers. In [Hea08], it is stated, that "there is currently no theoretical reason to use neural networks with any more than two hidden layers". For determining the number of hidden neurons, several guidelines and best practices exist, while in the end, this is still based on trial and error for complex problems where the structure of the analyzed data is not absolutely clear beforehand. The authors of [SD13] have made an investigation on this topic and in [Hea08], some starting points are given:

- The size of the hidden layer should be between the size of the input and output layer.
- The size of the hidden layer should be $\frac{2}{3}$ the size of the input layer plus the size of the output layer.
- The size of the hidden layer should be less then twice the size of the input layer.

Both, too few and too many hidden neurons entail some problems: Too few neurons have meant that the model does not represent the data well leading to an inaccurate model. On the other side, too many neurons result in an over-fitted model and an increased computational effort. [Hea08] Hence, one has to carefully choose the number of hidden layers and hidden neurons.

2.3.2. Unsupervised Learning

An example for unsupervised learning is shown in Figure 2.11 where observations are clustered in two groups of observations. Having just a small dataset with only two dimensions, i.e., features, it is easy to find meaningful clusters. In the example it is quite obvious to split the dataset into two clusters containing $\{A, B, C, D, E\}$ and $\{F, G, H, I\}$ respectively. However, with each additional feature and a larger dataset, the task of partitioning becomes increasingly more complex and nearly impossible to be done manually demanding for a method to understand and evaluate data automatically.

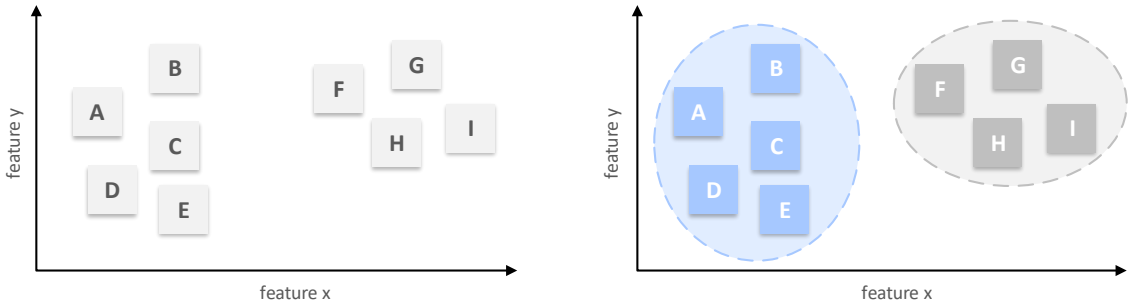


Figure 2.11.: Clustering of a dataset (left) into two clusters (right)

In contrast to supervised learning, unsupervised learning does not need labeled data and commonly is not applied to predict continuous values. The datasets do not contain Y_is to compare the $\hat{Y}_i$s to (hence unsupervised), meaning an observation only consists of n features $X = x_1, x_2, ..., x_n$, without any label y. The goal of clustering algorithms is to examine the similarity of observations based on their features, and then find groups of observations which share a high similarity within the group compared to observations from outside said group [Bis06]. Therefore, a distance function needs to be defined in order to identify similarities of observations based on their distance. Any results can not be validated through the error metrics of Section 2.3.1 since there is no y_i for each observation. To assess the quality of a model, usually, the user needs to apply his domain knowledge in order to verify the utility of discovered structures. [Bis06]

The clustering domain is separated into three types of algorithms [KR08]:

Centroid-based Clustering defines clusters through centroids which are virtual data points representing the proto-element of each group. The number of

clusters has to be defined beforehand and is fixed. In the beginning, cluster centroids are defined randomly and will be shifted in the state space iteratively until the specified distance function is minimized.

Hierarchical Clustering is a group of algorithms where the number of clusters iteratively increases or decreases. That means, the user does not need to predefine the number of clusters beforehand. Hierarchical clustering starts with all observations in one cluster and stepwise divides the cluster with the biggest distance between its data points. This process can be interrupted at any time when a sufficient number of clusters is attained.

Density-based Clustering tries to find areas in the state space with a high density of observations and areas with a low observation density. If a fixed number of data points is within a specified distance, a data point becomes the center of a cluster and the distance criterion is assigned to this cluster. Data not being centers but belonging to a cluster becomes border data. That way data is categorized into cluster, border and noise data.

K-Means Clustering

Although there are many approaches to unsupervised learning, for this thesis only K-means clustering, a centroid-based clustering approach, is relevant. K-means depicted in Figure 2.12 is one of the simplest and fastest algorithms for creating a fixed number of clusters K [Jam+13].

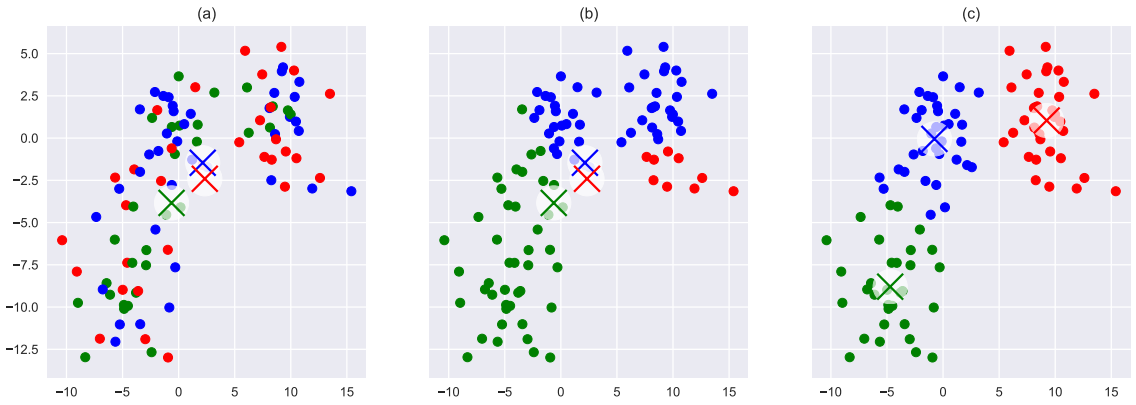


Figure 2.12.: Process of K-means clustering (cf. Algorithm 1): Panel (a) depicts the cluster centroids after the initial random classifications of each observation. Panel (b) shows the first reassignment of the observations to their nearest centroids. In Panel (c) the state after seven iterations can be seen.

It is important to note that the user needs to decide manually on the K when running the algorithm, so even more domain knowledge is necessary for a successful application of this algorithm. Furthermore, the K-means algorithm has some other drawbacks, such as that the algorithm is not deterministic and that

results can vary based on the initial random classification of the observations. Additionally, certain data distributions can lead the algorithm to the production of counter-intuitive results [SCL07]. Nevertheless, in this thesis the data appears to not be in this specific shape and it serves only as a preliminary analysis. For this reason the application of more sophisticated algorithms, as shown in [AV07], is left as future work and will be part of Section 9.2.

The calculation of optimal clusters can be done in many ways and depends on a lot of parameters. The following description of an algorithm is widely used and based on the algorithm described in [Jam+13]. Algorithm 1 defines how the K-means algorithm works in principle. As already mentioned, the number of clusters has to be manually set by the user. In a second step, each of N observations is randomly assigned to exactly one cluster. Then,

$$C_1 \dot{\cup} C_2 \dot{\cup} ... \dot{\cup} C_K = \{o_1, o_2, ..., o_N\} \tag{2.36}$$

with C_k being a set of observations and o_n being an observation whereby none of the (disjunct) clusters can be empty, hence, $N \geq K$. Each observation o_n thereby is a set of p features, i.e., $o_n = \{o_{n,1}, o_{n,2}, ..., o_{n,p}\}$.

Algorithm 1 K-means clustering

1: Define number of clusters k
2: Randomly assign each observation a class from 1 to k
3: **repeat**
4: *Calculate Cluster Centroids*: For each cluster calculate the mean of each feature from all observations in this cluster
5: *Reassign Samples*: For each observation reassign it to the cluster of the closest centroid
6: **until** No observation gets reassigned to a different cluster

Since the quality of a clustering algorithm is defined by the variance of observations within its clusters, the goal can be defined as an optimization problem

$$\text{minimize}_{C_1,...,C_K} \left\{ \sum_{k=1}^{K} W(C_k) \right\} \tag{2.37}$$

where $W(C_k)$ defines the variance of observations in cluster k. In order to execute step 4 and 5 of Algorithm 1, Equation 2.37 must be solved by first defining $W(C_k)$. The calculation of $W(C_k)$ strongly depends on the distance function that is used as a measure to determine the variance within each cluster. In [Jam+13], the squared Euclidean distance is used to define $W(C_K)$ as

$$W(C_K) = \frac{1}{|C_k|} \sum_{i,i' \in C_k} \sum_{j=1}^{p} (x_{i,j} - x_{i',j})^2 \tag{2.38}$$

with $|C_k|$ being the number of observations in the kth cluster and p being the number of features being observed. Simply spoken, $W(C_K)$ is constructed by building the euclidean distance for a pairwise comparison of each observation with each other observation in the cluster, summing up these distance measures and building the mean of it. Inserting Equation 2.38 into Equation 2.37 leads to

$$\text{minimize}_{C_1,\ldots,C_K}\left\{\sum_{k=1}^{K}\frac{1}{|C_k|}\sum_{i,i'\in C_k}\sum_{j=1}^{p}(s_{i,j}-s_{i',j})^2\right\} \tag{2.39}$$

meaning that the sum of all averaged cluster distances needs to be minimized. This now allows to execute step 4 of Algorithm 1. When reassigning observations to the cluster of the closest centroid, and repeating steps 4 and 5 until the clusters do not change any more, the algorithm will terminate at some point delivering a local optimum.

Part II.

Evolution to Cognitive SON Management

3

Introduction to SON Management

In this chapter, a general overview of SON management is provided. The underlying architecture that all the detailed approaches in Chapter 4 - Chapter 7 have in common, is presented, together with a short description of the individual elements, i.e., models, that serve as input. The core component, the so-called objective manager, is introduced as well as a policy system. The chapter is concluded by a short overview of the further course of this thesis and a related work section.

3.1. Motivation

The operation of a SON system is a challenging task that is getting more and more complex for human operators. First, a SON system consists of an increasing number of SON functions which are either operated in a non-managed manner or where its management follows rather simple rules. Second, the SON functions operate within or across different RATs, and across different layers within these RATs. Third, the SON functions may come from different manufacturers, may be designed based on different assumptions, or may aim at different targets to be achieved.

The role of SON management is therefore to provide common means towards the operator to tailor the SON system, i.e., the multitude of independently operating and acting multi-RAT and multi-layer SON functions according to its needs, and to allow for building confidence into the autonomously operating SON system. Furthermore, the aim of SON management is to significantly simplify the operation of a SON-enabled mobile radio network and thereby contributing to a reduction of operational efforts and expenditures.

Human operators, on the one hand, have to manually translate objectives into SCV sets and, on the other hand, need to dynamically change configurations according to operational context. Due to the required efforts, mobile radio networks are currently often configured in a static and uniform way, i.e., the manufacturer of a SON function provides one default configuration, i.e., SCV set, which is deployed over the whole network. This way, the operator misses out on optimization potential with respect to his objectives.

Referring back to Section 1.2.1, this problem is called the *manual gap* and it is one of the main objectives of this thesis to overcome this gap:

Objective 1 *Automate the process of finding optimal SCV sets and close the manual gap between operator objectives and SCV sets.*

The manual gap has been subdivided into three major problems for which no solutions exist in current systems.

Automation Gap Operator objectives cannot be interpreted directly by the SON functions. To enable the operation of the SON-enabled mobile network through operator objectives, an automatic transformation of operator objectives to SCV sets is necessary.

Dynamics Gap Since a mobile network is subject to frequent changes, a concept is required to dynamically adapt the SON functions' SCPs.

Knowledge Gap The configuration of a mobile network is only based on human experience. In order to allow an automated configuration process, knowledge on operator and manufacturer side needs to be described in a way such that it can be automatically processed.

Closing the manual gap between operator objectives on the one side, and SCV sets on the other side, is an important issue for enabling operator objective-driven SON and network operation. It is necessary to link the sets with the operator objectives in such a way that the SON functions fulfill these operator objectives, and changes in the objectives quickly influence the network behavior.

In current systems there is neither an entity available that can manage this link between SCV sets and operator objectives, nor are methods available to perform this mapping in an automated way. In addition, models providing the necessary information about SON functions and operator objectives, and that can be further processed without manual interference, are currently missing. To some extent this also applies to the interface towards SON functions, through which the results of the mapping, the SCV sets, can be provided. Existing configuration management barely provides sufficient capabilities, and existing 3GPP standards for policy provisioning ([3GP12], [3GP13]) only allow a very reduced number of SON function-specific policies.

3.2. SON Management Architecture

The goal of SON management is to overcome this manual gap with automation. Hence, SON management needs to be designed in a way that it enables the automated configuration and dynamic reconfiguration of SON functions based on models which provide all the necessary information in a way that allows an automatic processing.

An overview of the general architecture that is the basis for all approaches that are presented in the course of this thesis, is illustrated in Figure 3.1. The whole SON management system can thereby be separated into three domains, i.e., areas with specific knowledge relevant for the configuration of the SON-enabled network.

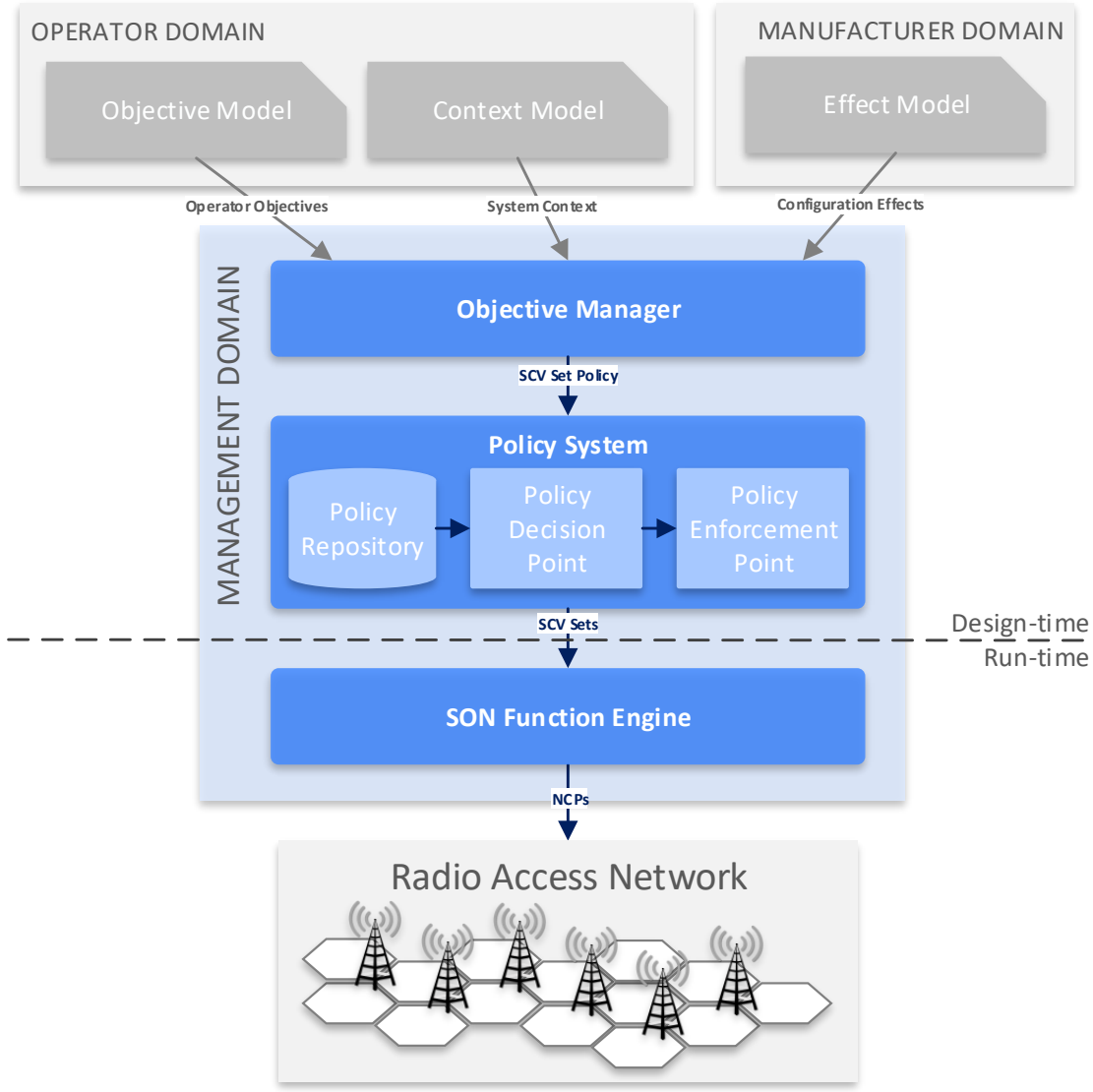


Figure 3.1.: General architecture of SON management with a design-time objective manager

Operator Domain This domain covers models describing knowledge that needs to be provided by MNOs, i.e., an objective model and a context model. The objective characterizes what an MNO wants to achieve in the network. The context model specifies properties of the network.

Manufacturer Domain This domain covers the effect model containing knowledge about SON functions that a SON function manufacturer makes avail-

able. The effect model thereby describes effects of certain SCV sets in the network.

Management Domain This domain covers the actual SON management. An objective manager forms the core component that gets knowledge from the operator and manufacturer domain as input. This knowledge is provided via formalized and automatically processable models. The second component is a policy system, which gets an SCV set policy as input provided by the objective manager. This policy system is split into three elements: A policy repository, a Policy Decision Point (PDP) and a Policy Enforcement Point (PEP). The third and last component is the SON function engine, responsible for deploying SCV sets in the RAN by adapting NCPs.

Input models to the SON management and the SON management elements are described in further detail in the following.

3.2.1. Context Model

Since it is not desired to configure the SON-enabled mobile network uniformly, MNOs distinguish between different contexts for which different goals should be achieved. To be able to differentiate between a variety of contexts, it is necessary to provide information about the current mobile radio network system deployment to the SON management system. The context model provides such a description of network and cell properties (called context properties in the further) that can be used to make the objective and effect model context-dependent. More precisely, it defines the domain, i.e., possible values, of the context properties that can be used in the predicates of the condition parts of these models. As such the context model can be seen as part of the objective model and effect model. Context model information is typically stored in network-wide configuration management and network planning systems and databases. The context model has the following general form:

```
1 {
    contextProperty_1 HAS DOMAIN [propertyValuesMargin_1],
3   contextProperty_2 HAS DOMAIN [propertyValuesMargin_2],
    ...,
5   contextProperty_n HAS DOMAIN [propertyValuesMargin_n]
  }
```

Listing 3.1: General form of a context model

Some of the information contained in a context model is listed below:

- Available RATs: EDGE, UMTS, LTE, ...
- Available layers for each RAT: macro cell, micro cell, pico cell, femto cell, ...
- Available frequency band for each RAT: 2600 MHz for LTE macro cells, ...

- Available location types: urban, suburban, rural, ...
- Most existent user mobility types within a cell's coverage area: static, pedestrian, car, highway, ...

3.2.2. Objective Model

In addition to the context model, the objective manager requires a formalized and processable model of the technical objectives containing context-dependent KPI targets and their precedences. *Context-dependent* thereby means that objectives can depend on operational context, e.g., it can vary for different times of the day, cell types and locations, or traffic patterns. Context within an objective thereby correlates with the context properties and respective values identified in the context model.

The objective model needs to be provided by the network operator. Besides enabling automation, the creation of this formal model also supports operators in becoming aware of their technical objectives in the first place. The model is implemented as a set of rules, since this is a simple and well-known approach which can be easily understood [Str03]. Thereby, each of these objective rules determines the precedence of a KPI target in a specific context. Hence, objective rules have the following general form:

```
IF context THEN KPI target WITH precedence
```

Listing 3.2: General form of an objective rule

Thus, they consist of three parts:

IF The condition part is a logical formula over predicates, which evaluates context properties and thereby determines the applicability of the objective rule in a specific context. This allows specifying under which condition, e.g., time periods or cell locations, a KPI target is active and which precedence it has. Note that the condition can be empty, indicated by the logical formula true, which leads to a general objective rule that is always applicable.

THEN The KPI target defines the KPI with the corresponding target value that the system should optimize.

WITH The precedence represents the importance of the technical objective to the MNO and, thus, allows to trade-off objectives against each other in case they cannot be satisfied simultaneously.

Within SON management, technical objectives are at a low level of abstraction, i.e., close to the technical details of the system like KPIs. In a realistic scenario, an operator may plan and operate the network in terms of high-level goals which are closer to the business view on the network. Hence, these high-level goals need

to be transformed into low-level technical objectives first. However, the transformation of high-level goals into technical objectives is not part of this thesis and is subject to future work.

3.2.3. Effect Model

The aim of SON management is to create configuration settings for the SON functions in such a way that they contribute to the operator objectives. Therefore, the system needs to be aware of the relationship between different SCV sets and their impact on KPIs. In order to be able to determine an SCV set policy, this relationship is again required in a formalized, machine-readable and processable way. Since SON functions are usually delivered as black boxes by manufacturers, i.e., an operator has no or only little information about the SON function algorithm or the corresponding mathematical utility function, the objective manager concept foresees an effect model, allowing manufacturers to provide only information about a SON function being required to implement and utilize it properly. This knowledge can be expressed in simple mappings from SCV sets to KPI effects, and this knowledge is within the domain of the SON function manufacturer. Such a model is required for each SON function (note that in the following, the term *effect model* always denotes the sum of effect models of all particular SON functions).

For each SON function the effect model needs to provide a default mapping defining a configuration if no rule matches the current operational context. This can be, e.g., a balanced configuration of the SON function which trades off different KPI targets.

Summarized, an effect model consists of a set of rules with the following general form:

```
1  IF context AND SCV set THEN KPI effect
```

Listing 3.3: General form of a rule in the effect model

Thus, a rule in the effect model comprises three parts:

IF Equal to the condition part in the objective model, the IF part is a logical formula over predicates evaluating context properties to decide about the applicability of a certain rule. Also, context properties and their values must be contained in the context model.

AND When a certain rule is applicable, the AND part specifies a certain SCV set. Since the main goal of this thesis is to provide an approach to manage a SON-enabled network while abstracting from technical details, SCV sets are always described by transcription such as *MLB_1*.

THEN In case, a condition is fulfilled, this part indicates the effect on KPIs under a specific SCV set (the AND part). A SON function can thereby contribute to a single KPI only, or a set of or even all network KPIs.

Note that the transcriptions used for the SCV sets always refer to a concrete set of SCPs and respective values for these parameters. For instance, an MLB function is allowed to modify the CIO, the upper CL threshold from which MLB becomes active, the lower CL threshold from which MLB returns to inactive state, and the load averaging time based on which the current CL is calculated. Exemplary values for an MLB SCV set are:

- Upper CIO limit: +6dB
- Lower CIO limit: -6dB
- Stepsize: 1dB
- Upper CL threshold: 50%
- Lower CL threshold: 30%
- Load averaging time: 60 seconds

For the presentation of the objective manager in this thesis, it is assumed that the KPIs used in the objective model and the effect model match each other, i.e., they have the same name and meaning. This simplifies the explanation but might be too inflexible in practice. However, this assumption is not a limitation of the general approach since a translation model can provide a mapping between the KPI definitions of both models.

3.2.4. Objective Manager

By means of performing a reasoning process, the objective manager determines the SCV set policy, or directly the appropriate SCV sets, according to the technical objectives defined by the MNO, the current network operational context and past experiences. To decide about the best SCV sets, some additional input to the objective manager is needed, namely, the above mentioned objective model, context model and effect model.

The objective manager can be operated in two different modes: a design-time option and a runtime option. Depending on the option, the reasoning process within the objective manager and the corresponding output to the underlying building blocks are different.

In the design-time option the objective manager creates an SCV set policy before the instantiation of the SON functions. This policy is thereby defined as a set of ECA rules. Each of these rules consists of an event on which it is triggered, one or more conditions which are to be checked, and an action that is taken when the conditions are met. For the SCV set policy within SON management, the events

and conditions are derived from the technical operator objectives and their associated target values respectively. The action part of an ECA rule states instructions. In the case of SON management, these instructions are the configuration values to be deployed to the SON functions or their instantiations, respectively. Note that there might be rules with an empty condition part, i.e., the action part of the rule is always executed if the triggering event has taken place. Furthermore, note that a policy does not need to explicitly state its purpose or goal.

The SCV set policy hence contains all information required to control the SON system, i.e., the multitude of SON function instances, according to the technical objectives of the network operator, i.e., KPI targets, KPI precedences, conditions like time, location or network status under which KPI targets and precedences apply, and additional restrictions that may apply. Thus, the objective manager determines a complete set of rules describing in detail what is to be done under every possible condition. The rules thereby have to be in line with the technical objectives, but whether or not this is the case cannot generally be determined from considering this rule in isolation. Therefore, the SCV set policy including the complete set of SON-related ECA rules needs to be consistent in such a way that all technical objectives are addressed. This implies that the resulting policy has to be defined conflict-free, i.e., possibly conflicting operator objectives need to be resolved when creating the SCV set policy.

Within the design-time option, the policy has to be recomputed in case KPI targets or their precedences change, if the definitions, properties or allocation of conditions and restrictions change, or if the predicted effects of SCV sets change.

In the runtime option (cf. Figure 3.2) the objective manager performs not only the reasoning and mapping processes between the objective and effect model, but also acts as decision point with respect to selecting appropriate SCV sets to be deployed. With the runtime option, the output of the objective manager does not consist of an SCV set policy, but directly of SCV sets that are deployed to, and enforced at the SON functions. The objective manager does not always need to perform a full mapping between all technical objectives and all effect models, but only SON function instances affected by the changes have to be updated and deployed.

Both options have their own advantages and drawbacks. On the one hand, the runtime option is computationally less complex. However, the functioning of the objective manager gets less visible to the MNO due to the lack of an SCV set policy. Since it is one of the main objectives of this thesis to develop a SON management system that leaves the highest possible degree of control to the MNO (cf. Section 1.2.4), the design-time option is chosen for all following approaches and hence, a policy system is needed enforcing the output of the objective manager in the network.

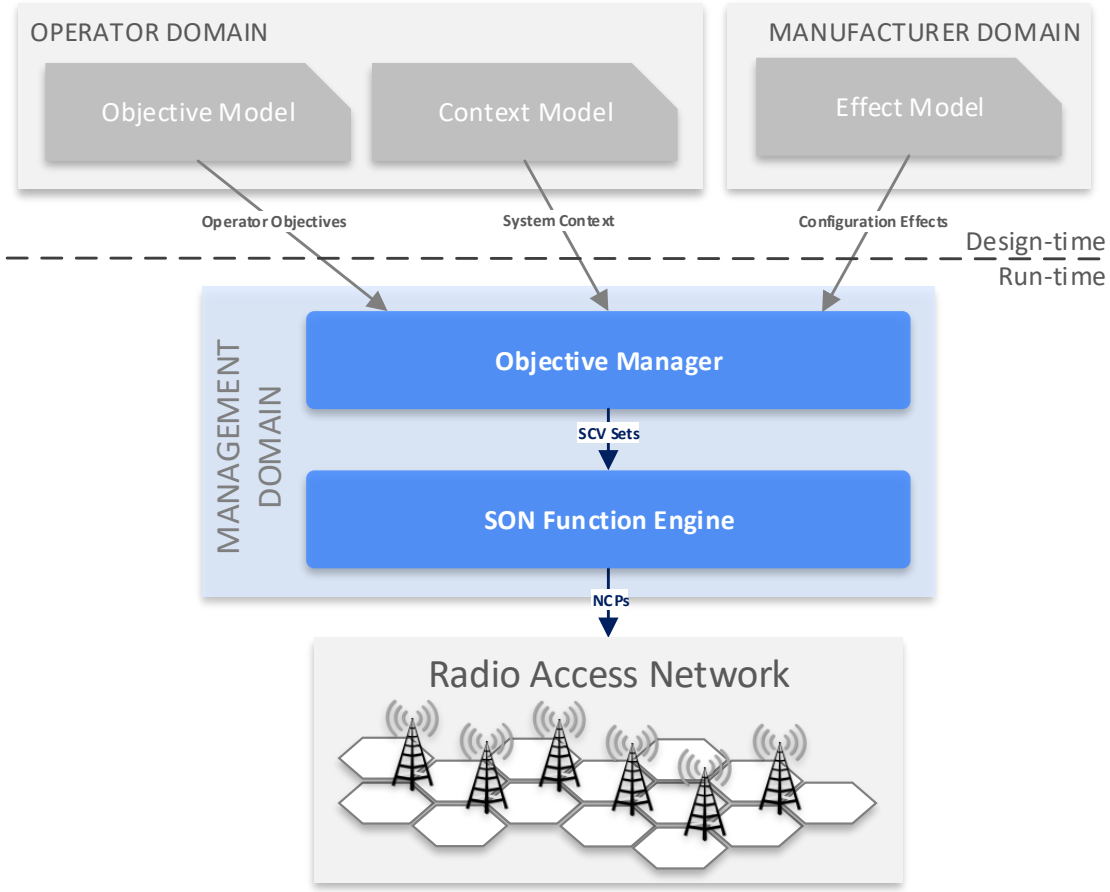


Figure 3.2.: General architecture of SON management with a runtime objective manager

3.2.5. Policy System

A policy system is an established approach for network management [Str+06]. Several implementations are available, e.g., JBoss Drools [DRO18]. The policy system as it is used for SON management, evaluates the SCV set policy generated by the objective manager at runtime and dynamically configures the SON functions. Basically, an SCV set policy is a set of condition-action policy rules with the following general form:

```
1  IF context THEN SCV set
```

Listing 3.4: General form of an SCV set policy rule

Thus, a policy rule consists of two parts:

IF Like the conditions in the objective model and effect model, the IF part specifies a logical formula that can be evaluated to decide whether a rule is applicable or not. The context properties and values used in the conditions are again in accordance with those defined in the context model. Note that the

sum of all policy rules does not necessarily cover all possible context property combinations and hence, it may be the case that in a certain operational context, no rule is applicable. In such a case, default SCV sets are used.

THEN The action part reflects the objective manager's decision with respect to optimal SCV sets under a specific operational context. That is, it provides the SCV sets that contribute optimally to the fulfillment of technical operator objectives.

In order to interpret the set of rules, there are commonly three components within the policy system: A policy repository, a PDP and a PEP [Wes+01].

Policy Repository The policy repository stores the SCV set policy, i.e., the entirety of all policy rules which have been generated by the objective manager.

Policy Decision Point (PDP) The decision, which policy rules must be applied, is taken by the PDP component. Therefore, it listens for trigger events and, once such an event occurs, the current context is needed which can be directly derived from the network. Using this context, the PDP evaluates the conditions of the rules in the SCV set policy, i.e., the IF parts, and gathers the applicable SCV sets for the SON functions. Since the SCV set policy is conflict-free, there is always exactly one SCV set selected for each SON function.

Policy Enforcement Point (PEP) The PEP component is responsible for the execution of the THEN part of the policy rules selected by the PDP. That is, the PEP configures the SON functions with the respective SCVs. For each SCV set, the PEP determines whether the respective SON function is already configured accordingly or otherwise, deploys a new SCV set to the SON function.

A policy system requires an external component that triggers the execution of a policy. This usually is a monitoring component that triggers the policy system at fixed time intervals such as every hour or in case the daytime changes from busy hours to low traffic hours. Also, more sophisticated trigger events are thinkable, e.g., when an undesired behavior is detected. However, this is a non-trivial task and is out of the scope of this thesis. An approach of a more complex monitoring and diagnosis component has been developed during the SEMAFOUR project and is provided in [Göt+15].

3.2.6. SON Function Engine

From the policy system or more precisely, the PEP, SON management sends the SCVs to the SON function instances. Each SON function has a number of SCPs and dedicated parameter values, i.e., SCVs, that can be assigned to the SON function instances. Examples for parameters are, e.g., activation thresholds, stepsize

or limits for the output NCPs. Different values for the parameters can influence the behavior of the SON function instance in such a way that its algorithmic behavior changes, thereby its influence on the configuration of the cell, network element or the complete network changes, and finally the behavior of the network changes. A dedicated set of parameter values for a SON function instance can influence the behavior of the network in such a way that it works towards dedicated KPI targets, and hence, operator objectives. Thus, the appropriate configuration (or instrumentation) of the SON function instances being operational in the network leads to the required network behavior according to the operator objectives. Standard formats and interfaces are required to provide the necessary information to SON functions from different manufacturers.

3.3. Evolution to Cognitive SON Management

As already mentioned, in this chapter a general SON management architecture is presented without going too much into the details of the input models and the functioning of the reasoning process of finding optimal SCV sets. Instead, it is the primary aim of this chapter to show the points of intersection between the different management approaches that are presented in the following chapters.

What all these approaches have in common is the type and structure of their input models, the objective manager as core component mapping data provided by MNOs and SON function manufacturers, and a policy system to enforce the objective manager's decisions in the network. However, input models in the PBSM, ODSM, ASM and Cognitive SON Management (CSM) approach strongly differ in the data they provide and their derivation. That is, first of all, the successively increasing amount of unstructured, heterogeneous and partially sketchy data, but also the complexity of converting them into an automatically processable, standardized format that can be easily understood by the objective manager. Also, the reasoning process performed by the objective manager gets more complicated since input models consist of more and more sub-models with each further SON management development stage. Even though the output of the objective manager is always an SCV set policy with the same format over all approaches, the policy rules have to be interpreted by the policy system in a different way.

3.3.1. Policy-based SON Management

Within PBSM, input models are described in a formalized way for the first time. Besides this, the primary aim of this approach is to overcome the manual gap as presented in Section 1.2.1. Hence, input models are not that sophisticated but provide a first draft on how to describe operator and SON function manufacturer

Table 3.1.: Variations in input models, the objective manager and policy system from PBSM to ODSM to ASM to CSM

	PBSM	**ODSM**	**ASM**	**CSM**
Context Model	context properties with value ranges	context properties with value ranges	classification of context properties into context classes by inspection	two sub-models: a model based on cell parameter classification and one based on KPI value classification
Objective Model	prioritized objectives for arbitrary combinations of context properties, KPI target is a minimum or maximum indication	weighted objectives for arbitrary combinations of context properties, KPI target is a concrete value to be achieved	weighted objectives for manually defined context classes, KPI target is a concrete value to be achieved	weighted objectives for manually defined context is a concrete value to be achieved
Effect Model	minimum or maximum KPI effect indications for each single SON function derived from simulations by the manufacturer	concrete KPI effect indications for SCV set combinations based on manufacturer effect model	two sub-models: a combined effect model and an effect model derived from real network measurements during operation, both dependent on context classes	three sub-models: a combined effect model, a real network effect model and a learned effect model dependent on automatically derived cell clusters
Objective Manager	optimal SCV sets for the highest prioritized objective, calculation done for each SON function individually	optimal combination of SCV sets with highest overall utility	optimal combination of SCV sets with highest overall utility, taking real network measurements into account	optimal combination of SCV sets with highest overall utility, considering real network measurements and untested SCV sets
Policy System	complete and conflict-free set of rules for all combinations of context properties	complete and conflict-free set of rules for all combinations of context properties	conflict-free set of rules, one rule for each context class	conflict-free set of rules, one rule for each cell

knowledge. The same applies for the objective manager. An SCV set policy is determined that calculates the best possible configuration for each SON function individually, hence not considering KPI effects over two or more different SON functions. Due to a relatively complex context model, the computational time of the output SCV set policy calculation is rather high. Although this approach has several drawbacks, it proposes a first promising method on the way to a fully developed SON management.

3.3.2. Objective-driven SON Management

ODSM aims at overcoming the main disadvantages of PBSM which foremost means the improvement of input models as described in Objective 2 (cf. Section 1.2.2). Objectives can be weighted and hence, they can be compromised against each other. Furthermore, objectives now aim at the achievement of concrete KPI values and so do SCV sets in the effect model. One of the biggest advantages compared to PBSM is the functioning of the objective manager. Instead of determining optimal SCV sets for each SON function individually, cross effects between them are considered resulting in a combined SCV set. Also, the output SCV set policy gets less complex and therefore, easier to understand for an MNO which is a first step to gain trust for an MNO according to Objective 4 (cf. Section 1.2.4).

3.3.3. Adaptive SON Management

ASM does not provide novelties with respect to the context and objective model, but enhances the initial effect model coming from simulations done by the SON function manufacturer by a second sub-model. This sub-model is generated by collecting, filtering and processing real network measurement data and hence, making the initial effect model more realistic. Furthermore, the problem of the relatively complex context model is concerned by partitioning the huge context space into easily manageable context classes. Having such a context classes model and an advanced effect model, the derivation of optimal SCV sets completely changes. The objective manager needs to handle both types of effect models which are generated in different environments. This leads to a more complex but also much more realistic methodology while on the other hand, computational effort is reduced by having a simplified context model. Calculations need to include data coming from different sources and combine them into a context classes-based SCV set policy covering both environments' effect models. Summarized, this approach mostly addresses the objective of making the effect model more realistic, see Objective 2 in Section 1.2.2, and, as a consequence, raises trust for an MNO, see Objective 4 in Section 1.2.4.

3.3.4. Cognitive SON Management

In the final development stage, the CSM, objectives described in Section 1.2.2, Section 1.2.4 and especially Section 1.2.3 are addressed. While the objective model stays equal compared to the two previous approaches, the context model and effect model are each enhanced by one sub-model. Both sub-models use machine learning techniques to approximate themselves closer to reality. Cells are classified into cell clusters based on current KPI measurements for a more precise SCV set selection. A learned effect sub-model predicts the effects of up-to-now untested SCV sets and thereby enables the usage of a wider variety of possible SCV sets which may lead to a better network performance in terms of KPIs. The existence of two more sub-models also leads to an increasingly more complex reasoning process of the objective manager. Objective definitions are dependent on context defined by the operator while effects in the effect model depend on the KPI-based context model. For the elicitation of SCV sets, all effect models need to be considered at certain points in time. Thereby, having the trust objective in mind does not facilitate the development of the objective manager in the CSM approach. The whole process which is getting more and more complex still needs to be understandable and visible for MNOs. Finally, the output SCV set policy differs from previous approaches in the fact that it suggests SCV sets for cells instead of context classes.

3.4. Related Work

Currently, an approach that starts at objectives and uses them to adapt SON functions, i.e., a method to overcome the gap between objectives and SCV sets, is missing, the reason why SON functions are executed with default parameter values. Thus, also approaches are considered which do not start at objectives, but only at a lower level of abstraction. The possibility to enable different constituencies to describe policies at different layers of abstraction, is presented by [Str+06] in a prototype implementation of the policy continuum. The necessity of having different abstraction views and the architecture are described in principle, but an automated approach to transform business policies, i.e., operator objectives, into low-level policies, i.e., SCV sets, is missing.

An automated approach that deals with the automated transformation of (operator) business policies into technical policies that can be executed by SON functions, is presented in [RBS11]. Even though this is a promising approach, it is not objective-driven. The main problem that is solved in this thesis, to overcome the manual gap between operator objectives and SCV sets, is not part of that approach since both high-level abstract configuration policies and low-level concrete parameter values are based on the same paradigm, the ECA-paradigm, making transformations considerably easier.

In [Gal+12], a policy-based framework is presented that combines the approaches published in [RBS11] and [Str+06]. This framework defines three layers on which policies are described at different degrees of abstraction and a refinement process that transforms higher-layer policies into policies of the subsequent layer. However, their transformation starts with business policies and does not consider operator objectives as described in Section 3.2.2.

The approach that is closest to the general SON management approach in this chapter is the policy refinement approach described in [Ban+04]. The authors present an approach that transforms goals into low-level policies, i.e., actions to take in response to some event, in a two-stage process. Thereby, goals are a high-level description of the expected system state after some event occurrence. First, a high-level goal is manually elaborated into more detailed sub-goals. Second, a sequence of concrete actions which achieve the goals are inferred through a process called abduction. Disadvantages of this approach are, that a detailed semantic description of the actions in form of pre- and postconditions is needed in order to be able to execute transformations in an automated way and, that operations are constant for given goals, i.e., it is not possible to define precedences for objectives.

A similar approach is presented in [SK05]. In contrast to the previous concept, it defines the semantics of the actions in form of forecast functions which estimate the system state after the execution of some action. Thereby, these functions can be learned. The disadvantage of this approach is the need for a formal, detailed action model which requires SON function manufacturers to reveal the details of their SON functions.

In [Kür+10], the authors present an idea which can potentially fill the manual gap in SON management and operation. This idea describes the refinement of operator policies into SON function-specific policies in order to configure the SON functions in a way that their behavior is aligned towards a common goal. However, it has never been described how this could be accomplished.

4

Policy-based SON Management

In this chapter, a first approach is presented that allows to overcome the manual gap described in Section 1.2.1. This approach is called PBSM and its architecture is strongly related to the architecture presented in the previous chapter. However, while in Chapter 3 input models and other SON management elements are described on a high level of abstraction, this chapter presents a first possible implementation of SON management in-depth. Besides a theoretical description of input models and a concrete specification of the functioning of the objective manager, an example is provided to make the process of finding optimal SCV sets more clear.

4.1. Motivation

In today's mobile radio networks the technical objectives are usually well-defined by MNOs, for example, using templates or handbooks describing how to manually configure the network in order to achieve the given objective. The transformation from technical objectives into SCV sets is mainly performed manually and strongly depends on the knowledge and experience of the human operators planning and managing the network. That's why the focus of PBSM is on identifying existing knowledge in the first place, and to develop a solution that allows an automated instrumentation of the SON functions such that they contribute towards achieving defined KPI targets. The advantage of having a high degree of automation is, that, apart from relieving the human operator from performing this work manually, network context and prioritization between KPI targets can be considered, which is barely possible at the level of individual cells in case of manual operation.

By developing the PBSM approach, mainly two objectives of this thesis are addressed. First, PBSM marks an important step towards an automated network configuration by overcoming the manual gap. A process is defined to bridge the gap between technical operator objectives and the adjustment of NCPs in the RAN. By introducing a policy system, the network can adapt itself at any time without the need of an operator action. Knowledge on operator and manufacturer side is thereby made available to the SON management in the form of models that can be automatically processed.

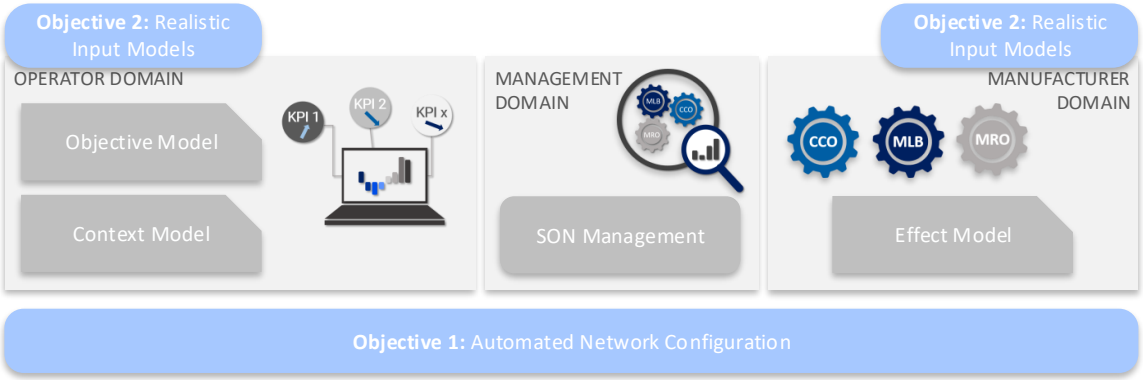


Figure 4.1.: Objectives of PBSM

Figure 4.1 shows the areas where PBSM meets objectives of this thesis. Overcoming the manual gap, i.e., the above characterized automation gap, dynamics gap and knowledge gap, covers all three domains and helps to achieve Objective 1:

Objective 1 *Automate the process of finding optimal SCV sets and close the manual gap between operator objectives and SCV sets.*

Besides Objective 1, a step forward towards achieving Objective 2 is made by defining an objective model, context model and effect model for the first time. Even though they may be not extensive, accurate and realistic enough to be trustworthy for an MNO, they provide the basis for further extensions of the input models.

Objective 2 *Automatically generate realistic and complete input models which are continuously updated according to the current network state such that they optimally support the SON management system in finding the best possible network configuration.*

4.2. Approach

The architecture of PBSM is shown in Figure 4.2 and is thereby consistent with the one presented in Figure 3.1 while presenting more details. The objective manager acts at design-time and hence, a policy system is needed to deploy SCV sets by means of a SON function engine in the network. Three types of models are used to perform the mapping process within the objective manager, namely, the objective model, context model and the effect model. The effect model is thereby provided as a set of sub-models each coming from the respective manufacturer of a SON function.

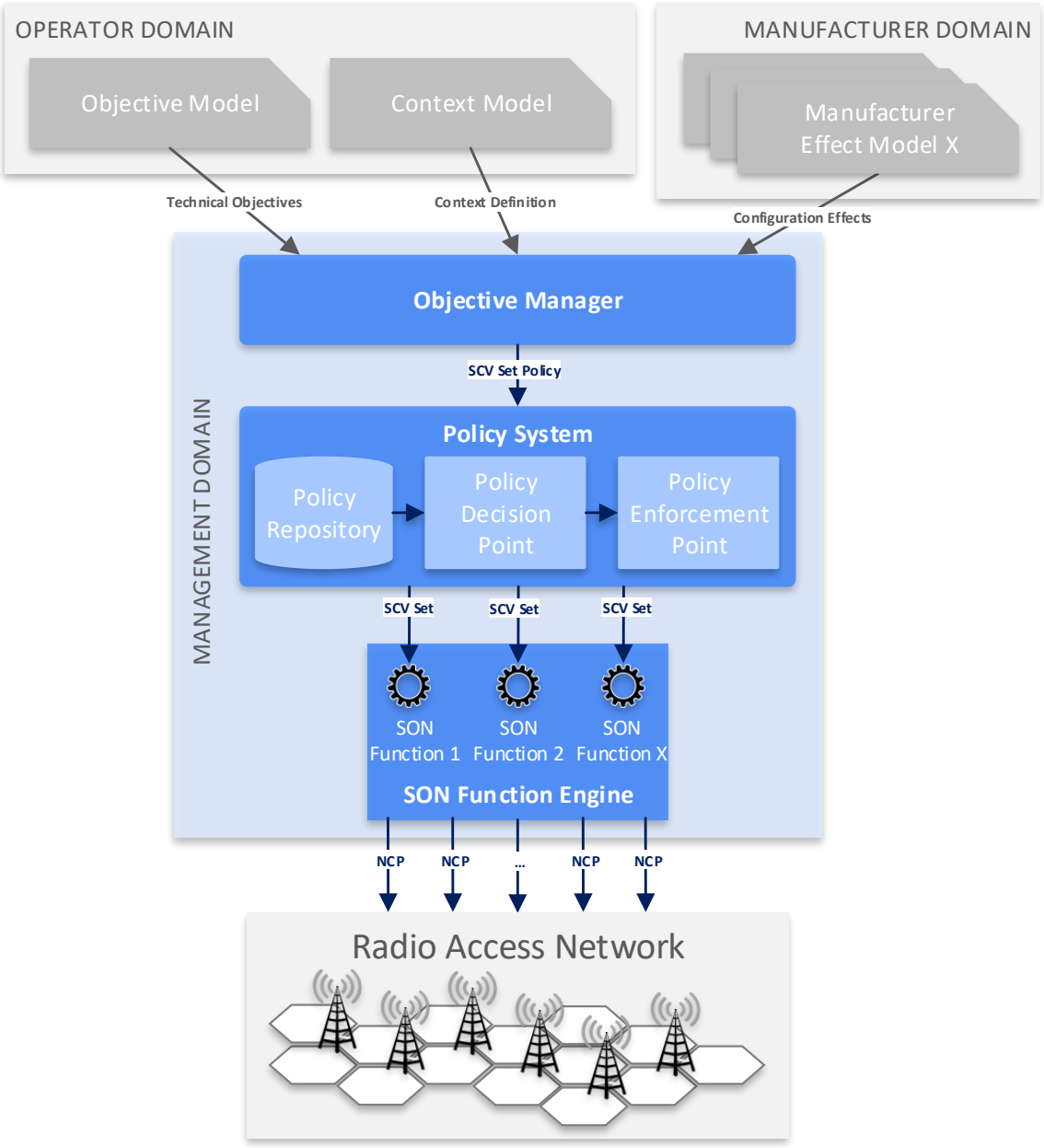


Figure 4.2.: Overview of PBSM

For further formalizations, the SON-enabled mobile network needs to be formalized as well. A is the set of all context attributes $a \in A$, i.e., properties that describe the network and the cells operating in the network.

$$A = \{a_1, a_2, ..., a_{|A|}\} \tag{4.1}$$

F is the set of all SON functions, e.g., MLB, MRO, CCO.

$$F = \{f_1, f_2, ..., f_{|F|}\} \tag{4.2}$$

For each SON function $f \in F$, a set S^f is given containing different SCV sets. An SCV set s^f is a set of values $v_1^{s^f}, v_2^{s^f}, ...$ for input parameters, i.e., SCPs, of a function f.

$$S^f = \left\{ s_1^f, s_2^f, ..., s_{|S^f|}^f \right\} \tag{4.3}$$

The performance of a mobile network is monitored through a set of system KPIs K, e.g., DCR, HOSR and CL. Thereby, it is assumed that each KPI $k \in K$ may have a different value range $\mathrm{Dom}(k) \subseteq \mathbb{R}$, e.g., Dom(DCR) = [0,1].

$$K = \{k_1, k_2, ..., k_{|K|}\} \tag{4.4}$$

In order to simplify the illustration of the PBSM concept, it is assumed that all SON functions have a cell scope, i.e., for each single cell in the network, there is an instance of each SON function adapting the NCPs of solely that cell. Accordingly, technical objectives in the objective model are evaluated in the context of single cells. This assumption allows to configure SON functions for each single cell according to the objectives applicable in the context of that cell.

In the following, the details of mentioned models and the process of bringing them together are explained. The explanation starts with the context model since its content is relevant within the objective model and the effect model.

4.2.1. Context Model

In nowadays heterogeneous network topologies, different cells play different roles in achieving the targeted network performance depending on various parameters such as the cell's type, used technologies, type of the cell's environment, or density of the surrounding network topology to name a few. These different roles may be expressed, for example, in terms of individual performance targets per cell and individual parameterizations of the employed SON functions. With thousands of cells in realistic networks, it becomes increasingly difficult to manually decide upon these individual settings. For this reason, the characterization of cells is abstracted by introducing what is called context properties describing the nature of the cell and the context it is working in.

For instance, such context properties can be:

- The time of the day, since KPI targets and their importance may be different during peak traffic hours and periods with low traffic, e.g., the time period from 08:00 till 17:59, or the time periods from 18:00 till 23:59 and from 00:00 till 07:59.
- The location of the cell, since the KPI targets and their importance may be different in, e.g., urban, suburban, and rural areas, due to user behavior, number of users, or coverage and capacity requirements.

- The cell type, e.g., macro cell, micro outdoor cell, or indoor cell, since the KPI targets and their importance may be different with respect to coverage and capacity requirements, user behavior, or the availability of cells.
- The status of the system based on performance or fault data, e.g., KPI values or alarms.

Following the general form of a context model as presented in Listing 3.1, a context model in PBSM looks as follows:

```
1 {
    time HAS DOMAIN [00:00, 23:59],
3   location HAS DOMAIN {urban, rural}
  }
```

Listing 4.1: Exemplary context model in PBSM

So, the context model defines the value ranges, i.e., domains, of all attributes $a \in A$. When defining domains in a formalized way, one has to distinguish between discrete and continuous context attribute domains. While discrete attributes a^d such as location and cell_type have a domain consisting of a finite set of values v_i with n possible values, i.e.,

$$\text{Dom}(a^d) = \{v_1, v_2, ..., v_n\} \tag{4.5}$$

continuous attributes a^c have a domain with a certain value range reaching from a minimum value v_{min} to a maximum value v_{max}, i.e.:

$$\text{Dom}(a^c) = [v_{\text{min}}, v_{\text{max}}] \tag{4.6}$$

Summarized, the context model is a set of attributes $a \in A$ containing context properties with either a discrete or a continuous domain.

4.2.2. Objective Model

The primary aim of mobile radio network operations is not the optimization of dedicated single performance indicators at the cell or base station level, but the achievement of dedicated KPI targets. Different KPI targets may be competing with each other, i.e., they are not achievable together. An MNO needs to define the importance of KPI targets in order to trade them off against each other. This importance can be expressed through allocating priorities to the individual KPI targets. Note that a priority here means a ranking of KPI targets and not a weighting. KPI targets and their priorities may change over time due to changing operator requirements. Furthermore, KPI targets and their priorities may depend on operational or network context properties as defined in the context model. That is, there may be different values assigned to the KPI targets, or a different prioritization between the KPI targets, depending on various context property values.

```
IF context properties combination THEN KPI target WITH priority
```

Listing 4.2: Technical objective rule in PBSM

In this thesis, a context-dependent KPI target and its associated priority is also called a technical objective (cf. Listing 4.2).

When combining KPI targets and their priorities with context information, dedicated technical objectives can be derived which build the basis for network operation and, hence, the SON system. Spoken in natural language, such technical objectives are, for example:

- With a very high priority, the cell load in an urban location during peak hours should be minimized.
- With a high priority, the dropped call rate in an urban location should be minimized.
- With a moderately high priority, the handover success rate during peak hours should be maximized.
- With a moderate priority, energy consumption in a rural location should be minimized.
- With a low priority, the cell load during peak hours should be minimized.
- With a very low priority, energy consumption during periods with low traffic should be maximized.

It is obvious that these technical objectives can be implemented as a set of rules matching the form of a technical objective defined in Listing 4.2:

```
1 {
    IF time in [08:00, 17:59] AND location = urban THEN CL_MIN WITH 1
3   IF location=urban THEN DCR_MIN WITH 2
    IF time in [08:00, 17:59] THEN HOSR_MAX WITH 3
5   IF location=rural THEN EC_MIN WITH 4
    IF time in [08:00, 17:59] THEN CL_MIN WITH 5
7   IF time in [00:00, 07:59] OR time in [18:00, 23:59] THEN EC_MIN
      WITH 6
9 }
```

Listing 4.3: Exemplary objective model in PBSM

CL_MIN thereby refers to the minimization of the KPI CL, *DCR_MIN* to the minimization of DCR, *HOSR_MAX* to the maximization of HOSR and *EC_MIN* to the minimization of Energy Consumption (EC). The priority encodes the importance of the KPI target to the operator. The KPI target with the highest importance is indicated with a priority of 1, decreasing importance is indicated with priority 2, 3, 4, etc.

Formally, the target t_k on a KPI is either approximating 0 in case it should be minimized or 1 in case it should be maximized. Both, the KPI target t_k and the priority p_k, make up a (un-conditioned) technical objective $o_k \in O_k$ for a KPI $k \in K$.

$$o_k = (t_k, p_k) \tag{4.7}$$

with $t_k \in \{0, 1\}$ (depending on, if a KPI should be minimized, i.e., $t_k = 0$ or maximized, i.e., $t_k = 1$), $p_k \in \mathbb{N}$ and $p_k \geq 1$. Since a priority must be unique, the following must apply:

$$\forall o_{k_i}, o_{k_j} \in O_k : p_{k_i} \neq p_{k_j} \tag{4.8}$$

For instance, the first objective in Listing 4.3 would be represented as $o_{\text{CL}} = (0, 1)$. Considering the IF part, the objective model OM can be seen as a function that maps a specific operational context $\chi \in X$ to a tuple of objectives for all $|K|$ KPIs, i.e.:

$$\text{OM} : X \mapsto O_{k_1} \times O_{k_2} \cdots \times O_{k_{|K|}} \tag{4.9}$$

Note that some important points apply for this implementation of the objective model. First, the priorities do not need to be unambiguous in some specific context, i.e., it can be the case that one KPI target has two different assigned priorities. This can happen if two objective rules with overlapping conditions and the same KPI target are triggered. An overlap thereby means that at least one specific context exists in which both conditions are true. In such cases, this conflict is resolved by solely considering the higher priority.

Second, it should never be the case that two different KPI targets have an equal priority in a certain situation, since this would mean that it does not make a difference to the MNO which KPI target is pursued. In such a situation, the system cannot make a deterministic decision. Instead, the triggered KPI targets must be in a total, strict order regarding the priorities in every context (cf. Equation 4.8). This requirement makes the development of the objective manager more complex. However, the objective manager algorithm provides support for validation and verification of the objective model.

Third, the objective model does not need to be complete, i.e., not all KPI targets need to be defined in all contexts, i.e., $O_k = \varnothing$ for certain $k \in K$. As presented later, this might result in the selection of a default SCV set for some SON functions.

4.2.3. Effect Model

SON functions are usually delivered by SON function manufacturers as black boxes, meaning the operator has no insight into the actual functioning of the algorithms. Hence, an effect model is responsible for encoding a functional description of a specific SON function. That is, the model describes which KPI targets

the SON function can pursue and how to configure the SON function accordingly. This knowledge can be expressed in simple mappings from KPI effects to SCV sets.

```
{
  MLB Model
    IF CL_MIN THEN MLB_1
    IF EC_MIN THEN MLB_2
    IF HOSR_MAX THEN MLB_3
    IF default THEN MLB_3

  CCO Model
    IF DCR_MIN THEN CCO_1
    IF CL_MIN THEN CCO_2
    IF default THEN CCO_2

  ES Model
    IF EC_MIN THEN ES_1
    IF CL_MIN THEN ES_2
    IF default THEN ES_2

  MRO Model
    IF HOSR_MAX THEN MRO_1
    IF DCR_MIN THEN MRO_1
    IF CL_MIN THEN MRO_2
    IF default THEN MRO_2
}
```

Listing 4.4: Exemplary effect model in PBSM, one function-specific effect model for each SON function

Listing 4.4 illustrates four sub-models of an effect model, one for each of the SON functions MLB, CCO, ES and MRO. As can be seen, a mapping in the effect model links a KPI effect to a single SCV set. Note that the SCV set names, e.g., *MLB_1*, are visual placeholders for concrete parameter values as shown in Section 3.2.3.

Formally, the function-specific model for one dedicated SON function f defines a mapping FM^f between a set of possible SCV sets S^f and their effects on the KPIs E^f, i.e.

$$\mathrm{FM}^f : S^f \mapsto E^f \tag{4.10}$$

$$E^f = \left\{ \varepsilon_1^f, \varepsilon_2^f, ..., \varepsilon_{|E^f|}^f \right\} \tag{4.11}$$

Thus, the entire manufacturer-provided effect model EM is defined as a set of function-specific sub-models FM^f, one for each SON function:

$$\mathrm{EM} = \left\{ \mathrm{FM}^{f_1}, \mathrm{FM}^{f_2}, ..., \mathrm{FM}^{f_{|F|}} \right\} \tag{4.12}$$

It is assumed that KPI effects in the effect model match KPI targets in the objective model, i.e., they must have the same name and meaning. Otherwise, the objective manager can not relate them to each other. Furthermore, an effect model must be unambiguous, i.e., for each KPI effect there can be at most one SCV set defined. Otherwise, the objective manager would not know which SCV set to use. Finally, each sub-model needs to provide a default mapping defining an SCV set if no matching KPI effect is relevant to the operator. This can be, e.g., a balanced configuration of the SON function which trades off different KPI targets.

A possible extension of the effect model is to make SCV sets context-dependent like the objective model. This would allow to express different SCV sets for each SON function, for example, whether the cell on which ES is active overlaps with other cells or not, given a so-called heterogeneous networks scenario [Ame12]. However, this increases modeling complexity because it has to be ensured that the rules of the effect model are conflict-free.

4.2.4. Methodology

Based on the three previously introduced models, the objective manager derives the SCV set policy according to the algorithm depicted in Figure 3. In principle, it determines the best SCV sets with respect to the technical objectives in all possible contexts and subsequently creates policy rules from this information.

4.2.4.1. Generation of Partitioned Context State Space

In the first step, the objective manager builds a space of all possible contexts the system could be in, referred to as state space. Therefore, it analyzes the context model: each context property represents a dimension in the state space and the domain refers to the scale of this dimension. As described in Section 4.2.1, a context parameter can have a discrete or continuous scale. Since the state space can be infinitely large, i.e., it contains an infinite number of states, the system needs to reduce the state space. Therefore, the algorithm divides the state space into a finite number of state space regions with respect to the technical objectives. Specifically, a region is a set of adjacent states which have the same KPI targets t_k and priorities p_k, i.e., in which the SON system should be configured equally. The regions can be computed by analyzing the conditions of the objective rules: for each predicate π, the dimension of the context attribute a in π is partitioned according to the value v in π.

Formally, a predicate is defined as

$$\pi = (a, v) \tag{4.13}$$

with $a \in A$ and $v \in \mathrm{Dom}(a^d)$ if a is a context property with a discrete scale and $v \subseteq \mathrm{Dom}(a^c)$ if a is a context property with a continuous scale.

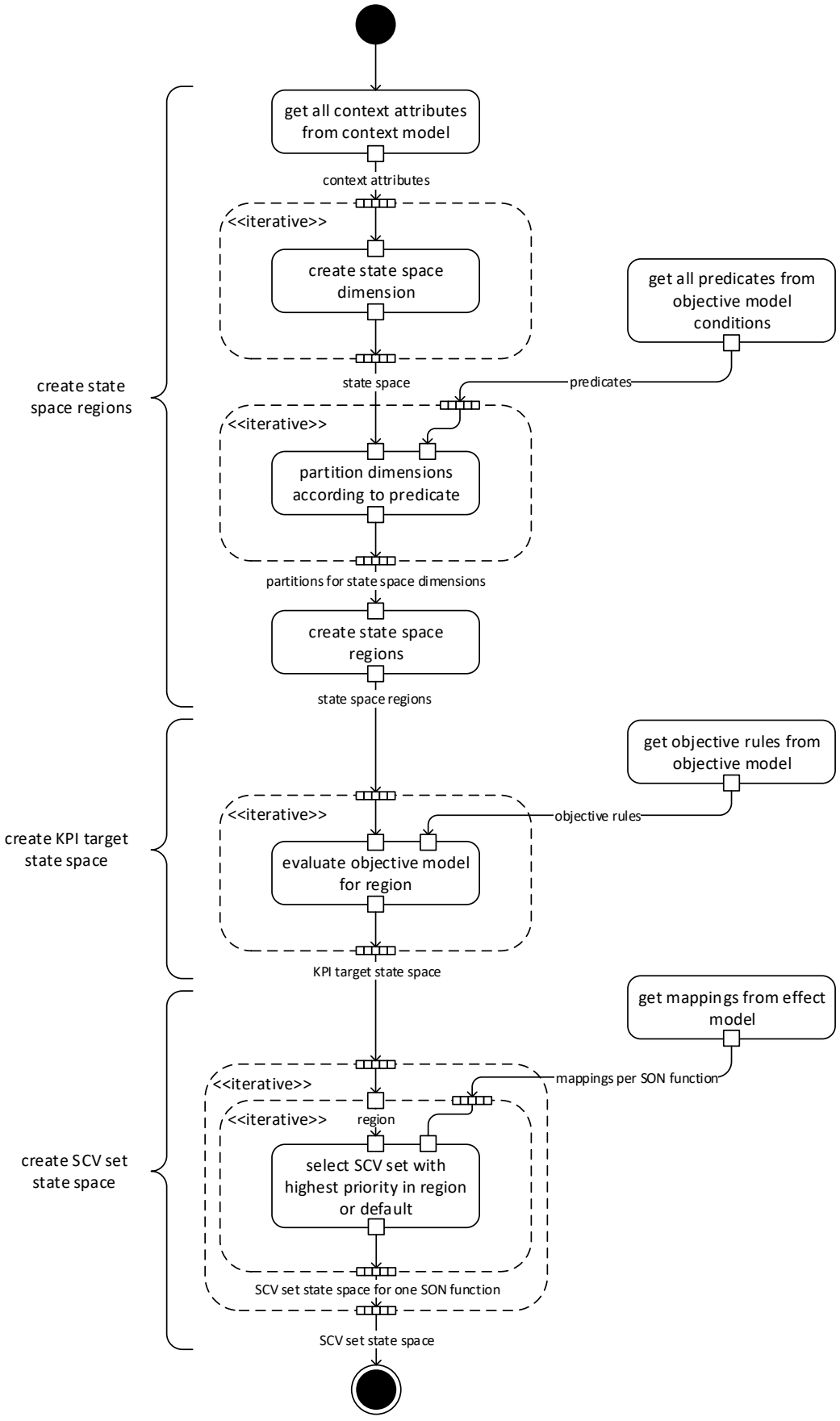


Figure 4.3.: SCV set policy derivation algorithm in PBSM (noted in UML 2)

For parameters with a discrete scale, such as the location, the partitioning is performed as follows: Regions are defined by the values of the different predicates. For parameters with a continuous scale, such as the time, the value of the predicate is a range of possible values and hence, the dimension is split into regions according to upper and lower limits of the range which is defined in the condition parts of the technical objectives. Consequently, a region ρ_i is defined as the cross product of a set of predicates π_{a_j}, one for each context attribute a_j, i.e.,

$$\rho_i = \pi_{a_1} \times \pi_{a_2} \times \cdots \times \pi_{a_{|A|}} \tag{4.14}$$

For instance, consider the following predicate *time in [08:00, 17:59]*. Here, the dimension for the context attribute time would be split into three partitions: *[00:00, 07:59]*, *[08:00, 17:59]*, and *[18:00, 23:59]*. After partitioning the dimensions for all objective rules, the state space regions are defined as the elements of the cross product of the partitions of all dimensions. For instance, one region could be defined by the tuple *(time in [18:00, 23:59], location = urban)*.

Note that the number of regions grows exponentially. For instance, a context model with ten parameters and one threshold for each parameter results in 2^{10} regions. In general, the number of regions $|P|$ in a context space is defined as:

$$|P| = \prod_{i=1}^{|A|} |\Pi_{a_i}| \tag{4.15}$$

with Π_{a_i} being the set of predicates that belong to a certain attribute a_i. Consequently, the partitioned context state space P is defined as a set containing all regions:

$$P = \left\{\rho_1, \rho_2, ..., \rho_{|P|}\right\} \tag{4.16}$$

4.2.4.2. Generation of KPI Target State Space

In the second step of the algorithm, the objective manager determines KPI targets and their priorities in each region. Since all contexts in a region trigger the same objective rules, this can be done by picking a random state from the region and evaluating the objective model for it. The result, after doing this for all regions, is a KPI target state space T with a set of KPI targets τ_{ρ_j} for each region, i.e.

$$T = \left\{\tau_{\rho_1}, \tau_{\rho_2}, ..., \tau_{\rho_{|P|}}\right\} \tag{4.17}$$

with τ_{ρ_j} being a set of applicable KPI targets in region ρ_j. Applicable thereby means that a region ρ_j is a subset or equal to the condition χ_i of the technical objective o_i, defined by the following function:

$$\text{isApp}(o_i) = \begin{cases} true & \text{if } \rho_j \subseteq \chi_i \\ false & \text{otherwise} \end{cases} \tag{4.18}$$

Consequently, τ_{ρ_j} is defined as the union of all matching KPI targets, i.e, where the function isApp(o_i) returns true:

$$\tau_{\rho_j} = \{o_i | 1 \leq i \leq |O| \wedge \text{isApp}(o_i) = true\} \tag{4.19}$$

Note that it is possible that a KPI target appears several times in a region with different assigned priorities.

The KPI target state space is not just an intermediate product of the algorithm but can also be used for validation and verification of the objective model. On the one hand, the users of the system can inspect KPI targets and their priorities for all regions and validate that the objective rules correctly represent their requirements. On the other hand, the system can verify that there are no two KPI targets with the same priority within a region, i.e., there is no confusion in the priority order of the KPI targets.

4.2.4.3. Generation of SCV Set State Space

In the third step of the algorithm, the system determines the SCV sets for each region based on the KPI target state space. This is an iterative mapping process for each region ρ_i and each SON function f_j: from the SCV sets in the function-specific model for f_j, the system selects the one whose KPI target has the highest priority in τ_{ρ_i}. If none of the effects in f_j's effect model matches any KPI target in τ_{ρ_i}, the system selects the default SCV set $s^{f_j}_{\text{default}}$. Note that the priorities of the technical objectives in τ_{ρ_i} are unique as defined in Equation 4.8. This leads to the following function selectSet(τ_{ρ_i}, f_j):

$$\text{selectSet}(\tau_{\rho_i}, f_j) = \begin{cases} \text{map}(\text{eq}(t_l)) & \text{if } \exists p_l \forall (t_m, p_m) \in \tau_{\rho_i}, \text{eq}(t_m) \in E^{f_j}, l \neq m : \\ & \quad p_l > p_m \\ s^{f_j}_{\text{default}} & \text{otherwise} \end{cases} \tag{4.20}$$

with eq(t_l) delivering the effect indicator in the effect model which is equivalent to t_l. map(eq(t_l)) names a function that maps the selected highest prioritized KPI target to the respective SCV set $s \in S^{f_j}$. Since there are no two SCV sets with the same effect indicator, this function returns exactly one SCV set.

Since this function only selects the SCV set for one specific SON function, σ_{ρ_i} is a set of SCV sets, one set for each SON function $f_j \in F$:

$$\sigma_{\rho_i} = \left\{\text{selectSet}(\tau_{\rho_i}, f_1), \text{selectSet}(\tau_{\rho_i}, f_2), ..., \text{selectSet}(\tau_{\rho_i}, f_{|F|})\right\} \tag{4.21}$$

Finally, the result of this process is an SCV set state space Σ which is a set of SCV sets σ_{ρ_i} for each region ρ_i, i.e.:

$$\Sigma = \left\{\sigma_{\rho_1}, \sigma_{\rho_2}, ..., \sigma_{\rho_{|P|}}\right\} \tag{4.22}$$

Based on the SCV set state space, the algorithm can finally compile the SCV set policy. The SCV set policy is a set of IF-THEN rules, referred to as SCV set policy rules which, based on some condition over the context, define configuration sets for the SON functions. A simple approach to build the SCV set policy is to create a policy rule for each region and each SON function which sets the corresponding SCV sets. Thereby, the components of the region tuples are translated into the conjunctive condition of the SCV set policy rule. This, of course, results in a large number of policy rules. An approach, which overcomes this shortcoming, creates the policy rules by combining neighboring regions with equal SCV sets. Note that the SCV set policy is complete and conflict-free because the objective model has a strict order of the priorities and the effect model is unambiguous. In other words, there is always exactly one possible SCV sets for each SON function in every context defined.

4.2.5. Policy System

As described in Section 3.2.5, the policy system in PBSM also consists of a policy repository, a PDP and a PEP. The output policy generated by the objective manager is stored in a policy repository and consulted at any time a triggering situation is monitored. In this case, the PDP gets the current operational context of each network entity as input. Based on the current context, the PDP can decide which policy rules are applicable. Note that there is only one applicable rule for each SON function, since the SCV set policy is complete and conflict-free. As a final step, the PEP determines whether the respective SON function is already configured accordingly or, otherwise, deploys the SCV set to the SON function.

4.3. Example

In order to get a better understanding of the PBSM work flow, an example shall illustrate the particular steps. Therefore, the context model depicted in Listing 4.1, the objective model depicted in Listing 4.3 and the effect model depicted in Listing 4.4 serve as exemplary input models and basis for the further calculations.

4.3.1. Generation of Partitioned Context State Space

In the first step of the objective manager, context attributes and their values are investigated. In the example there are two dimensions: the time with the continuous scale *[00:00, 23:59]* and the location with a discrete scale over the values *rural* and *urban*. While the discrete values for location can be easily identified from the technical objective conditions, it is more complicated for the time since this is an attribute with a continuous value scale. Looking into the IF parts of the objective

rules in Listing 4.3 shows, that it has to be distinguished between three time slots: *[00:00, 07:59]*, *[08:00, 17:59]* and *[18:00, 23:59]*. This spans the partitioned context state space illustrated in Figure 4.4.

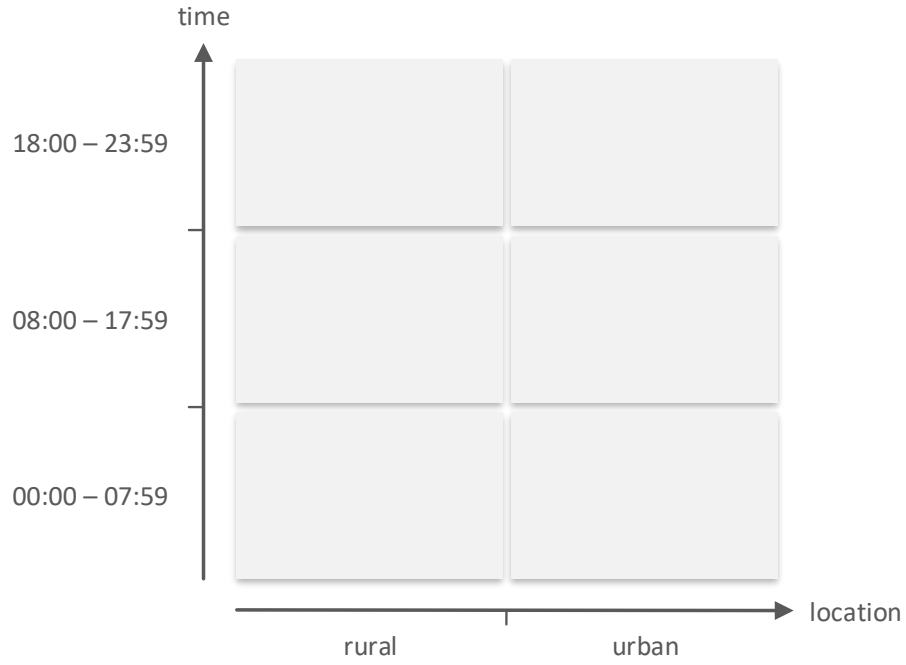


Figure 4.4.: Partitioned context state space for two context attributes *time* and *location*, each rectangle forming one region in the context state space

4.3.2. Generation of KPI Target State Space

In the second step of the PBSM objective manager algorithm, the KPI target state space is built. That is, the objective model is scanned for applicable KPI targets and their priorities in each region. For instance, in the region *(time in [18:00, 23:59], location = urban)* only the second and the sixth objective rules in Listing 4.3 apply, thus, defining the KPI target *DCR_MIN*, i.e., the minimization of the dropped call rate, with priority 2 and EC_MIN, i.e., the minimization of the energy consumption, with priority 6. Repeating this step for all regions in the partitioned context state space leads to the KPI target state space illustrated in Figure 4.5.

4.3.3. Generation of SCV Set State Space

The final step of the objective manager comprises the determination of optimal SCV sets for each SON function. In this example, four SON functions are used, namely MLB, CCO, ES and MRO. Based on the KPI target state space, the objective manager selects the SCV set for each SON function which fulfills the highest priority of a region. The resulting SCV set state space is shown in Figure 4.6.

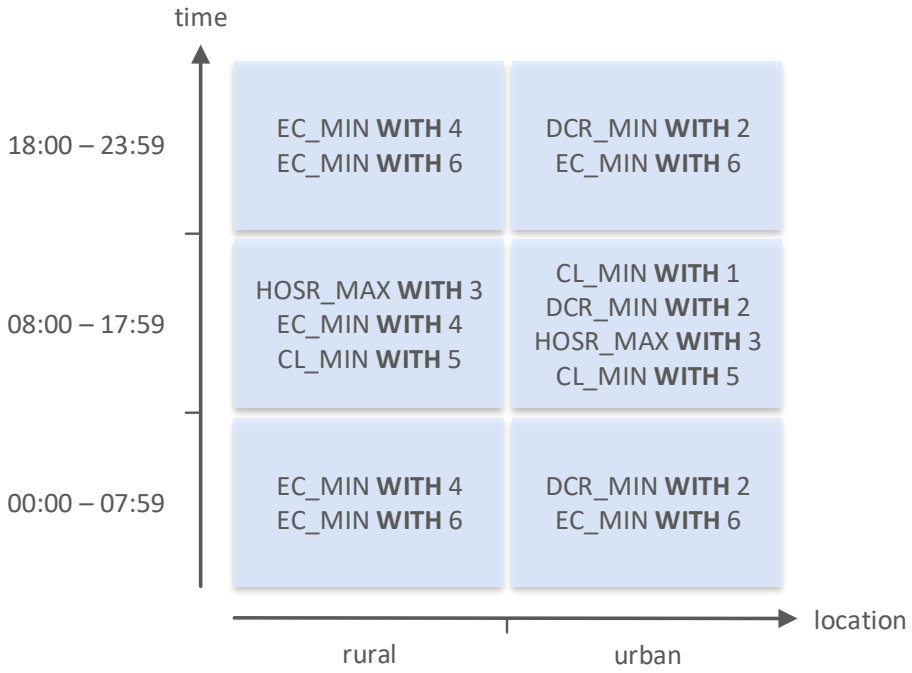


Figure 4.5.: KPI target state space with applicable KPI targets for each region

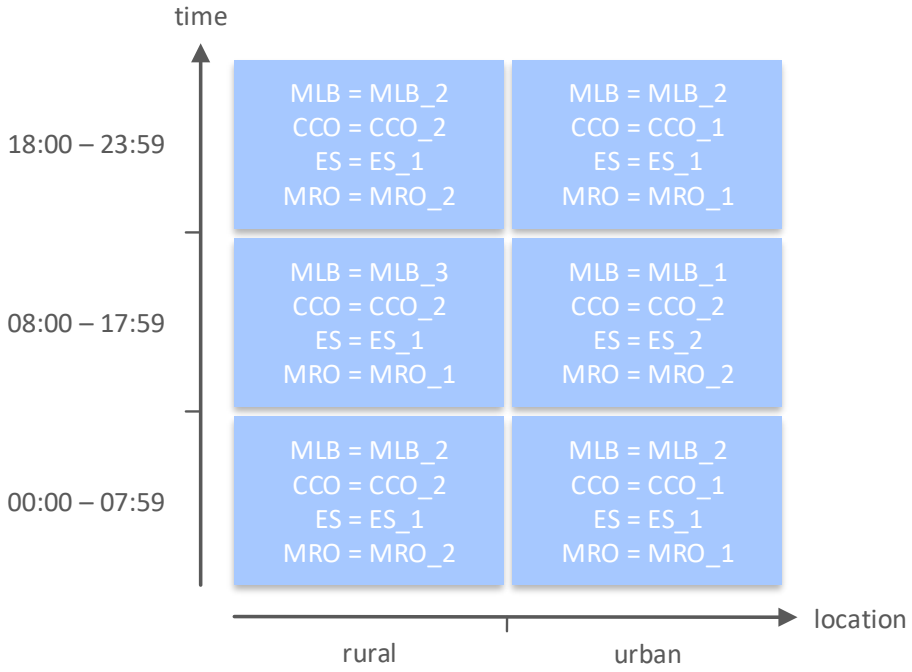


Figure 4.6.: SCV set state space with best possible SCV sets for each region

For instance, in the region *(time in [18:00, 23:59], location = urban)* the SCV set for MLB is *MLB_2* because the KPI target with the highest priority matching provided effects in the effect model is *EC_MIN*, the minimization of the energy consumption. Similarly, the SCV set for MRO is *MRO_2* because no effect in the effect model matches the KPI targets in this region and thus, the default SCV set is selected.

4.3.4. Policy Generation

Based on the SCV set state space, the output SCV set policy can be generated. Thereby, neighboring regions with the same result, i.e., with the same SCV sets for all SON functions, are combined such that the number of policy rules is reduced. The resulting SCV set policy is depicted in Listing 4.5.

```
{
  IF time in [00:00, 07:59] OR time in [18:00, 23:59]
    THEN MLB = MLB_2
  IF time in [08:00, 17:59] AND location = rural
    THEN MLB = MLB_3
  IF time in [08:00, 17:59] AND location = urban
    THEN MLB = MLB_1
  IF (time in [00:00, 07:59] OR time in [18:00, 23:59])
    AND location = urban
      THEN CCO = CCO_1
  IF time in [08:00, 17:59] OR location = rural
    THEN CCO = CCO_2
  IF time in [00:00, 07:59] OR time in [18:00, 23:59]
    OR location = rural
      THEN ES = ES_1
  IF time in [08:00, 17:59] AND location = urban
    THEN ES = ES_1
  IF ((time in [00:00, 07:59] OR time in [18:00, 23:59])
    AND location = urban) OR (time in [08:00, 17:59]
    AND location = rural)
      THEN MRO = MRO_1
  IF ((time in [00:00, 07:59] OR time in [18:00, 23:59])
    AND location = rural) OR (time in [08:00, 17:59]
    AND location = urban)
      THEN MRO = MRO_2
}
```

Listing 4.5: SCV set policy based on the SCV set state space

4.3.5. Configuration Deployment

Depending on the current operational context, SON function instances installed on cells in a mobile network can be configured. For instance, imagine a cell in an

urban area at 8 o'clock pm, building the operational context *(time = 20:00, location = urban)*, the first, fourth, sixth and eighth policy rules are applicable. The PEP then configures that cell according to the SCV sets in Listing 4.6 in case they are differently configured at the moment. Otherwise, no action is required since the cell is already configured in an optimal way.

```
{
  MLB = MLB_2
  CCO = CCO_1
  ES = ES_1
  MRO = MRO_1
}
```

Listing 4.6: Applied SCV sets in the operational context *time = 20:00, location = urban*

4.4. Related Work

Most of the prior work on goal- or objective-driven management of autonomic systems assumes that the internal logic of the autonomic functions, i.e., the SON functions, can be directly defined as a policy. This enables the operator to gain full knowledge of the SON function algorithms, allows a prediction of their behavior and effects and facilitates their design such that they interfere as little as possible, i.e., such that no two SON functions affect the same objective. However, this requires the manufacturers to provide a detailed action model and reveal the details of their SON functions.

In the context of autonomic computing research, the Self-Net project [KN10] defines the policy as a set of rules describing the behavior of the system at a low level, i.e., which NCPs should be changed in response to some problem. When lifting this approach to a higher abstraction level, the rules describe the SCV sets for SON functions, but it is left to the operator to define a conflict-free policy satisfying the objectives. The Generic Autonomic Network Architecture (GANA) architecture [Ari+10] can be seen as a similar approach.

Several projects working on SON touch SON management, but they mainly describe abstract ideas without providing detailed solutions regarding the problems and objectives discussed in this chapter. SOCRATES [Kür+10] presents an idea describing the refinement of operator policies into SON function-specific policies such that the SON functions are configured to achieve a common goal. UniverSelf [Tsa+13b] presents a governance component controlling Network Empowerment Mechanisms (NEMs) that refer to SON functions. This component translates service-level goals that are related to the objectives described in this paper to NEM-level policies which compare to the SCV sets. This translation is based on policy templates defined for the service-level goals, i.e., they relate to the effect

model. The Cognitive Network Management under Uncertainty (COMMUNE) project [Bar+13] describes a system model called GARSON containing a policy control plane that controls the cognitive network functions, i.e., SON functions, via "high-level goals". Therefore, the policy plane has a set of policy rules defining the configuration parameters of the algorithms in specific situations.

5 Objective-driven SON Management

In Chapter 4 an approach has been presented that overcomes the manual gap and focuses on a policy-based representation of input models and the output generated by the objective manager. Even though ODSM also applies policy-based techniques, it describes an approach that focuses on the definition of a more precise objective model and the better fulfillment of technical objectives by means of a more accurate effect model. Hence, this SON management approach is called objective-driven. ODSM can be seen as an enhancement or a further expansion stage compared to PBSM. In this chapter, especially the delta to the previous SON management is described which means a new objective model, an enhancement to the initial effect model and a new methodology for deriving the SCV set policy in the first place. It starts with a motivation identifying problems of the PBSM approach. Afterwards, models and the methodology will be described in a theoretical manner. The chapter is concluded with an example illustrating the approach and a related work section.

5.1. Motivation

In Chapter 4 a concept for an objective manager is described that automatically selects the best SCV sets for the SON functions with respect to the operator objectives. It uses three types of models as input: first, an effect model for each SON function defining the KPIs that are optimized or worsened by a specific SCV set, second, an objective model describing the operator's context-specific and prioritized KPI targets and third, a context model describing context attributes of the network and cells operating in it. The objective manager generates an SCV set policy whose execution deploys those SCV sets to the SON functions that optimize the highest prioritized KPI in the current context. Despite being an important step towards automated network management, this SON management approach has three shortcomings.

First, the effect model can only describe the maximization or minimization of a KPI value but not that a specific SCV set keeps the KPI value within some range.

The same applies for the objective model. Second, the ranking of the KPI targets through operator defined priorities does not allow a trade-off between the objectives, if not all KPI targets can be satisfied. Thus, the network will always be optimized to only the highest ranked objective. Third, in the PBSM approach, KPI targets used in the objective model and predicted effects in the effect model need to exactly match each other which might be too inflexible in practice. On the one hand, that means that they must have the same name and meaning which is not a problem when using only standardized KPIs or using a translation model for non-standardized KPIs. On the other hand, also the targets must match each other which significantly delimitates an MNO when defining his or her objectives.

The ODSM approach overcomes the aforementioned shortcomings of the PBSM approach and thereby, aims mainly at one objective of this thesis (cf. Figure 5.1):

Objective 2 *Automatically generate realistic and complete input models which are continuously updated according to the current network state such that they optimally support the SON management system in finding the best possible network configuration.*

Figure 5.1.: Objectives of ODSM

Due to a new structure of the input models, the reasoning process performed by the objective manager changes accordingly.

For the sequel of this chapter, the following formalizations of the SON-enabled mobile network as done in the PBSM approach, also apply for ODSM:

$$F = \{f_1, f_2, ..., f_{|F|}\} \tag{5.1}$$

$$S^f = \left\{s_1^f, s_2^f, ..., s_{|S^f|}^f\right\} \tag{5.2}$$

$$K = \{k_1, k_2, ..., k_{|K|}\} \tag{5.3}$$

5.2. Approach

To overcome the shortcomings described in Section 5.1, this chapter presents an improved approach. First, the expressiveness of the effect model is enhanced such that a certain SCV set influences a KPI by maximization, minimization, neutrality, or by keeping it within a specific interval. Furthermore, the objective model allows to express concrete value ranges for the KPIs that the SON should satisfy. With this differentiation, the objective manager is enabled to better adapt the SCV sets to the operator objectives and can identify conflicts between different SCV sets. Finally, KPI targets are weighted instead of prioritized in the objective model allowing to select the best SCV set for a weighted satisfaction of the KPI targets. In summary, ODSM clearly allows more complex models compared to PBSM reflecting the requirements on KPI target setting in real systems.

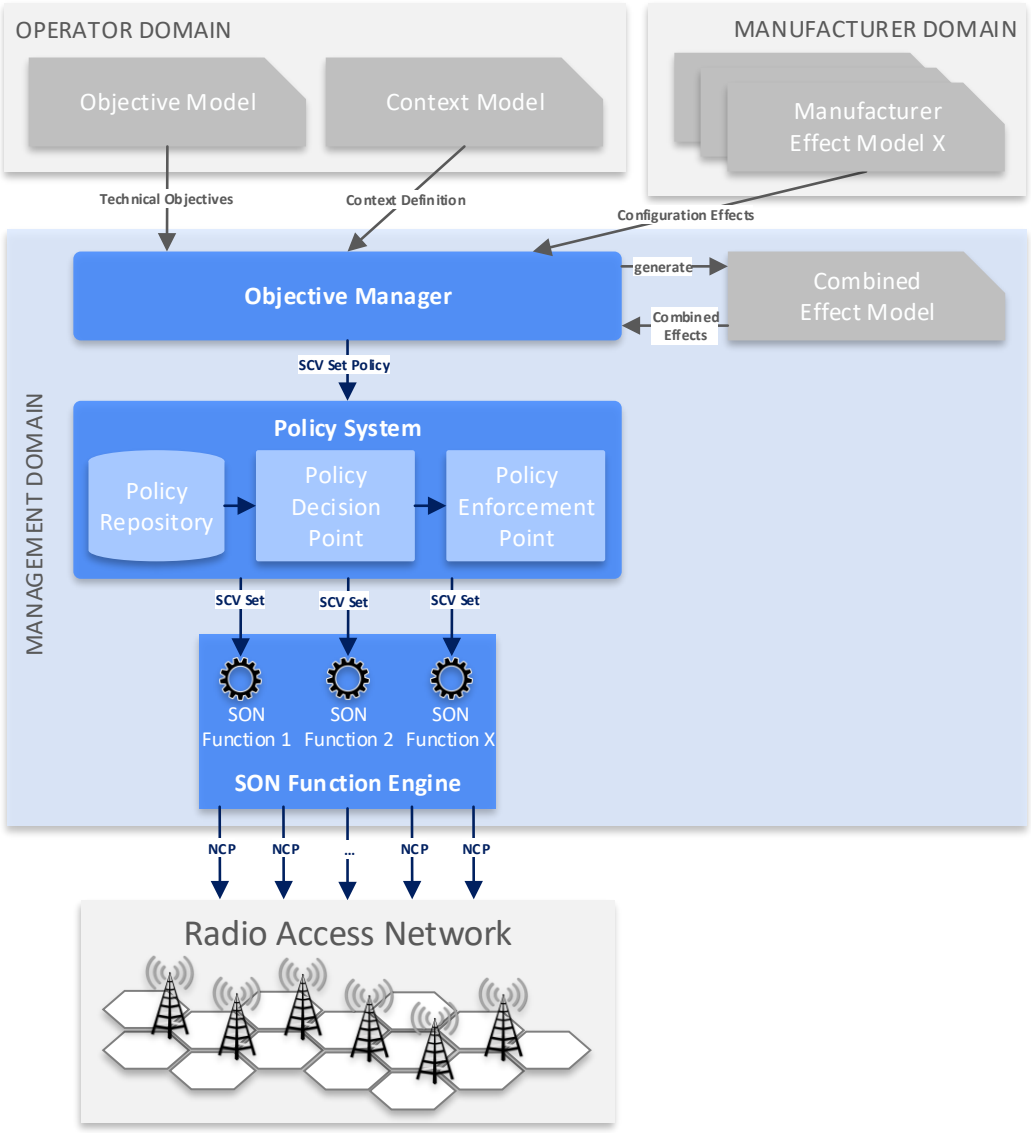


Figure 5.2.: Overview of ODSM

Figure 5.2 shows the architecture of the ODSM approach. In general, the type of input models are the same as in the PBSM approach. The main difference are the structure of these already known models and an extension to the effect model, the so-called combined effect model. However, the combined effect model does not belong to the domain of the SON function manufacturer since it is an intermediate product of the reasoning process performed by the objective manager. While the manufacturer-provided SON function-specific effect models are used only once for the generation of the combined effect model, the latter is used any time the SCV set policy needs to be recalculated, e.g., when technical objectives change. Thus, roughly speaking, the combined effect model replaces the manufacturer-provided effect models.

The context model as well as the policy system are equal to the ones presented in Section 4.2.1 and Section 4.2.5 and are not explained in more detail in this chapter.

5.2.1. Objective Model

The objective model defines the operator's objectives, i.e., for each KPI a target value together with a weight. The latter represents the importance of the objective and thus, allows to trade-off the objectives against each other in case they cannot be satisfied simultaneously. This is especially useful if there are conflicting KPI targets like the minimization of the energy consumption and the minimization of the CL. The ODSM approach is based on a weighted sum preference model which is very common due to its simplicity [BD09]. An objective can furthermore depend on operational context, i.e., it can vary for different context attributes such as time of the day, cell types and locations, or traffic patterns. A concrete syntax to express the objective model are production rules, as depicted in Listing 5.1.

```
IF condition THEN KPI target WITH weight
```

Listing 5.1: Technical objective rule in ODSM

In Listing 5.2 an exemplary objective model is depicted with conditioned and weighted KPI target rules. Thereby, three KPIs, namely DCR, HOSR and CL are considered. The dots are placeholders for a set of KPI targets in the respective operational context.

Formally, the target for a KPI is a set of acceptable values for the KPI, whereas the weight for a KPI is a real valued number between 0 and 1. Thereby, a higher weight represents a higher importance. Both the KPI target t_k and the weight w_k, make up an objective $o_k \in O_k$ for a KPI $k \in K$.

$$o_k = (t_k, w_k) \tag{5.4}$$

Thereby, $t_k \subseteq \mathrm{Dom}(k)$ and $w_k : [0,1]$. For instance, the operator objective to keep the DCR below 2% with a medium weight is represented as $o_{\mathrm{DCR}} = ([0, 0.02], 0.5)$.

The objective model can be seen as a function that maps a specific operational context $\chi \in X$ to a tuple of objectives for all $|K|$ KPIs, i.e.:

$$\mathrm{OM} : X \mapsto O_{k_1} \times O_{k_2} \times \cdots \times O_{k_{|K|}} \tag{5.5}$$

```
1 {
    IF time in [00:00, 07:59] AND location = urban THEN ...
3   IF time in [00:00, 07:59] AND location = rural THEN ...
    IF time in [08:00, 17:59] AND location = urban
5     THEN DCR ≤ 0.02 WITH 0.5
    IF time in [08:00, 17:59] AND location = urban
7     THEN HOSR ≥ 0.95 WITH 0.2
    IF time in [08:00, 17:59] AND location = urban
9     THEN CL ≤ 0.5 WITH 0.3
    IF time in [08:00, 17:59] AND location = rural THEN ...
11  IF time in [18:00, 23:59] AND location = urban THEN ...
    IF time in [18:00, 23:59] AND location = rural THEN ...
13 }
```

Listing 5.2: Exemplary objective model in ODSM

5.2.2. Effect Model

In principle, an effect model defines predicted effects on KPI values, by a specific SCV set for a SON function, without considering other SON functions. Analogous, the combined effect model makes a forecast on the effect of a combination of SCV sets, i.e., one for each SON function.

The model for one SON function f defines a mapping FM^f between a set of possible SCV sets S^f and their effect indicators on the KPIs E^f.

$$\mathrm{FM}^f : S^f \mapsto E^f \tag{5.6}$$

$$E^f = \left\{\varepsilon_1^f, \varepsilon_2^f, ..., \varepsilon_{|E^f|}^f\right\} \tag{5.7}$$

Thus, the entire effect model EM is defined as a set of function-specific submodels FM^f, one for each SON function:

$$\mathrm{EM} = \left\{\mathrm{FM}^{f_1}, \mathrm{FM}^{f_2}, ..., \mathrm{FM}^{f_{|F|}}\right\} \tag{5.8}$$

The function-specific effect models are usually created by the manufacturer of the respective SON function, but the SON functions themselves are generally provided as black boxes in order to reveal as little information as possible about

their internal logic. The concrete value a KPI takes under some SCV set can depend on the concrete mobile network, hence, the effect models need to provide as much information as necessary for operations but as little as possible. A SON function optimizes one or several dedicated KPIs, e.g., an MRO function optimizes HOSR, and the SON function is usually configured such that it keeps the KPI value above or below some threshold, e.g., DCR $\leq$ 2% and HOSR $\geq$ 99%. However, a SON function might affect a KPI in less predictable ways, i.e., the manufacturer can solely predict that the KPI value is maximized or minimized, or a SON function might have no effect at all on a certain KPI.

Consequently, an effect indicator $\varepsilon^f \in E^f$ is represented by a set of tuples k_i, p_i, with $k_i \in K$ indicating the affected KPI and p_i specifying the predicted value for k_i.

$$\varepsilon^f = \left\{(k_1, p_1), (k_2, p_2), ..., (k_{|K|}, p_{|K|})\right\} \tag{5.9}$$

The predicted value p_i can be defined as $\leq x$, $\geq x$, $\uparrow$, $\downarrow$, $\leftrightarrow$ with $x \in \text{Dom}(k_i)$ indicating the effects of keeping k_i below or above some threshold x as well as the maximization, minimization and non-influence of k_i. For instance, $(\text{DCR}, \leq 0.02)$ describes the effect that the DCR is kept below 2%. For simplicity, it is assumed that there is exactly one effect tuple for each KPI, i.e., $|\varepsilon^f| = |K|$.

The formalized effect $\varphi_s(k)$ of a KPI k for an SCV set s of SON function f is interpreted as a subset of the KPIs domain.

$$\varphi_s(k) = \begin{cases} [\max(\text{Dom}(k))] & \text{if } (k, \uparrow) \in \text{FM}^f(s) \\ [\min(\text{Dom}(k))] & \text{if } (k, \downarrow) \in \text{FM}^f(s) \\ [\text{Dom}(k)] & \text{if } (k, \leftrightarrow) \in \text{FM}^f(s) \\ [\min(\text{Dom}(k)), x] & \text{if } (k, \leq x) \in \text{FM}^f(s) \\ [x, \max(\text{Dom}(k))] & \text{if } (k, \geq x) \in \text{FM}^f(s) \end{cases} \tag{5.10}$$

For instance, $(\text{DCR}, \leq 0.02)$ is interpreted as $\varphi_s(\text{DCR}) = [0, 0.02]$.

In the first two cases of Equation 5.10, an interval containing only one value means that $\varphi_s(k)$ can only take on this specific value.

Listing 5.3 shows an exemplary effect model as it is used in ODSM, for two SON functions MRO and MLB.

5.2.3. Methodology

The basic idea of the ODSM approach is to separate the technical knowledge about how a SON function affects KPIs through a dedicated SCV set, from the definition of the operator objectives. The objective manager puts the separate function-specific effect models and the objective model together and determines

```
1 {
    MLB Model
3     IF MLB_1 THEN DCR = ↔ AND HOSR = ↓ AND CL ≤ 0.6
      IF MLB_2 THEN DCR ≤ 0.02 AND HOSR = ↔ AND CL ≤ 0.5
5
    MRO Model
7     IF MRO_1 THEN DCR = ↓ AND HOSR = ↔ AND CL ≤ 0.65
      IF MRO_2 THEN DCR = ↔ AND HOSR ≥ 0.99 AND CL = ↔
9 }
```

Listing 5.3: Exemplary effect model in ODSM, one function-specific effect model per SON function

the best combination of SCV sets, i.e., one SCV set per SON function per network cell.

Figure 5.3 depicts a functional overview of the SON management system and the configuration process which consists of two steps. First, the SON function model combination step merges the models of all available SON functions into a combined effect model, which allows the estimation of the network performance for combinations of SCV sets. This step needs to be performed whenever an effect model changes or a new model is added. The second step of the algorithm combines the generation of context state space regions and assigning applicable KPI targets to these regions. This process is equal to the first and second step in Section 4.2.4 and hence, is not presented in full detail in Figure 5.3. For the details regarding the generation of a KPI target state space see Section 4.2.4. Third, the SCV set selection step determines the applicable objectives for the current cell context and evaluates the network performance estimations from the combined effect model against these objectives. Thereby, the objective manager can select the best combined SCV set for all SON functions in every context, generates an output SCV set policy and afterwards, the policy system deploys the changed SCVs to the network if they differ from the current SCV set. Note that the latter step depends on the context of the cell and is performed iteratively for each and every cell depending on the SCV set policy. The decision for a reconfiguration of network cells is triggered by events which are raised by entities external to the actual SON management, e.g., a timer raising events in regular intervals, or a CM system informing the SON management system about changes in the network layout.

The basic idea for combining the function-specific effect models and their joint evaluation is founded on a set-based description of the possible KPI values under some SCV set, and the expectations of the operator regarding the KPI values. This allows three interpretations: a pessimistic view, an optimistic view and a function-specific view.

Optimistic View A KPI value is possible for a combined SCV set if it is possible for a single SCV set of the combination.

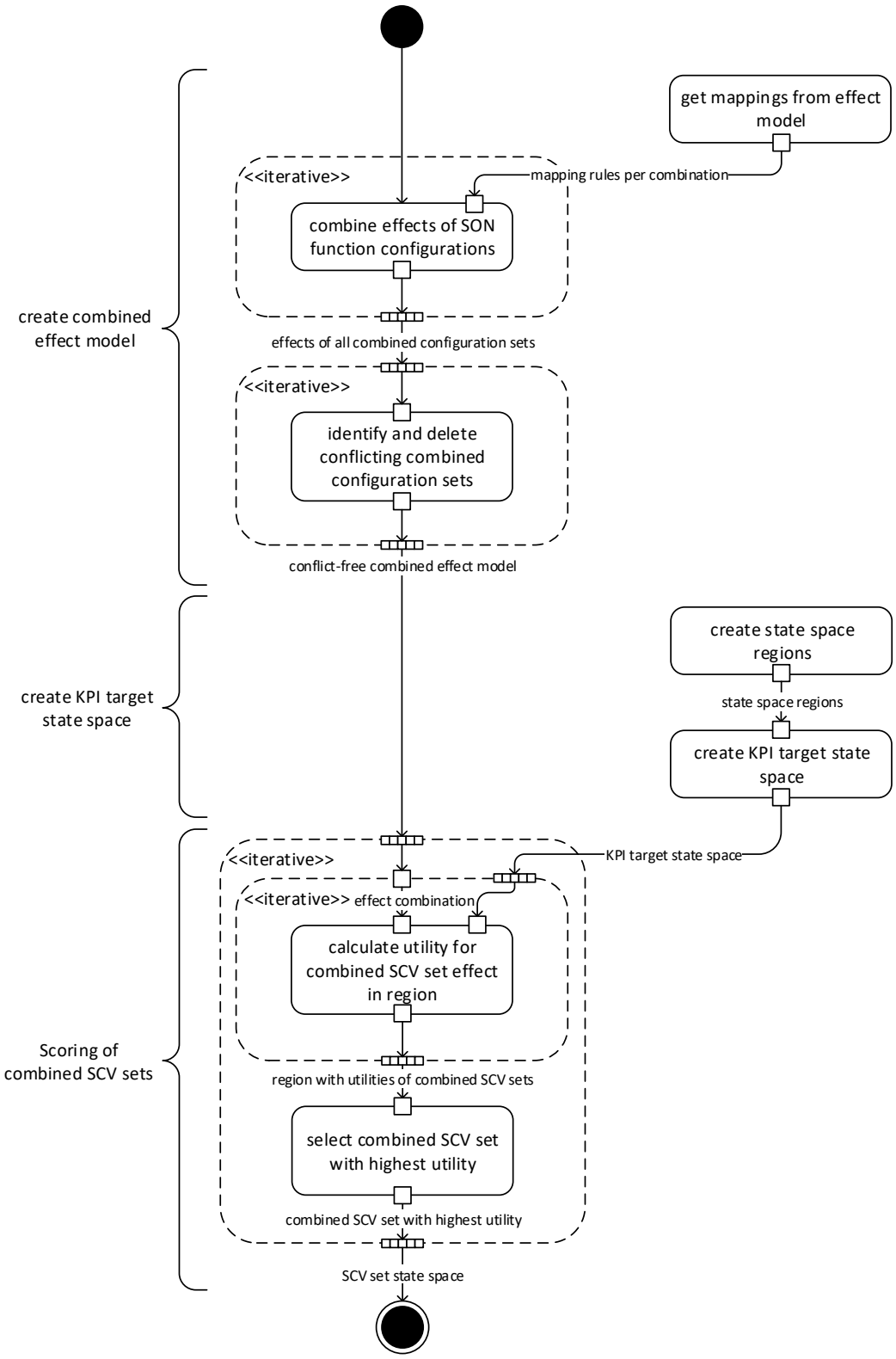


Figure 5.3.: SCV set policy derivation algorithm in ODSM (noted in UML 2)

Pessimistic View A KPI value is possible for a combined SCV set if it is possible for each and every single SCV set of the combination.

Function-specific View A KPI value is possible for a combined SCV set if it is possible for the single SCV set whose SON function has the highest impact on that KPI.

In all cases, an objective is satisfied if the possible KPI values are a subset of those KPI values expected by the operator. Hence, the pessimistic view sees all SCV set combinations which definitely meet the objectives as satisfying, whereas the optimistic view sees all SCV set combinations which maybe meet the objectives as satisfying. The function-specific view relies on the fact that SON functions usually aim at the optimization of one (or mostly one) KPI. Giving an MNO the possibility to decide about the type of combination marks a further important step towards gaining trust in the SON management system.

5.2.3.1. Effect Model Combination

The main goal of the effect model combination is to predict the effects on KPIs if several SON functions with some SCV set combination are concurrently active. Thereby, the set of possible SCV set combinations is the cross product of the SCV sets for all SON functions.

$$\Sigma : S^{f_1} \times S^{f_2} \times \cdots \times S^{f_{|F|}} \tag{5.11}$$

In the optimistic view, the combined effect $\tilde{\varphi}_\sigma(k)$ of a SCV set combination $\sigma \in \Sigma$ on a KPI k is built by the intersection of the effects of the different SCV sets, i.e.:

$$\tilde{\varphi}_\sigma(k) = \bigcap_{1 \le i \le |F|, s = \mathrm{proj}_i(\sigma)} \varphi_s(k) \tag{5.12}$$

with $\mathrm{proj}_i(\sigma)$ being the projection on the ith element of σ.

In the pessimistic view, the combined effect $\tilde{\varphi}_\sigma(k)$ of a SCV set combination $\sigma \in \Sigma$ on a KPI k is built by the union of the effects of the different SCV sets, i.e.:

$$\tilde{\varphi}_\sigma(k) = \bigcup_{1 \le i \le |F|,\ s = \mathrm{proj}_i(\sigma)} \varphi_s(k) \tag{5.13}$$

with $\mathrm{proj}_i(\sigma)$ again being the projection on the ith element of σ.

Table 5.1.: Overview of combined effects on a KPI with the domain Dom = $[0, 100]$ for two SCV sets, applying the optimistic combination

p		$\uparrow$	$\downarrow$	$\leftrightarrow$	≥ 1	≥ 3	≤ 1	≤ 3
	φ	$[100]$	$[0]$	$[0, 100]$	$[1, 100]$	$[3, 100]$	$[0, 1]$	$[0, 3]$
$\uparrow$	$[100]$	$[100]$						
$\downarrow$	$[0]$	$\varnothing$	$[0]$					
$\leftrightarrow$	$[0, 100]$	$[100]$	$[0]$	$[0, 100]$				
≥ 1	$[1, 100]$	$[100]$	$\varnothing$	$[1, 100]$	$[1, 100]$			
≥ 3	$[3, 100]$	$[100]$	$\varnothing$	$[3, 100]$	$[3, 100]$	$[3, 100]$		
≤ 1	$[0, 1]$	$\varnothing$	$[0]$	$[0, 1]$	$[1]$	$\varnothing$	$[0, 1]$	
≤ 3	$[0, 3]$	$\varnothing$	$[0]$	$[0, 3]$	$[1, 3]$	$[3]$	$[0, 1]$	$[0, 3]$

Table 5.2.: Overview of combined effects on a KPI with the domain Dom = $[0, 100]$ for two SCV sets, applying the pessimistic combination

p		$\uparrow$	$\downarrow$	$\leftrightarrow$	≥ 1	≥ 3	≤ 1	≤ 3
	φ	$[100]$	$[0]$	$\varnothing$	$[1, 100]$	$[3, 100]$	$[0, 1]$	$[0, 3]$
$\uparrow$	$[100]$	$[100]$						
$\downarrow$	$[0]$	$[0] \cup [100]$	$[0]$					
$\leftrightarrow$	$\varnothing$	$[100]$	$[0]$	$\varnothing$				
≥ 1	$[1, 100]$	$[1, 100]$	$[0] \cup [1, 100]$	$[1, 100]$	$[1, 100]$			
≥ 3	$[3, 100]$	$[3, 100]$	$[0] \cup [3, 100]$	$[3, 100]$	$[1, 100]$	$[3, 100]$		
≤ 1	$[0, 1]$	$[0, 1] \cup [100]$	$[0, 1]$	$[0, 1]$	$[0, 100]$	$[0, 1] \cup [3, 100]$	$[0, 1]$	
≤ 3	$[0, 3]$	$[0, 3] \cup [100]$	$[0, 3]$	$[0, 3]$	$[0, 100]$	$[0, 100]$	$[0, 1]$	$[0, 3]$

In the function-specific view, the combined effect $\tilde{\varphi}_\sigma(k)$ of a SCV set combination $\sigma \in \Sigma$ on a KPI k is built by the effect of the SCV set of the SON function with the biggest influence on that KPI, i.e.:

$$\tilde{\varphi}_\sigma(k) = \varphi_s(k) \tag{5.14}$$

with $1 \leq i \leq |F|, s = \text{proj}_i(\sigma)$ being the projection on the ith element of σ and i being the index of the SON function with the highest impact on k. The knowledge about which SONfunction most influences a KPI, must be externally provided by the MNO.

Table 5.3.: Overview of combined effects on a KPI with the domain Dom = $[0, 100]$ for two SCV sets, applying the function-specific combination; the horizontal SCV set is the one with a higher impact on the KPI

p		$\uparrow$	$\downarrow$	$\leftrightarrow$	≥ 1	≥ 3	≤ 1	≤ 3
	φ	$[100]$	$[0]$	$[0,100]$	$[1,100]$	$[3,100]$	$[0,1]$	$[0,3]$
$\uparrow$	$[100]$	$[100]$						
$\downarrow$	$[0]$	$[100]$	$[0]$					
$\leftrightarrow$	$[0,100]$	$[100]$	$[0]$	$[0,100]$				
≥ 1	$[1,100]$	$[100]$	$[0]$	$[0,100]$	$[1,100]$			
≥ 3	$[3,100]$	$[100]$	$[0]$	$[0,100]$	$[1,100]$	$[3,100]$		
≤ 1	$[0,1]$	$[100]$	$[0]$	$[0,100]$	$[1,100]$	$[3,100]$	$[0,1]$	
≤ 3	$[0,3]$	$[100]$	$[0]$	$[0,100]$	$[1,100]$	$[3,100]$	$[0,1]$	$[0,3]$

As part of this combination process, conflicting SCV sets of different SON functions are identified. Conflicts are SCV set combinations that might lead to an unstable and undesired system behavior such as a constantly oscillating network reconfiguration. A pair of SCV sets is in conflict if their possible effects on a KPI do not overlap, i.e., they do not agree on their optimization target and optimize against each other. In other words, the intersection (in case of the optimistic combination) or the union (in case of the pessimistic combination) of the formalized effects is empty, i.e., they have no common KPI value they both target.

$$\text{isConf}(s_i, s_j) = \exists k \in K.\varphi_{s_i}(k) \cap \varphi_{s_j}(k) = \varnothing \tag{5.15}$$

Consequently, an SCV set combination is conflicting if any two contained SCV sets are in conflict. The set of conflicting SCV set combinations Γ can thus be determined by:

$$\Gamma = \{\sigma \in \Sigma | \exists 1 \leq i \leq j \leq |F|.\text{isConf}(\text{proj}_i(\sigma), \text{proj}_j(\sigma))\} \tag{5.16}$$

Examples for combining KPI effects with an optimistic, a pessimistic and a function-specific view are given in Table 5.1, Table 5.2 and Table 5.3 respectively, whereby conflicts are gray shaded. Note that a conflict is defined in the same way for all types of combinations. In the function-specific view it may be not obvious why some combinations are marked as conflicts since only the effect of the most important SON function for a KPI is considered. However, even though some SCV sets do not have a huge impact on specific KPIs, their combination with other SCV sets should be sorted out in case they are possibly working against each other. Furthermore note that in the pessimistic view, neutral effects ($\leftrightarrow$) are interpreted as an empty range.

The combined effect model, which is the result of the first step of the objective manager process, consists of two sets: the set of all conflict-free SCV set combinations Σ^Γ and the set of their combined effects Φ.

$$\Sigma^\Gamma = \Sigma \setminus \Gamma \tag{5.17}$$

Note that the dot $\cdot$ in $\tilde{\varphi}_\sigma(\cdot)$ represents the union of effects for all network KPIs.

$$\Phi = \left\{ \tilde{\varphi}_\sigma(\cdot) | \sigma \in \Sigma^\Gamma \right\} \tag{5.18}$$

5.2.3.2. Generation of KPI Target State Space

This step summarizes the generation of a partitioned context state space based on an evaluation of the context model and the objective model's conditions as well as the determination of applicable KPI targets for each region in the partitioned context state space. Equal to the generation of the partitioned context state space in PBSM, the result of this step is a context state space P containing the set of $|P|$ context regions ρ_i.

$$P = \left\{ \rho_1, \rho_2, ..., \rho_{|P|} \right\} \tag{5.19}$$

Afterwards, the objective manager determines KPI targets and associated weights by means of a function isApp(o_k) (cf. Section 4.2.4.2) resulting in the KPI target state space:

$$T = \left\{ \tau_{\rho_1}, \tau_{\rho_2}, ..., \tau_{\rho_{|P|}} \right\} \tag{5.20}$$

Formally, T looks similar to the one in PBSM, however, there are two differences. First, $\tau_{\rho_i} \in T$ consist of objectives that define precise KPI targets that are weighted instead of prioritized minimum or maximum indications. Second, while in PBSM a KPI target can occur several times with different assigned priorities, in ODSM there is exactly one target defined per KPI, i.e.:

$$|\tau_{\rho_i}| = |K|, \ \forall \tau_{\rho_i} \in T \tag{5.21}$$

5.2.3.3. Scoring of Combined SCV Sets

In order to select optimal SCV sets, the objective manager iterates over all regions in the KPI target state space T and selects the best combined SCV set for each region based on the combined effect model and the objective model. Therefore, it calculates the utility for each applicable (i.e, conflict-free) combined SCV set $\sigma \in \Sigma^\Gamma$, i.e., its degree of satisfaction of the context-dependent operator objectives, and selects the SCV set combination with the highest utility. The following description outlines the selection process for a single network region ρ_i.

In order to calculate the utility for an SCV set combination $\sigma \in \Sigma^\Gamma$, the objective manager first gets the applicable objectives τ_{ρ_i} for the region ρ_i. Subsequently, it determines for each applicable objective $o_k = (t_k, w_k) \in \tau_{\rho_i}$ whether σ satisfies the objective based on its possible values in the combined effect model $\tilde{\varphi}_\sigma(\cdot) \in \Phi$. It

must be ensured that all possible states of the region configured with σ satisfy the objectives. Formally, this means that for a KPI $k \in K$, the satisfaction of the target t_k by a combined KPI value $\tilde{\varphi}_\sigma(k)$ is defined as:

$$\text{sat}(t_k, \tilde{\varphi}_\sigma(k)) = \begin{cases} 1 & \text{if } \tilde{\varphi}_\sigma(k) \subseteq t_k \\ 0 & \text{otherwise} \end{cases} \tag{5.22}$$

Since the objective model is based on the weighted sum preference model, the utility of a combined SCV set σ is, in principle, the sum of the satisfaction values multiplied with the objective weight over all KPIs. However, it is advisable to use normalized objective weights such that a utility of 1 means the satisfaction of all objectives. As a result, the utility for a combined SCV set σ is calculated as:

$$U(\sigma) = \sum_{k \in K} \text{sat}(t_k, \tilde{\varphi}_\sigma(k)) \frac{w_k}{\bar{w}} \tag{5.23}$$

with $\bar{w} = \sum_{i \in K} w_i$.

The combined SCV set σ with the highest utility $U(\sigma)$ is the one which satisfies the operator objectives the most. Hence, it should be selected and configured for the network region ρ_i. However, if $U(\sigma) < 1$, not all objectives could be satisfied by the SCV set in this specific region. This important feedback can be additionally provided to the operator.

In case that two or more combined SCV sets have the same utility, one of them is chosen randomly since they both fulfill the operator objectives to the same degree. Instead of choosing a random combined SCV set, another option would be to calculate a distance indicating how far a combined SCV set is away from fulfilling KPI targets. The combined SCV set with the lowest distance would then be selected in order to deploy not only a set with the highest degree of objective fulfillment, but also the one with the overall best possible KPI effects.

5.3. Example

In this section, the ODSM concept is exemplified with two SON functions MRO and MLB, each with two SCV sets according to the effect model, and three KPIs, namely DCR, HOSR, and CL. Note that all three KPIs all have the domain $[0, 1]$. Furthermore, the objective model in Listing 5.2 serves as input to the objective manager while the focus will be on the fulfillment of objective rules with the operational context *time in [08:00, 17:59] AND location = urban*. The context model is equal to the one presented in Listing 4.1.

5.3.1. Effect Model Combination

According to Figure 5.3 the first step of the objective manager is the generation of a combined effect model. Table 5.4 depicts the result of combining the two function-specific effect models in Listing 5.3 with an optimistic view. On the one hand, the lines on the top and those on the left show the effect indicators as well as the KPI effects for the two SCV sets of MRO and MLB respectively. On the other hand, the cells in the middle show the resulting combined effects $\tilde{\varphi}$ for the combination of the SCV sets indicated by the row and column.

Table 5.4.: Optimistic effect model combination for MRO and MLB

SON Function			MRO	
	Effect Indicator		$\varepsilon_{s_1}^{\text{MRO}} = \{(\text{DCR}, \downarrow), (\text{HOSR}, \leftrightarrow), (\text{CL}, \leq 0.65)\}$	$\varepsilon_{s_2}^{\text{MRO}} = \{(\text{DCR}, \leftrightarrow), (\text{HOSR}, \geq 0.99), (\text{CL}, \leftrightarrow)\}$
		Effect	$\varphi_{s_1}^{\text{MRO}}(\text{DCR}) = [0]$ $\varphi_{s_1}^{\text{MRO}}(\text{HOSR}) = [0,1]$ $\varphi_{s_1}^{\text{MRO}}(\text{CL}) = [0, 0.65]$	$\varphi_{s_2}^{\text{MRO}}(\text{DCR}) = [0,1]$ $\varphi_{s_2}^{\text{MRO}}(\text{HOSR}) = [0.99,1]$ $\varphi_{s_2}^{\text{MRO}}(\text{CL}) = [0,1]$
MLB	$\varepsilon_{s_1}^{\text{MLB}} = \{(\text{DCR}, \leftrightarrow), (\text{HOSR}, \downarrow), (\text{CL}, \leq 0.6)\}$	$\varphi_{s_1}^{\text{MLB}}(\text{DCR}) = [0,1]$ $\varphi_{s_1}^{\text{MLB}}(\text{HOSR}) = [0]$ $\varphi_{s_1}^{\text{MLB}}(\text{CL}) = [0, 0.6]$	$\tilde{\varphi}_{\sigma_{11}}(\text{DCR}) = [0]$ $\tilde{\varphi}_{\sigma_{11}}(\text{HOSR}) = [0]$ $\tilde{\varphi}_{\sigma_{11}}(\text{CL}) = [0, 0.6]$	$\tilde{\varphi}_{\sigma_{12}}(\text{DCR}) = [0,1]$ $\tilde{\varphi}_{\sigma_{12}}(\text{HOSR}) = \varnothing$ $\tilde{\varphi}_{\sigma_{12}}(\text{CL}) = [0, 0.6]$
	$\varepsilon_{s_2}^{\text{MLB}} = \{(\text{DCR}, \leq 0.02), (\text{HOSR}, \leftrightarrow), (\text{CL}, \leq 0.5)\}$	$\varphi_{s_2}^{\text{MLB}}(\text{DCR}) = [0, 0.02]$ $\varphi_{s_2}^{\text{MLB}}(\text{HOSR}) = [0,1]$ $\varphi_{s_2}^{\text{MLB}}(\text{CL}) = [0, 0.5]$	$\tilde{\varphi}_{\sigma_{21}}(\text{DCR}) = [0]$ $\tilde{\varphi}_{\sigma_{21}}(\text{HOSR}) = [0,1]$ $\tilde{\varphi}_{\sigma_{21}}(\text{CL}) = [0, 0.5]$	$\tilde{\varphi}_{\sigma_{22}}(\text{DCR}) = [0, 0.02]$ $\tilde{\varphi}_{\sigma_{22}}(\text{HOSR}) = [0.99,1]$ $\tilde{\varphi}_{\sigma_{22}}(\text{CL}) = [0, 0.5]$

As can be seen, s_1^{MLB} optimizes HOSR in a way that its value will be minimized whereas s_2^{MRO} optimizes against s_1^{MLB} since its usage would lead to a value ≥ 0.99 for HOSR. According to Equation 5.15, this is denoted as a conflict since the intersection of $\varphi_{s_1}^{\text{MLB}}(\text{HOSR}) = [0]$ and $\varphi_{s_2}^{\text{MRO}}(\text{HOSR}) = [0.99, 1]$ results in an empty

set. Hence, the set of conflict-free SCV set combinations is $\Sigma^{\Gamma} = \{\sigma_{11}, \sigma_{21}, \sigma_{22}\}$. These results can be directly translated into a combined effect model matching the general form of an effect model (cf. Listing 3.3) which is shown in Listing 5.4.

```
1 {
    IF MLB_1 AND MRO_1 THEN DCR = 0 AND HOSR = 0 AND CL ≤ 0.6
3   IF MLB_2 AND MRO_1 THEN DCR = 0 AND HOSR ≥ 0 AND CL ≤ 0.5
    IF MLB_2 AND MRO_2 THEN DCR ≤ 0.02 AND HOSR ≥ 0.99 AND CL ≤ 0.5
5 }
```

Listing 5.4: Combined effect model in ODSM, optimistic view

```
1 {
    IF MLB_1 AND MRO_1 THEN DCR ≤ 1 AND HOSR = 0 AND CL ≤ 0.65
3   IF MLB_2 AND MRO_1 THEN DCR ≤ 0.02 AND HOSR ≥ 0 AND CL ≤ 0.65
    IF MLB_2 AND MRO_2 THEN DCR ≤ 0.02 AND HOSR ≥ 0 AND CL ≤ 0.5
5 }
```

Listing 5.5: Combined effect model in ODSM, pessimistic view

```
1 {
    IF MLB_1 AND MRO_1 THEN DCR = 0 AND HOSR ≥ 0 AND CL ≤ 0.6
3   IF MLB_2 AND MRO_1 THEN DCR = 0 AND HOSR ≥ 0 AND CL ≤ 0.5
    IF MLB_2 AND MRO_2 THEN DCR ≤ 1 AND HOSR ≥ 0.99 AND CL ≤ 0.5
5 }
```

Listing 5.6: Combined effect model in ODSM, function-specific view

One can easily do the combination with a pessimistic or a function-specific view. The resulting combined effect models for these views are shown in Listing 5.5 and Listing 5.6 respectively. For the function-specific view it is assumed that the main driver for DCR and HOSR is the MRO function, while MLB has the highest impact on the CL. For both types of combinations the combined set σ_{12} also results in a conflict.

5.3.2. Generation of KPI Target State Space

In the second step of the algorithm for the derivation of an SCV set policy, the KPI target state space is generated in order to define which objective rules apply for which region in the context state space. Due to the same context model as in the PBSM example, the partitioned context state space looks just as the one depicted in Figure 4.4. The KPI target state space (cf. Figure 5.4) contains six regions whereby only a closer look at $\tau_{(\text{time in [08:00, 17:59], location=urban})}$ is given in this example. For this region, three objectives, one per KPI, are applicable:

- $o_{\text{DCR}} = ([0, 0.02], 0.5)$
- $o_{\text{HOSR}} = ([0.95, 1], 0.2)$
- $o_{\text{CL}} = ([0, 0.5], 0.3)$

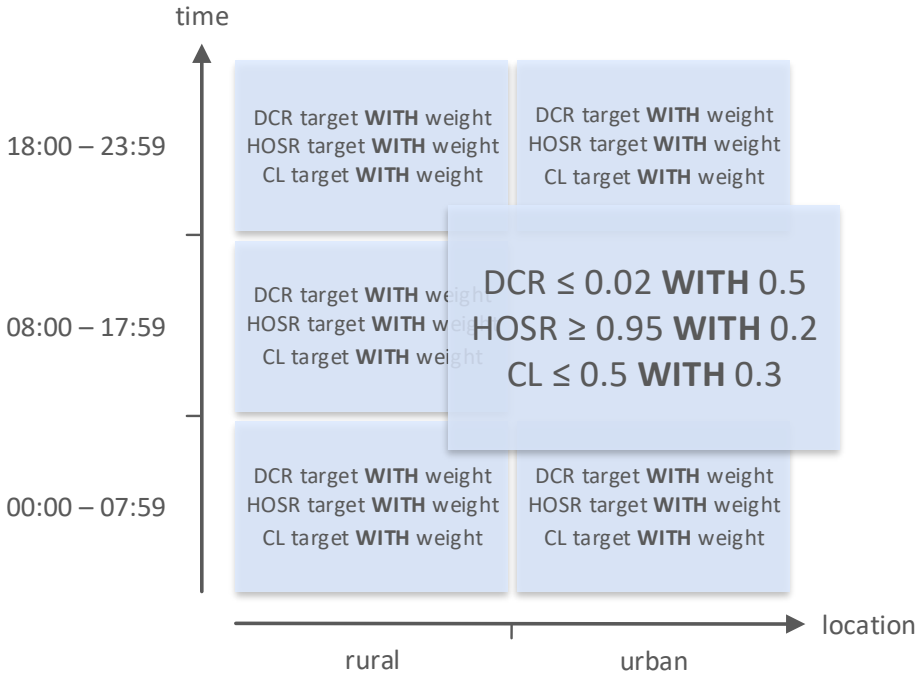


Figure 5.4.: KPI target state space with applicable KPI targets in region *(time in [08:00, 17:59], location = urban)*

5.3.3. Scoring of Combined SCV Sets

Table 5.5 visualizes the scoring of the conflict-free combined SCV sets in the SCV set selection step with an optimistic view. Thereby, the columns in the header depict the objectives for the three KPIs DCR, HOSR and CL in the region $\tau_{(\text{time in } [08{:}00,\ 17{:}59],\ \text{location=urban})}$ and the rows show the performed computations for each combined SCV set. In $\varphi_{\cdot}(k)$ and $\text{sat}(\cdot)$ the $\cdot$ refers to the respective combined SCV set, i.e., either σ_{11} or σ_{21} or σ_{22}. As can be seen, combined SCV set σ_{11} does only satisfy objective o_{DCR} and, thus, has an overall utility of 0.5. SCV set combination σ_{21} does satisfy o_{DCR} and o_{CL} which leads to an overall utility of 0.8. Since the SCV set σ_{22} satisfies all objectives, it gets with 1.0 the highest score and, consequently, is selected.

In contrast to the optimistic view, the results in the pessimistic view strongly differ in the calculated utilities for the particular combined SCV sets, see Table 5.6. This is due to wider ranges of the combined SCV sets having the result that Equation 5.22 is not fulfilled. However, even though σ_{22} does not fulfill all objectives in the pessimistic view, it is still the combined SCV set with the highest utility and hence, the one to be selected for this region.

In the function-specific view, not only the utilities are varying from the previous views, but also the resulting combined SCV set. Similar to the pessimistic view,

none of the σs can fulfill all KPI targets. However, the one with the highest utility is σ_{21} since it fulfills the objectives o_{DCR} and o_{CL}.

Table 5.5.: Scoring of the exemplary combined SCV sets with an optimistic view

	$o_{\text{DCR}} = ([0, 0.02], 0.5)$			$o_{\text{HOSR}} = ([0.95, 1], 0.2)$			$o_{\text{CL}} = ([0, 0.5], 0.3)$			$U(\cdot)$
	$\tilde{\varphi}_\cdot(\text{DCR})$	$\text{sat}(\cdot)$	w_{DCR}	$\tilde{\varphi}_\cdot(\text{HOSR})$	$\text{sat}(\cdot)$	w_{HOSR}	$\tilde{\varphi}_\cdot(\text{CL})$	$\text{sat}(\cdot)$	w_{CL}	
σ_{11}	$[0]$	1	0.5	$[0]$	0	0.2	$[0, 0.6]$	0	0.3	0.5
σ_{21}	$[0]$	1		$[0, 1]$	0		$[0, 0.5]$	1		0.8
σ_{22}	$[0, 0.02]$	1		$[0.99, 1]$	1		$[0, 0.5]$	1		1.0

Table 5.6.: Scoring of the exemplary combined SCV sets with a pessimistic view

	$o_{\text{DCR}} = ([0, 0.02], 0.5)$			$o_{\text{HOSR}} = ([0.95, 1], 0.2)$			$o_{\text{CL}} = ([0, 0.5], 0.3)$			$U(\cdot)$
	$\tilde{\varphi}_\cdot(\text{DCR})$	$\text{sat}(\cdot)$	w_{DCR}	$\tilde{\varphi}_\cdot(\text{HOSR})$	$\text{sat}(\cdot)$	w_{HOSR}	$\tilde{\varphi}_\cdot(\text{CL})$	$\text{sat}(\cdot)$	w_{CL}	
σ_{11}	$[0, 1]$	0	0.5	$[0]$	0	0.2	$[0, 0.65]$	0	0.3	0
σ_{21}	$[0, 0.02]$	1		$[0, 1]$	0		$[0, 0.65]$	0		0.5
σ_{22}	$[0, 0.02]$	1		$[0, 1]$	0		$[0, 0.5]$	1		0.8

Table 5.7.: Scoring of the exemplary combined SCV sets with a function-specific view

	$o_{\text{DCR}} = ([0, 0.02], 0.5)$			$o_{\text{HOSR}} = ([0.95, 1], 0.2)$			$o_{\text{CL}} = ([0, 0.5], 0.3)$			$U(\cdot)$
	$\tilde{\varphi}_\cdot(\text{DCR})$	$\text{sat}(\cdot)$	w_{DCR}	$\tilde{\varphi}_\cdot(\text{HOSR})$	$\text{sat}(\cdot)$	w_{HOSR}	$\tilde{\varphi}_\cdot(\text{CL})$	$\text{sat}(\cdot)$	w_{CL}	
σ_{11}	$[0]$	1	0.5	$[0, 1]$	0	0.2	$[0, 0.6]$	0	0.3	0.5
σ_{21}	$[0]$	1		$[0, 1]$	0		$[0, 0.5]$	1		0.8
σ_{22}	$[0, 1]$	0		$[0.99, 1]$	1		$[0, 0.5]$	1		0.5

5.3.4. Policy Generation and SCV Set Deployment

An excerpt of the resulting policy comprising the scoring results with an optimistic view is depicted in Listing 5.7. For each cell which is located in an urban area, the MLB function is configured with an SCV set *MLB_2* and the MRO function with an SCV set *MRO_2* during peak traffic hours, i.e., between 8 o'clock am and 5:59 pm.

```
1 {
    ...
3   IF time in [08:00, 17:59] AND location = urban
      THEN MLB = MLB_2
5   IF time in [08:00, 17:59] AND location = urban
      THEN MRO = MRO_2
7   ...
  }
```

Listing 5.7: Excerpt of SCV set policy based on the scoring of the combined SCV sets

For the pessimistic and function-specific view, the policy generation and the selection of appropriate SCV sets works analogously.

5.4. Related Work

Most of the work related to the presented approach has already been presented in Section 3.4 and Section 4.4. Several projects working on SON management were tackled while none of them describe concrete ideas and a detailed solution for the realization of a SON management system. Other approaches describe ways to refine high-level business policies into technical policies, however, this refinement process is mainly done manually or done on a more abstract level.

In this chapter, an approach has been presented that combined effect models of different SON functions into a combines effect model, thereby enabling a design-time coordination of SON functions by identifying SCV sets that have a conflictive influence on each other. Several approaches deal with the coordination of SON functions. The authors of [Iac+14c], [Iac+14b] and [Iac+16] present a reinforcement learning framework that uses function approximation to coordinate the actions of two distributed SON functions MLB and MRO. While this SON coordinator tries to resolve such conflicts after their appearance, i.e., at runtime, SON management as described in this chapter tries to find non-conflicting SCV sets beforehand that do not influence each other negatively. In contrast, the authors of [Ban13], [Ban+11] and [BSR11] have already identified the existence of a variety of conflict types. Besides configuration conflicts which are typical runtime conflicts as described above, they have also identified design-time conflicts called characteristic conflicts in their work. However, these conflicts appear when two or more SON functions touch the same NCP which is a different type of conflict as those defined in this chapter. Also, the authors of [CAA13] describe an approach to coordinate autonomic functionalities, i.e., SON functions, while also focusing on conflicts after their appearance. In [Tsa+13a], a prototypical approach is proposed to implement a SON function coordinator in a unified management framework.

The conflicts described in this chapter are based on manufacturer effect models and hence, describe a new reason for an undesired network behavior. In order to guarantee the best results, both the pre-coordination in this management approach as well as the post-coordination within the mentioned related work, should run in parallel. However, the development of a SON coordinator is not in the focus of this thesis.

The approach that is closest to the presented solution is [Fre16]. The author presents a SON management approach that is driven by more sophisticated objectives. While objectives in the ODSM approach only have a binary degree of fulfillment, the author describes the degree of fulfillment by means of fuzzy ranges (cf. [FSB13]) allowing for a more detailed objective evaluation. Since the focus in this thesis is mainly on the generation of a more sophisticated context and effect model, these approaches can be easily combined. A difference to this work lies in the fact that the author does the calculation of optimal SCV sets at runtime. The advantages and shortcomings of both approaches have already been discussed in Section 3.2.4.

6

Adaptive SON Management

The third SON management approach presented in this thesis is ASM. Compared to the PBSM and ODSM approaches, ASM is the first approach to use feedback coming from the real network and using this feedback to adapt existing input models, hence it is called adaptive. ASM is based on the concepts and findings of ODSM and enhances its input models, thereby overcoming shortcomings of ODSM. The focus of this chapter is on an extension to the context model and the effect model. Furthermore, the usage of two new sub-models accounts for a completely new methodology of the component which brings all these models together, the objective manager. Also the policy system, i.e., the structure of the SCV set policy, is affected by these changes. After describing the models and methodology in a theoretical way, the process of finding optimal SCV sets in ASM is exemplified and related work is presented.

6.1. Motivation

The policy-based and objective-driven approaches entail some shortcomings. The design-time computation of the SCV set policy may be computationally expensive due to an exponential growth of the considered context state space. This is due to the fact that within the objective manager, the SCV sets for each SON function in each possible operational context needs to be calculated any time a reconfiguration of the network is triggered. The runtime option described in Section 3.2.4 avoids this problem by only determining optimal SON function SCV sets for the one cell or the set of cells that require a reconfiguration. However, this option has the disadvantage that the management of SON functions gets more nebulous for MNOs and thereby disagrees with one of the main objectives of this thesis: Gaining trust in SON management. Hence, a method needs to be found to significantly reduce the exponentially growing context state space to a manageable set of states.

A second shortcoming that both previous approaches have in common, is, that the manufacturer-provided effect model predicts KPI effects under a certain SCV set uniformly for all cells in the network without considering the cell's operational context. Consequently, a possible extension of the effect model is to make

the SCV sets context-dependent like the objective model. This allows for expressing different KPI effects for different combined SCV sets and hence, a more precise forecasting of network behavior. However, this increases modeling complexity because it has to be ensured that the rules of the effect model are conflict-free in order to guarantee a conflict-free SCV set policy. If the SCV set policy is not conflict-free, the selection of SCV sets is ambiguous which could lead to errors when reconfiguring a SON function.

In the PBSM approach, the effect model could only describe the minimization, maximization or neutrality of of a KPI value, but not that a specific SCV set keeps the KPI value within some range, e.g., DCR lower than 0.5%. In the ODSM approach, the effect model is extended in a way that allows the definition of value ranges for the KPI targets. However, it is the goal to make the effect model more precise and hence, to define concrete KPI effects instead of just giving a vague indication about the impact of certain SCV sets in the network.

Finally, the manufacturer-provided effect model is only based on simulations done in the manufacturer's network environment and hence, the predicted effects in the real network could be inaccurate due to the different environments. It is expected that the predictions do not hold true in a real network environment and therefore, feedback in terms of KPI measurements has to be taken into account. Besides the adaptation of the initially provided effect model, such an effect sub-model marks an important step forward towards a cognitive management approach. Such a system would not only rely on simulated results, but instead it does a simple learning by permanently enriching, analyzing and thereby improving information required for the calculation of optimal SCV sets. Hence, three objectives are addressed in this chapter, depicted in Figure 6.1.

Figure 6.1.: Objectives of ODSM

By defining two new sub-models, one for the context model and one for the effect model, a step forward to fulfilling the subsequent objective is made:

Objective 2 *Automatically generate realistic and complete input models which are continuously updated according to the current network state such that they optimally support the SON management system in finding the best possible network configuration.*

Also, having more trustful data within the effect model and making the context model more manageable helps to fulfill Objective 4 which is defined as:

Objective 4 *Develop a fully automated SON management system which allows MNOs to comprehend, restrict and influence automated actions at any time.*

6.2. Approach

Similar to previous approaches ASM also requires three different types of initial input models: the context model and the objective model, both provided by the MNO, and the effect model which is provided by the SON function manufacturer. With these input models, the central SON management entity, denominated as objective manager, generates an SCV set policy as output, i.e., a list of rules giving an indication how to configure the SON functions to best fulfill operator objectives. A policy system decides which policy rule has to be executed and hence which SCV set is deployed to the network. To this point the approach seems similar to the PBSM and ODSM approaches. However, the input models strongly differ in the data they provide.

The context model is separated into two sub-models: One that describes context attributes and possible values for these attributes, the so-called context attributes model. Another sub-model is based on the definition of context attributes and their values, the so-called manual context classes model which needs to be provided by the MNO. Within this model, the operator defines partitions of the context state space in which he or she assumes that cells behave similar. Technical operator objectives defined in the objective model depend on these context classes, thereby significantly reducing the number of objectives that have to be specified by the MNO. KPI effects in the effect model are also dependent on these classes, leading to more accurate performance prediction. Note that this applies for all types of sub-models of the effect-model, i.e., the initially provided effect model, the combined effect model and the real network effect model. The latter represents a model that is derived by collecting data about how different SCV sets effect KPIs in the real network, hence leading to a more realistic description of the SON functions' behavior. This model is generated by the objective manager and continuously adapted (hence adaptive SON management) and improved during runtime by using real network measurements in terms of FM, PM and CM data. An overview of the general ASM approach is depicted in Figure 6.2.

By starting with a simulated effect model, i.e., the combined effect model, and, when a sufficient number of measurements has been collected, replacing it with a model that describes the effects in the real network, a more accurate SCV set policy can be generated that can yield a better performance in achieving operator objectives.

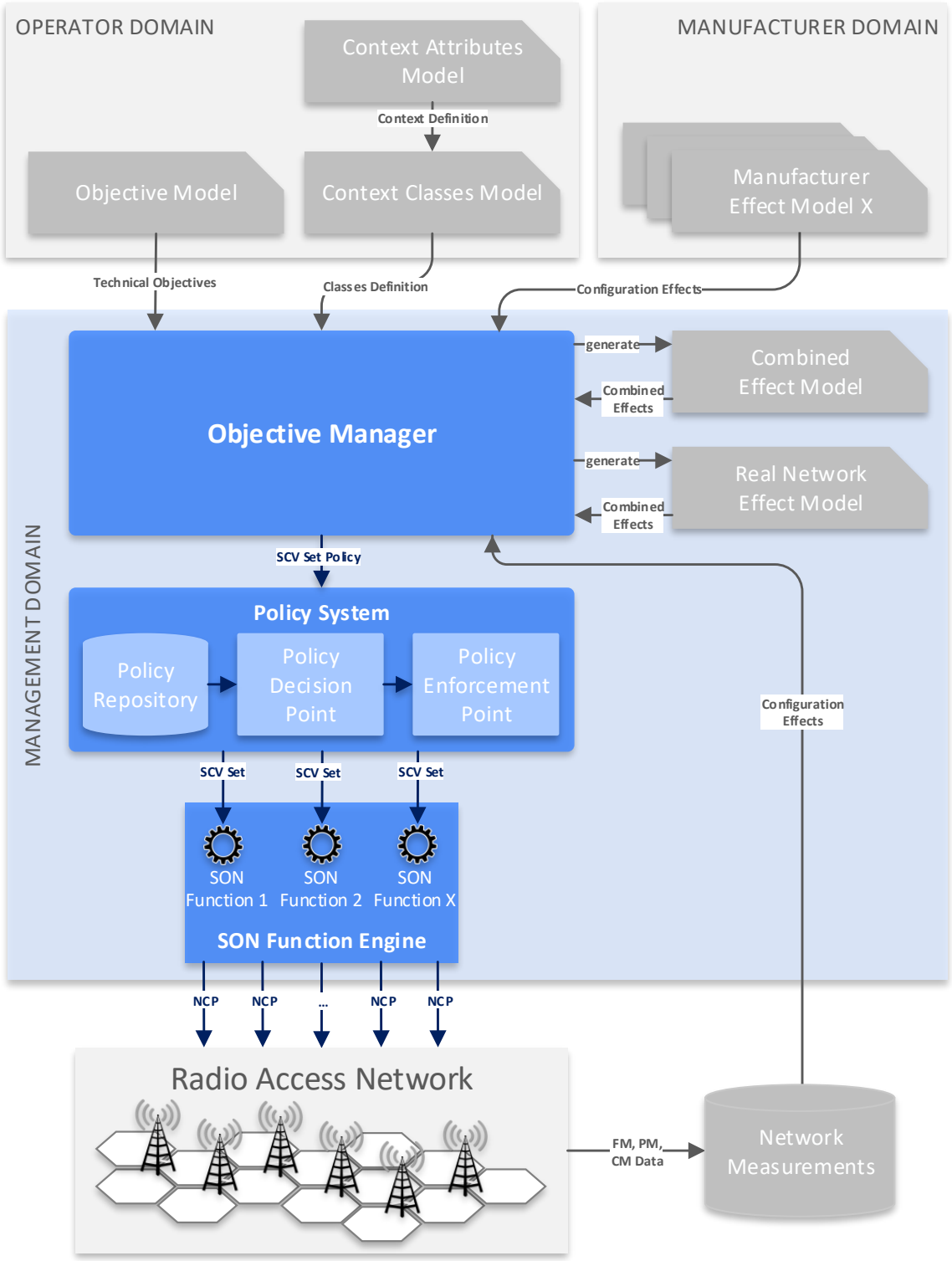


Figure 6.2.: Overview of ASM

Note that for the sequel of this chapter, the formalizations of the SON-enabled mobile network, i.e., the definition of SON functions (cf. Equation 5.1), the SCV sets per SON function (cf. Equation 5.2) and the KPIs (cf. Equation 5.3) as done

in the PBSM and ODSM approach, also apply for ASM.

6.2.1. Context Model

In general, the context model in ASM is provided through the MNO and describes the network environment through cell attributes such as the geographical area, the data traffic situation or the predominant user mobility type. By the values of these attributes the actual context of a cell can be represented. Due to different cell contexts the KPI target values defined by the MNO might be different, see Figure 6.3.

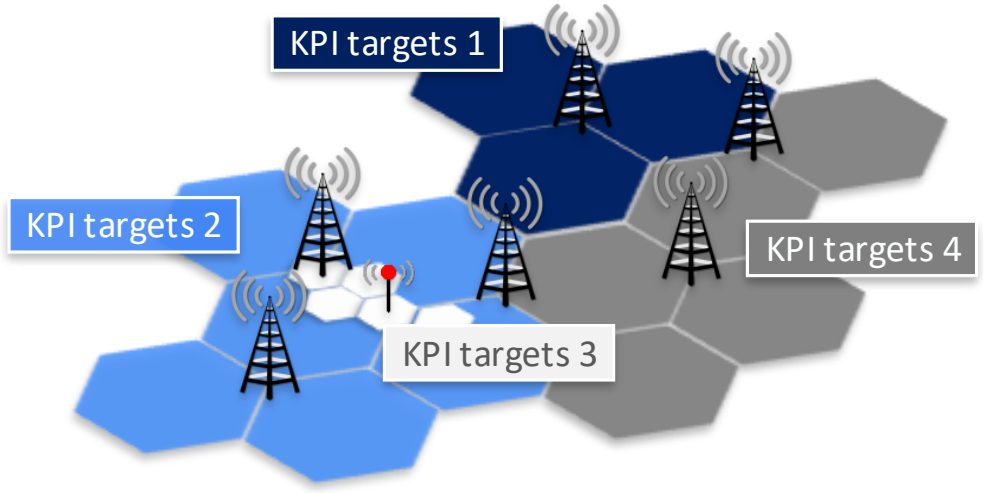


Figure 6.3.: KPI targets for different cells in the network

Manual Context Classes Model
In previous approaches, the MNO had to specify operator objectives for arbitrary combinations of these context attributes, building the condition part of the technical objectives and potentially leading to a big number of objective rules due to the exponential growth of the context state space as described in Section 6.1. The given set of context attributes spans a context state space consisting of all possible context attribute combinations that may exist. For each context attribute an additional dimension is required which significantly increases the number of potential context parameter combinations. In order to reduce the size of the context state space, it shall be partitioned into disjoint classes where each class is a combination of context attributes and respective values that represents a certain cell type in the network. The specific classes are defined by the operator individually depending on which types of cells should be considered jointly.

The definition of context classes, i.e., cells with similar context, reduces the complexity for the MNO because it allows to formulate KPI target values for the

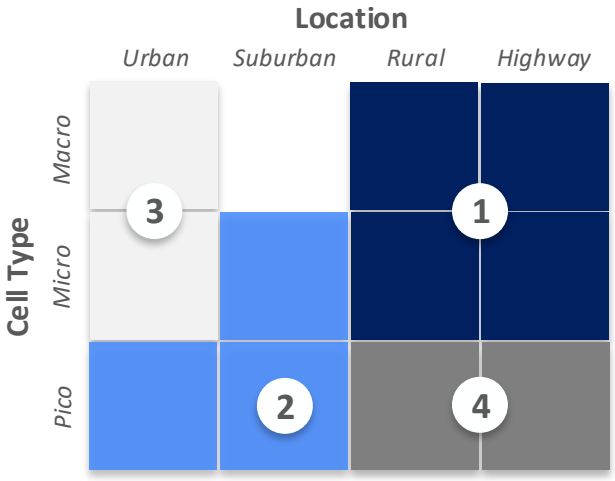


Figure 6.4.: Exemplary partitioning of the context state space into four disjoint context classes

relative small number of context classes instead of defining values for each attribute combination. Note that not all context attribute combinations need to be covered by the context classes. A combination that is not considered within the classes definition means that no cell exists in the network with this operational context. This is important since it is assumed that not all combinations of context attributes are relevant for an operator and hence, the manual effort for defining context classes is limited.

For instance, manual context class *3* in Figure 6.4 is defined as:

- location = {Urban}
- cell_type = {Micro, Macro}

Taking the above mentioned definition, context classes can be expressed by means of condition-action rules where the IF part specifies the combination of context attributes and their values, and the THEN part defines the name of the context class:

```
IF context parameter combination THEN context class
```

Listing 6.1: Classes definition rule in ASM

Bringing together the general form of context classes rules and the partitioned state space in Figure 6.4, an exemplary context classes model could look as depicted in Listing 6.2.

On a more formal level, the manual context classes model CM defines a mapping from the context state space R to context classes C with c_i being a single context class.

$$\mathrm{CM} : R \mapsto C \tag{6.1}$$

```
{
  IF (location = rural OR location = highway) AND (cell_type = micro
    OR cell_type = macro) THEN CLASS_1
  IF (location = urban AND cell_type = pico) OR (location = suburban
    AND (cell_type = micro OR cell_type = pico)) THEN CLASS_2
  IF location = urban AND (cell_type = micro OR cell_type = macro)
    THEN CLASS_3
  IF (location = rural OR location = highway) AND cell_type = pico
    THEN CLASS_4
}
```

Listing 6.2: Exemplary context classes model in ASM

$$C = \{c_1, c_2, ..., c_{|C|}\} \tag{6.2}$$

Each c_i consists of an arbitrary number of context state space regions $\rho_j \in R$ as defined in Equation 4.16 and consequently:

$$c_i = \{\rho_j | 1 \leq j \leq |R| \wedge \forall c_k \in C, i \neq k : \rho_j \notin c_k\} \tag{6.3}$$

Note that a region can only belong to one context class as defined above, since otherwise the selection of appropriate SCV sets in the end would lead to a conflict due to ambiguous policy rules.

6.2.2. Objective Model

With the objective model, the MNO can express requirements about the network performance through KPI target values. Equal to the ODSM approach, weights represent the importance of a KPI target and allow a trade-off between them in case they cannot be satisfied simultaneously. In contrast to the previous approach, KPI targets are defined per manual context class. Combining this information, a technical objective is defined as condition-action rule with the following general form:

```
IF context class THEN KPI targets WITH weight
```

Listing 6.3: Technical objective rule in ASM

Following this definition, the objective model OM can be seen as a function mapping context classes to tuples of objectives for all $|K|$ KPIs, i.e:

$$\text{OM} : C \mapsto O_{k_1} \times O_{k_2} \times \cdots \times O_{k_{|K|}} \tag{6.4}$$

with O_{k_i} being a set of objectives $o_{k_i} \in O_{k_i}$ for a KPI $k_i \in K$.

An exemplary objective model with context class-dependent KPI targets for three KPIs DCR, HOSR and CL is shown in Listing 6.4. As can be seen, the objective model is much more clearly arranged compared to previous objective models due to a reduced number of technical objectives rules, i.e., one per KPI and context class.

```
{
  IF CLASS_1 THEN ...
  IF CLASS_2 THEN ...
  IF CLASS_3 THEN DCR ≤ 0.02 WITH 0.5
  IF CLASS_3 THEN HOSR ≥ 0.95 WITH 0.2
  IF CLASS_3 THEN CL ≤ 0.5 WITH 0.3
  IF CLASS_4 THEN ...
}
```

Listing 6.4: Exemplary objective model in ASM

6.2.3. Effect Model

Information about the SON functions is provided through the effect model which shall describe the behavior of the SON functions regarding the coherence between a dedicated SCV set and the corresponding impact on the KPIs. On the one hand, an effect model may be provided from the manufacturer of the SON function together with the black box SON algorithm itself. Presumably, this manufacturer-provided model is generated using simulation tools, with the shortcoming that a simulator-generated model is unlikely to be adapted to the actual network and SON environment the SON function is finally implemented in. Hence, this simulator-generated model may not lead to the intended network performance and a method must be found to take the real network into account. Therefore, an effect model in ASM is split into two sub-models: A combined effect model generated by bringing together initially provided manufacturer effect models and a real network effect model which is derived during operation of the network. The latter is built based on real measurements from the MNO's operational network. With this approach, a new dedicated model is created based on collected and analyzed network measurements. These KPI measurements are put into relation with the respective SCV set and the current operational context, i.e., the context class. The creation of such an adaptive effect model and the possibilities for combining the real network effect model with the combined effect model are further described in the course of this chapter.

6.2.3.1. Manufacturer Effect Model

In contrast to the definition of the initial effect model as described in the ODSM approach, one major difference in ASM is that this model is context-dependent. Different SCV sets may have an unequal impact on the KPIs for cells in various operational contexts. Hence, SON function manufacturers need to provide an effect model that describes KPI effects of possible SCV sets for manual context classes. The context classes model has to be made available to manufacturers by the MNOs such that KPI effect predictions can be calculated based on these context classes. Since it is assumed that cells in the same context class have a similar behavior in terms of network KPI, such a context-dependent effect model describes effect predictions for cells in a more accurate way.

Furthermore, while an effect prediction in ODSM is, amongst others, defined as the minimization, maximization or neutrality of a KPI, in ASM an effect prediction only defines a certain value range the KPI will be in when deploying the respective SCV set. This also increases accuracy of the effect model which is necessary for the calculation of optimal SCV sets in the ASM objective manager. These two points mentioned above lead to the following general form of a manufacturer effect model in ASM:

```
IF context class AND SCV set THEN KPI effects
```

Listing 6.5: Effect model rule in ASM

Hence, both the name of the context class and an SCV set of a specific SON function f, make up a condition $b_i^f \in B$ in the effect model for f:

$$B^f = \left\{ b_1^f, b_2^f, ..., b_{|B^f|}^f \right\} \tag{6.5}$$

$$b^f = (c, s^f) \tag{6.6}$$

The sub-model for one SON function f defines a mapping between a set of possible conditions B^f and their effect indicators on the KPI E^f.

$$\mathrm{FM}^f : B^f \mapsto E^f \tag{6.7}$$

with E^f referring to the set of KPI indicators as defined in Equation 5.7 and the particular effect indicators $\varepsilon^f \in E^f$ being defined according to Equation 5.9. One major difference to previous approaches is the definition of the formalized effect $\varphi_{b^f}(k)$ of a KPI k for a condition b^f within a SON function f which is confined to value ranges, i.e:

$$\varphi_{b^f}(k) = \begin{cases} [\min(\mathrm{Dom}(k)), x] & \text{if } (k, \leq x) \in \mathrm{FM}^f(s) \\ [x, \max(\mathrm{Dom}(k))] & \text{if } (k, \geq x) \in \mathrm{FM}^f(s) \end{cases} \tag{6.8}$$

6.2.3.2. Combined Effect Model

The generation of a combined effect model in principal works similar to the methodology described in Section 5.2.3.1. The possibilities of combining the value ranges of KPIs are significantly reduced due to the lack of minimization, maximization and neutrality values. The only difference is that the initial effect models are context-dependent which does not effect the general way of combining manufacturer effect sub-models. However, the combination has to be done for each manual context class individually in order to not mix up the effect indications.

The result of combining context-dependent effect sub-models is a set of conflict-free SCV set combinations $\Sigma_c^{\Gamma_c}$ for each context class $c \in C$ and Γ_c being the set of conflicting SCV set combinations for that context class. Note that the set of conflicting SCV set combinations may be different for different context classes.

$$\Sigma_c^{\Gamma_c} = \Sigma \setminus \Gamma_c \tag{6.9}$$

Consequently, the conditions of the generated combined effect model can be described as tuples consisting of the context class c and the non-conflicting combined SCV sets $\sigma \in \Sigma_c^{\Gamma_c}$. The set of conditions is defined as Ψ_c for each context class c and Φ^c as the set of combined KPI effect indicators $\tilde{\varphi}_\sigma^c$.

$$\Psi^c = \left\{(c, \sigma) | \sigma \in \Sigma_c^{\Gamma_c}, c \in C\right\} \tag{6.10}$$

$$\Phi^c = \left\{\tilde{\varphi}_\sigma^c(\cdot) | \sigma \in \Sigma_c^{\Gamma_c}, c \in C\right\} \tag{6.11}$$

Combining this and following the general form of an effect model rule, the combined effect model can be described as a mapping function from conflict-free combined SCV sets to combined KPI effect indicators, i.e.:

$$\mathrm{EM}_{\mathrm{combined}} : \Psi \mapsto \Phi \tag{6.12}$$

6.2.3.3. Real Network Effect Model

One of the major shortcomings of PBSM and ODSM is that the effect models in these approaches only come from the manufacturers of a SON function and therefore, predictions in there are inaccurate in the real network with the utmost probability. In order to overcome this problem, a new type of effect model is introduced which is called real network effect model. As the name suggests, the model is created by taking real network measurements into account and processing these data to a more accurate model reflecting the actual effects of combined SCV sets in the network.

The processing of KPIs is divided into two steps: the selection of suitable KPI measurements and the generation of SCV set statistics, i.e., the real network effect model.

KPI Measurements Selection
A KPI measurement $\mu_\sigma^c(\cdot)$ with a combined SCV set combination σ and a context class c is described as a set of several KPI values μ_k^c, one for each of the $|K|$ KPIs, see Equation 6.13.

$$\mu_\sigma^c(\cdot) = \left\{\mu_{k_1}^c, \mu_{k_2}^c, ..., \mu_{k_{|K|}}^c\right\} \tag{6.13}$$

Consequently, the set of all measurements MM contains all raw measurements for all context classes and all combined SCV sets that have been applied while measurements were taken. The set of applied SCV set combinations $\Sigma_{\text{app}}^{\Gamma_c}$ is a subset of the set of conflict-free set combinations Σ^{Γ_c} for the respective manual context class $c \in C$, i.e. $\Sigma_{\text{app}}^{\Gamma_c} \subseteq \Sigma^{\Gamma_c}$.

$$\text{MM} = \{\mu_\sigma^c(\cdot) | \sigma \in \Sigma_{\text{app}}^{\Gamma_c}, c \in C\} \tag{6.14}$$

Note that MM must not necessarily contain measurements of all combined SCV sets. This depends on which SCV set combinations were already deployed in the network.

The bulk of available KPI measurements needs to be categorized per source context class and, within these context classes, per SCV set combination. The reason for this categorization is that the SCV set combination is performed per context class. Hence, for determining the effect of an SCV set combination on the behavior of a SON function, only those KPI measurements are relevant for a context class that stem from exactly this context class. The KPI selection has to be performed separately for each combined SCV set within a context class, as they are not comparable. This process is depicted in Figure 6.5.

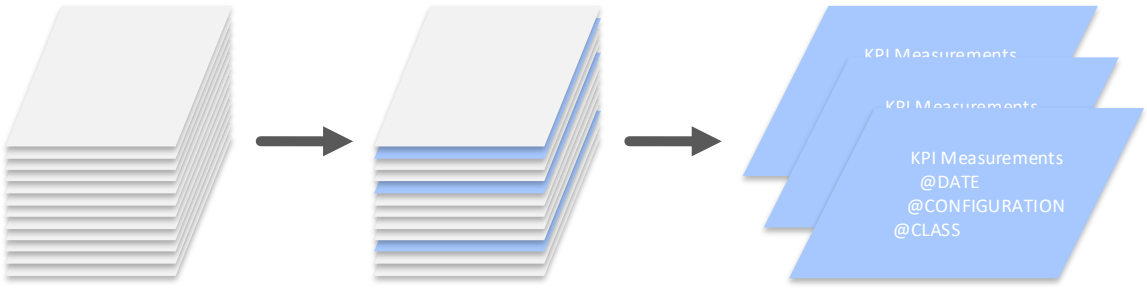


Figure 6.5.: Selection of appropriate measurements for each SCV set in a context class

Hence, MM can be also represented as a set of measurement subsets M_σ^c, categorized per context class c and SCV set combination σ.

$$M_\sigma^c = \{\mu_\sigma^c(\cdot)_1, \mu_\sigma^c(\cdot)_2, ..., \mu_\sigma^c(\cdot)_n\} \tag{6.15}$$

with $M_\sigma^c \subseteq \text{MM}$.

As can be seen in Figure 6.5, KPI measurements are also tagged with the date at which they were taken. This is relevant for the next step, the generation of statistics out of these measurements.

Generation of Measurement Statistics
In this step, a statistical processing is performed on the sub-categorized KPI measurement sets M_σ^c, for which a simple arithmetic mean calculation is used in the context of this work, as shown in Equation 6.16.

$$\tilde{\mu}_\sigma^c(k) = \frac{\sum_{i=1}^{n} \mu_\sigma^c(\cdot)_i(k)}{n} \tag{6.16}$$

with $\mu_\sigma^c(\cdot)_k \in M_\sigma^c$. Consequently, the combined effect on all KPIs $\tilde{\mu}_\sigma^c(\cdot)$ is defined as:

$$\tilde{\mu}_\sigma^c(\cdot) = \left\{ \tilde{\mu}_\sigma^c(k_1), \tilde{\mu}_\sigma^c(k_2), ..., \tilde{\mu}_\sigma^c(k_{|K|}) \right\} \tag{6.17}$$

with $\tilde{\mu}_\sigma^c(\cdot)$ being in the set of calculated measurement effects $\tilde{M}^c$, i.e., $\tilde{\mu}_\sigma^c(\cdot) \in \tilde{M}^c$. Other options for the statistical processing could be, for example, standard deviation methods or different types of averaging such as the weighted moving average which is presented in the further course of this chapter.

The measurement statistics then finally represent the mean over all measurements that come from the same context class and the same combined SCV sets. Note that this processing needs to be done for each context class and each combined SCV set individually. Following the formalization done in Section 6.2.3.2, the real network measurement effect model $\mathrm{EM}_{\mathrm{real}}$ can be described as a function mapping all applied SCV set combinations within a certain context class $\tilde{\Psi}$ to the calculated KPI effects $\tilde{M}$, i.e.:

$$\mathrm{EM}_{\mathrm{real}} : \tilde{\Psi} \mapsto \tilde{M} \tag{6.18}$$

with $\tilde{\Psi}$ being the set of $\tilde{\Psi}^c$s and $\tilde{M}$ being the set of $\tilde{M}^c$s for all context classes $c \in C$. The set of conditions $\tilde{\Psi}^c$ and the set of aggregated KPI measurement, i.e., the effect indicators, are thereby defined as:

$$\tilde{\Psi}^c = \left\{ (c, \sigma) | \sigma \in \Sigma_{\mathrm{app}}^{\Gamma_c}, c \in C \right\} \tag{6.19}$$

$$\tilde{M}^c = \left\{ \tilde{\mu}_\sigma^c(\cdot) | \sigma \in \Sigma_{\mathrm{app}}^{\Gamma_c}, c \in C \right\} \tag{6.20}$$

Note that for the real network effect model it is necessary to first have a sufficient amount of KPI measurements available. The availability thereby depends, for example, on the activity of the SON functions, i.e., how often a change to the network configuration is performed through the SON algorithm (which may be, e.g., once per second, per hour, per day, etc.), but also on the statistical relevance, i.e., how much data for a certain SCV set within a certain context class is actually available.

An exemplary derivation of SCV set statistics for one manual context class is shown in Figure 6.6. Here, the KPI measurements all feature the same context class (CLASS_1). The outcome are statistics for different SCV sets in the shape of rules for the real network effect model.

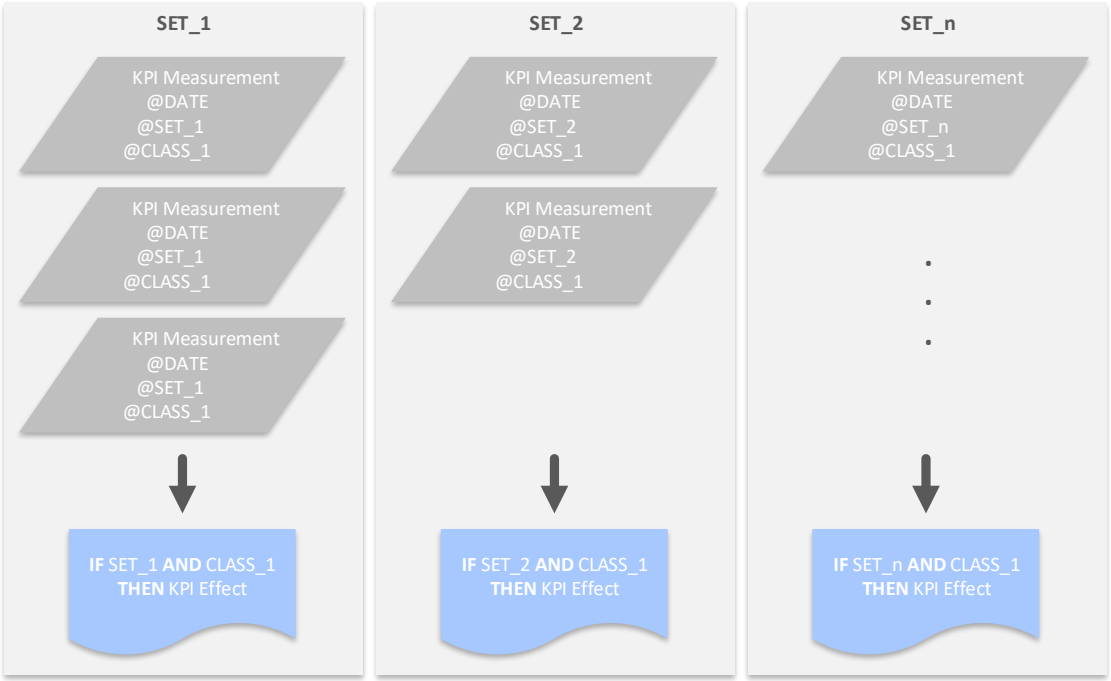


Figure 6.6.: Generation of a statistics for the real network effect model

6.2.4. Methodology

In general, the methodology of the objective manager, i.e., the selection of appropriate SCV sets according to technical objectives, consists of two consecutive parts. The first part performs the KPI value processing where measurements from the network are acquired, selected, and statistically processed in relation to the SCV sets being active in the network. In the second part, namely, the calculation of combined SCV sets, these statistics, together with the combined manufacturer effect model, are filtered and evaluated according to their impact on the network performance in order to determine the best possible SCV for a set of dedicated KPI targets. While the derivation of the real network effect model has already been described in Section 6.2.3.3, the latter, i.e., the mapping process between objective model, context model and effect model is described in detail in the following. This process is depicted in Figure 6.7 and, compared to the PBSM and ODSM approaches, describes a much more complex yet more accurate methodology for the selection of optimal combined SCV sets.

The generation of the real network effect model is divided into two steps: The selection of relevant measurements per context class and the creation of statistics out of these sorted measurements, resulting in a new sub-model of the effect model. This model and the combined effect model serve as input to the calculation step which is subdivided into five consecutive parts, namely the *delta determination*, the *SCV set filtering*, the *KPI target vs. network performance comparison*, the *performance indication* and finally, the *selection of combined SCV sets* with the highest overall utility for each manual context class.

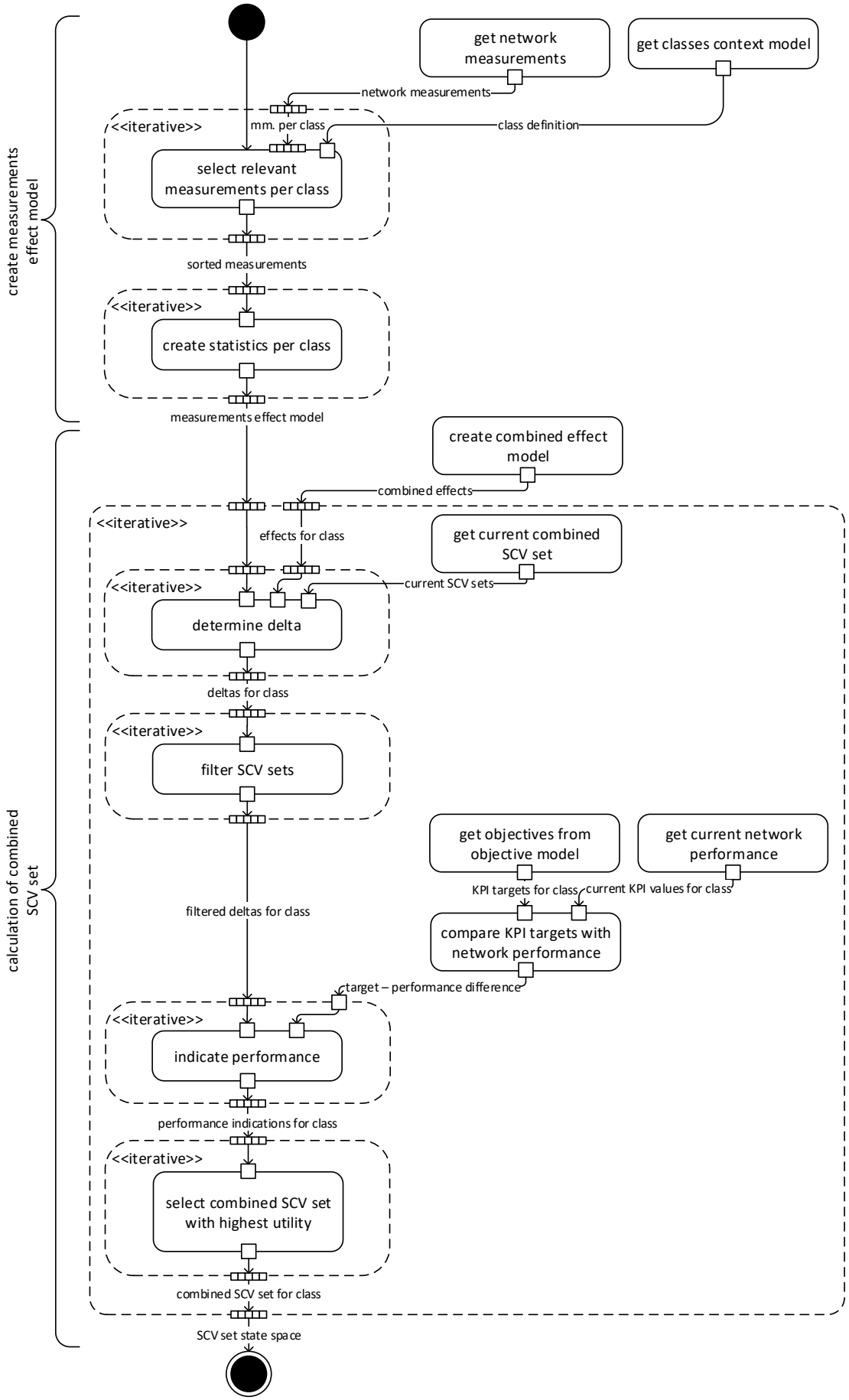


Figure 6.7.: SCV set policy derivation algorithm in ASM (noted in UML 2)

Note that, in the remainder of this chapter, the real network measurement model as well as the combined manufacturer effect model are often both together denoted as effect model. In so far, measurements in the context of the combined effect model denote data coming from the initial manufacturer-provided effect sub-models while they denote real network acquired KPI measurements in the context of real network effect model. However, the syntax of the content of both types of effect models is the same, describing the interrelation between context class-specific SCV set combinations and the corresponding impact on the KPI values using condition-action rules. Furthermore, the context attribute model and the manual context classes model both together are denoted as context model while only the context classes model is necessary for further calculations.

6.2.4.1. SCV Set Calculation

After deriving an effect model, the SCV set calculation is the second part towards deciding on the best possible combined SCV sets. This decision is made based on evaluating the KPI effect predictions in the effect model which result from the KPI processing and the combination of initial effect sub-models, and putting them into relation with the MNO-defined objective model and the context classes model. The outcome of this is an SCV set policy that indicates for each context class a combination of SCV sets that can be evaluated by the policy system. The SCV set calculation part is divided into five consecutive steps that are described in detail in the following. Note that this process needs to be performed for both sources of the effect model which can be easily done due to the same structure and meaning of the contained datasets. The process is exemplified with the real network effect model in the following, meaning that the formalizations refer to this sub-model. However, it works analogously for the combined effect model.

Delta Determination
In this first step, a delta between the expected network performance, i.e., the KPI effect prediction in the effect model for the currently active SCV set combination σ_{current} and all effects of other possible SCV set combinations σ_i in the effect model is calculated (cf. Equation 6.21).

$$\delta_c(\sigma_i, \sigma_{\text{current}}) = \tilde{\mu}^c_{\sigma_i}(\cdot) - \tilde{\mu}^c_{\sigma_{\text{current}}}(\cdot) \tag{6.21}$$

This is done for each context class individually, i.e., the expected performance of cells in the same context class is compared to only effect predictions in the SCV set statistics usable in the context of this class. $\delta_c(\sigma_i, \sigma_{\text{current}})$ thereby means the resulting delta and $\tilde{\mu}^c_{\sigma_i}(\cdot)$ means the predicted effect of an SCV set combination coming from the SCV set statistics. Note that $\tilde{\mu}^c_{\sigma}(\cdot)$ refers to a set of KPI values and this delta calculation is done for each KPI individually. The resulting delta as well as the $\tilde{\mu}^c_{\sigma}(\cdot)$s can be expressed by vectors indicating the difference for each KPI and the KPI values respectively.

For instance, when having three KPIs DCR, HOSR and CL (in exactly this order), the determination of delta could look as follows:

$$\delta_{\text{CLASS}_1}(\sigma_{12}, \sigma_{22}) = \begin{pmatrix} 0.05 \\ 0.95 \\ 0.63 \end{pmatrix} - \begin{pmatrix} 0.03 \\ 0.97 \\ 0.58 \end{pmatrix} = \begin{pmatrix} 0.02 \\ -0.02 \\ 0.05 \end{pmatrix} \tag{6.22}$$

with σ_{12} being the combined SCV set under investigation and σ_{22} being the currently active SCV set combination. With this delta, the MNO gets an impression about the impact on the network performance when changing from the current SCV set combination to another. In case of the presented example, this means that DCR would increase by 0.02, HOSR would decrease by 0.02 and the CL would increase by 0.05 when changing from the combined SCV set σ_{22} to σ_{12} for context class CLASS_1.

By calculating such a delta, the two sources of effect models get more comparable. It is assumed that measurements within each sub-model are consistent leading to the same level of aggressiveness for deltas from different sources even though the absolute values may be different. Aggressiveness here indicates the performance impact on certain KPIs when choosing a particular SCV set combination.

The result of this step is a set Δ of δs for each context class $c \in C$, i.e.,

$$\Delta_c = \left\{ \delta_c(\sigma_i, \sigma_{\text{current}}) | \sigma_i, \sigma_{\text{current}} \in \Sigma_c^{\Gamma_c},\ c \in C \right\} \tag{6.23}$$

and hence, $|\Delta_c| = |\Sigma_c^{\Gamma_c}|$ since each possible SCV set combination, which includes the currently active one, is compared to the expected performance of the currently active combined SCV set.

SCV Set Filtering

In this step, the list of possible SCV sets is filtered such that SCV sets that have a negative effect on all KPIs can be discarded. Thereby, one needs to distinguish between two different types of KPIs: On the one hand, KPIs where an MNO aims for the highest possible value (e.g., HOSR) and on the other hand, KPIs where the lowest possible value is targeted (e.g., DCR). In order to make these KPIs comparable, the δs have to be scaled first. That is, they have to be multiplied by a scaling vector indicating whether a KPI aims for the highest or lowest possible value. This vector needs to be defined by the MNO and provides a scaling value for each observed KPI, i.e., size(ς) = $|K|$. For instance, mentioning the three KPIs from Equation 6.22, such a scaling vector ς could look as follows:

$$\varsigma = \begin{pmatrix} -1 \\ +1 \\ -1 \end{pmatrix} \text{for} \begin{pmatrix} \text{DCR} \\ \text{HOSR} \\ \text{CL} \end{pmatrix} \tag{6.24}$$

When multiplying the δs with the scaling vector ς (note that they have the same size due to the same number of observed KPIs), the results are vectors δ^* where

positive values indicate a KPI improvement and negative values indicate a worsening, i.e.:

$$\delta_c^*(\sigma_i, \sigma_{\text{current}}) = \delta_c(\sigma_i, \sigma_{\text{current}}) \cdot \varsigma \tag{6.25}$$

Continuing the example in Equation 6.22, the scaled $\delta^*_{\text{CLASS}_1}(\sigma_{12}, \sigma_{22})$ would be calculated as follows:

$$\delta^*_{\text{CLASS}_1}(\sigma_{12}, \sigma_{22}) = \begin{pmatrix} 0.02 \\ -0.02 \\ 0.05 \end{pmatrix} \cdot \begin{pmatrix} -1 \\ +1 \\ -1 \end{pmatrix} = \begin{pmatrix} -0.02 \\ -0.02 \\ -0.05 \end{pmatrix} \tag{6.26}$$

Interpreting this example, it can be said that the combined SCV set σ_{12} worsens the performance of all KPIs compared to the currently deployed combined SCV set σ_{22}.

The normalized deltas that contain only negative values definitely lead to a worse performance for all KPIs and hence, can be rejected for further calculations. Hence, a function needs to be defined identifying these combined SCV sets:

$$\text{filter}(\delta_c^*(\sigma_i, \sigma_{\text{current}})) = \begin{cases} 0 & \text{if } \forall k_i \in K : \text{proj}_i(\delta_c^*(\sigma_i, \sigma_{\text{current}})) < 0 \\ 1 & \text{otherwise} \end{cases} \tag{6.27}$$

Thereby, $\text{proj}_i(\delta_c^*(\sigma_i, \sigma_{\text{current}}))$ is a function that delivers the projection on the ith element of the vector $\delta_c^*(\sigma_i, \sigma_{\text{current}})$. Consequently, the set of combined SCV sets that can be filtered out for a specific context class $c \in C$, can be defined as:

$$Z_c = \left\{ \sigma_i | \sigma_i \in \Sigma_c^{\Gamma_c} \wedge \text{filter}(\delta_c^*(\sigma_i, \sigma_{\text{current}})) = 0 \right\} \tag{6.28}$$

Finally, the set $\Sigma_c^{Z_c}$ containing only combined SCV sets that have the potential to leading to a better network performance, are described as:

$$\Sigma_c^{Z_c} = \Sigma_c^{\Gamma_c} \setminus Z_c \tag{6.29}$$

and accordingly, the set $\Delta_c^{Z_c}$ only contains scaled deltas that have at least one KPI with a positive delta value (cf. Equation 6.30).

$$\Delta_c^{Z_c} = \left\{ \delta_c^*(\sigma_i, \sigma_{\text{current}}) | \sigma_i, \sigma_{\text{current}} \in \Sigma_c^{Z_c}, c \in C \right\} \tag{6.30}$$

These filtered deltas $\Delta_c^{Z_c}$ serve as input for the performance indication step. However, another step needs to be executed beforehand to do this performance indication, namely, the KPI target vs. network performance comparison.

KPI Target vs. Network Performance Comparison
After determining in the previous steps what impact certain combined SCV sets would have on the network compared to the currently active SCV set, it has to be identified which KPIs need an improvement in order to fulfill the technical objectives. Therefore, the KPI target vs. network performance comparison step takes the technical objectives into account. This requires the consideration of the current network behavior in terms of KPIs. The KPI targets ϑ_c given in the objective model are subtracted from the current network performance ξ_c and the result is multiplied by the scaling vector ς as defined in Equation 6.24. This step again needs to be done for each individual context class, i.e., the current network performance is also indicated per context class by averaging over the KPI values of all cells in the respective context class.

$$\lambda_c = (\xi_c - \vartheta_c) \cdot \varsigma \tag{6.31}$$

λ_c represents a vector where negative values indicate that a KPI needs to be improved in order to fulfill the KPI target of context class c and positive values indicate that the current SCV set combination fulfills the respective KPI target. ξ_c represents a vector containing an aggregated KPI measurement (coming from the network) for each KPI that is monitored and ϑ_c represents a vector of target values for each KPI (coming from the objective model). Consequently, the set of λs, one for each context class c_i, is defined as:

$$\Lambda = \{\lambda_{c_1}, \lambda_{c_2}, ..., \lambda_{c_{|C|}}\} \tag{6.32}$$

For instance, assuming a current network performance for a c = CLASS_3 of DCR = 0.06, HOSR = 0.97 and CL = 0.43 and the KPI targets of CLASS_3 depicted in Listing 6.4, $\lambda_{\text{CLASS\_3}}$ would be calculated as follows:

$$\lambda_{\text{CLASS\_3}} = \left(\begin{pmatrix} 0.06 \\ 0.97 \\ 0.43 \end{pmatrix} - \begin{pmatrix} 0.02 \\ 0.95 \\ 0.50 \end{pmatrix} \right) \cdot \begin{pmatrix} -1 \\ +1 \\ -1 \end{pmatrix} = \begin{pmatrix} -0.04 \\ 0.02 \\ 0.07 \end{pmatrix} \tag{6.33}$$

The result indicates that only the KPI DCR needs to be improved in order to fulfill the technical objectives of CLASS_3 while the targets for HOSR and CL are already fulfilled. Using this information and the filtered deltas (cf. Section 6.2.4.1), the objective manager can give an indication about the performance when changing from the current combined SCV set to another one, i.e., one that possibly does better in fulfilling operator objectives.

Performance Indication
Based on the filtered deltas $\Delta_c^{Z_c}$ for each context class and the respective $\lambda_c \in \Lambda$ vector a prediction can be made about the performance of all possible combined SCV sets, i.e., sets that can improve at least one KPI compared to the currently active combined SCV set. This is again done for each context class individually. Therefore, the respective λ_c vector is added to each $\delta_c^*(\sigma_i, \sigma_{\text{current}}) \in \Delta_c^{Z_c}$ resulting

in the performance indication $\iota_c(\sigma_i, \sigma_{\text{current}})$ as shown in Equation 6.34. Note that both, the delta vector as well as the KPI target vs. network performance comparison vector are already scaled. Hence, the result of this calculation is also scaled and can be interpreted directly.

$$\iota_c(\sigma_i, \sigma_{\text{current}}) = \delta_c^*(\sigma_i, \sigma_{\text{current}}) + \lambda_c \tag{6.34}$$

Consequently, the result of doing this for all filtered deltas $\delta_c^*(\sigma_i, \sigma_{\text{current}})$ within a context class $c \in C$ is a set of $\iota_c(\sigma_i, \sigma_{\text{current}})$, i.e.:

$$I_c = \left\{ \iota_c(\sigma_i, \sigma_{\text{current}}) | \sigma_i, \sigma_{\text{current}} \in \Sigma_c^{Z_c}, c \in C \right\} \tag{6.35}$$

This set contains a vector for each possible combined SCV set indicating for each KPI whether the respective target is fulfilled or not. Positive values thereby indicate the fulfillment of a target while negative ones imply the violation of a KPI target. And even further, these indicators not only give the MNO an impression about the possible fulfillment or violation of KPI targets, they also report the degree of fulfillment or violation. The more negative the indicated value, the farther is the predicted effect away from accomplishing a KPI target and vice versa.

For instance, when interpreting the result of Equation 6.33, it can be said that the current SCV set combination does not meet the operator objectives in terms of DCR. Hence, if possible, a set needs to be found that fulfills DCR while at the same time, does not worsen the other KPIs too much such that in the end, all KPI targets can be accomplished. For the further calculation, a $\delta^*_{\text{CLASS\_3}}(\sigma_{31}, \sigma_{22})$ is assumed with a scaled delta value for DCR = 0.05, HOSR = 0.00 and CL = −0.04.

$$\iota_{\text{CLASS\_3}}(\sigma_{31}, \sigma_{22}) = \begin{pmatrix} 0.05 \\ 0.00 \\ -0.04 \end{pmatrix} + \begin{pmatrix} -0.04 \\ 0.02 \\ 0.07 \end{pmatrix} = \begin{pmatrix} 0.01 \\ 0.02 \\ 0.03 \end{pmatrix} \tag{6.36}$$

As can be seen, the resulting $\iota_{\text{CLASS\_3}}(\sigma_{31}, \sigma_{22})$ only contains positive values, showing that all objectives can be fulfilled when applying the combined SCV set σ_{31}. Note that this is the case, even though this set worsens the currently active set σ_{22} in terms of CL, it does not downgrade the KPI value to a degree that the KPI target can not be met anymore.

SCV Set Selection

In the final step of the objective manager's process, the best possible combined SCV set is selected for each context class. By using the I_cs and considering the weights which the MNO has defined for the technical objectives, a utility for each SCV set combination can be calculated that reaches from 0 (none of the KPI targets fulfilled) to 1 (all KPI targets fulfilled). For each context class the SCV set with the highest utility will be chosen to be deployed to all cells in the network within the corresponding context class. Since it is highly possible that two or more SCV set combinations have the same (highest) utility, the MNO can decide between

different options for the selection of the SCV set. First of all, one of the SCV sets with the highest utility can be selected randomly. This would be a conceivable option if the MNO does not care about constantly improving the KPIs but is perfectly satisfied with just fulfilling the KPI targets. Also, this method increases the chance of deploying an up-to-now untested SCV set and hence, enhances the measurements data pool with KPI values of a new combined SCV set.

Formally, the utility is calculated using a function that indicates whether an objective is satisfied or not, i.e.:

$$\text{sat}(\sigma_i, k_j) = \begin{cases} 1 & \text{if } \text{proj}_j(\iota_c(\sigma_i, \sigma_{\text{current}})) \geq 0 \\ 0 & \text{otherwise} \end{cases} \tag{6.37}$$

That is, a set σ_i fulfills a KPI target if the performance indicator for a KPI k_j is greater or equal 0. The performance indicator is given by a function $\text{proj}_j(\iota_c)$, returning a projection on the jth element of ι_c. Since the MNO defines weights in the objective model, a weighted sum can be calculated using the sat() function in the utility function $U(\sigma_i)$.

$$U(\sigma_i) = \sum_{1 \leq j \leq |K|} \text{sat}(\sigma_i, k_j) \frac{w_{k_j}}{\bar{w}} \tag{6.38}$$

with $\bar{w} = \sum_{i \in K} w_i$.

Another option is to take an additional parameter into account. In case that an MNO wants to achieve an optimal network performance with respect to KPIs, instead of only aiming at the fulfillment of technical objectives, not only the utility, but also the distance between the expected network performance for different SCV sets and the KPI targets is relevant. Therefore, it is necessary to normalize the KPI ranges first, such that a distance method can be used afterwards in order to calculate the gap between the KPI values which would be achieved by a new combined SCV set, and the targeted KPI values.

Formally, for all combined SCV sets with the highest utility in a certain context class, the distance is calculated using a function $D(\sigma_i)$, i.e.:

$$D(\sigma_i) = \left(\text{proj}_1(\iota_c(\sigma_i, \sigma_{\text{current}})) \cdot w_{k_1}\right) + \cdots + \left(\text{proj}_{|K|}(\iota_c(\sigma_i, \sigma_{\text{current}})) \cdot w_{k_{|K|}}\right) \tag{6.39}$$

where $\text{proj}_j(\iota_c)$ refers to the same projection function as defined above. Hence, this distance function determines the combined SCV set with the highest distance value since this refers to an over-fulfillment of KPI targets. Note that weights of the KPI targets are again consulted in order to take the operator's preferences with respect to the KPIs into account.

The selection of best possible SCV sets is exemplified in Section 6.3.

6.2.5. Policy System

In general, the policy system as it is used within ASM looks similar to the one in PBSM and ODSM. That is, it consists of a policy repository containing the policy rules, a PDP which decides about the applicability of a rule, and the PEP which enforces SCV sets in the network. However, due to the context classes concept which is introduced in this chapter, the general form of a rule in the policy repository has changed. In contrast to previous approaches, the objective manager determines the set of best possible SCV sets for each context class instead of each region in the context state space.

```
1 IF context class THEN SCV set combination
```

Listing 6.6: SCV set policy rule in ASM

This reduces the size of the output SCV set policy since only one rule per SON function and context class is needed. Also, this structure of the policy requires an intermediate step between getting the current operational context and the decision for an applicable rule, since the operational context and the condition parts of the rules do not match each other. Hence, when receiving the operational context of a cell, it has to be checked by means of the manual context classes model first, to which context class the cell belongs. This context class can then be provided to the PDP which selects appropriate SCV sets.

6.2.6. Adaptation Process

Since the MNO wants to keep the network performance at a certain level, but the network is constantly affected by dynamic changes, the configuration process described in the previous section has to run continuously. Thus, the database containing real KPI measurements can be built up and extended every time KPI measurements are taken in the network. However, this pool of network measurements is not available from the beginning and hence, a starting point is needed for the initial calculation of optimal SCV sets. At a later point in time, the MNO then has to decide which of the effect sub-models is the preferable one, either the combined effect model that is based on simulations done by the SON function manufacturer or the real network effect model. Only the measurements coming from the real network can accurately represent the actual performance and setup within the MNO environment, in particular compared to the simulator-generated predictions in the combined effect model. Hence, these real measurements are the ones that are more promising in terms of network behavior prediction.

In general, the decision which pool is the preferable one depends on the time and the number of measurements available in the network measurements database. In the beginning, the MNO has to trust the manufacturer-provided model since this is the only available source for KPI effect predictions under certain SCV sets.

During this phase, as many measurements for different SCV sets and different context classes as possible should be gathered such that a big number of measurements is available after finishing the collection phase. In Section 6.2.4.1, two different methods for the selection of the best SCV sets have been proposed, i.e., the utility-based and a distance-based method. Combining both methods, but for different points in time, results in the most effective approach: During the collection phase, it is helpful to choose SCV set combinations randomly in case of equal utilities since it guarantees a big variety of combined SCV sets in the real network effect model. The duration of the collection phase can be defined by the MNO, and different options for this definition exist, e.g., a fixed number of measurement iterations that has to be completed or a defined degree of convergence has to be achieved either per SCV set or for all SCV sets. Which of the options is the preferable one, depends on the MNO and gives the operator another possibility to affect SON management actions, thereby gaining trust in the SON management system (cf. Objective 4 in Section 1.2.4).

After the collection phase, SCV sets are usually chosen based on the predictions in the real network effect model since this model reflects the behavior of SON functions in the network in a more realistic way. However, network measurements are still collected after finishing the collection phase. Note that the MNO can decide to hark back to the combined effect model at any time, e.g, when he or she recognizes a situation in the network where KPI targets for one or more context classes can not be fulfilled with the currently available SCV sets in the real network effect model. In that case, only combined SCV sets should be chosen for being deployed in the network where no measurements exist in the real network effect model yet. Therefore, the MNO could also define a lower threshold in terms of utility that indicates how much risk he is willing to take for testing SCV sets with expected lower utility. Doing so, combined SCV sets are taken into account whose predicted utility is lower than the predicted utility of already tested sets. However, since their KPI predictions have been generated in a simulation environment they may have a different (and possibly better) impact in the real network.

It is also possible to only rely on the real network effect model which could be the case if an MNO has trust issues regarding the simulations done by the SON function manufacturer or if a manufacturer does not want to provide an initial effect model. Note that the main intention of the manufacturer-provided effect models within this concept is to have a starting point or some clues of how a SON function might perform. The information stored in these models can be evaluated over time and discarded if enough knowledge is acquired to feed the SON management with a real network effect model solely. However, without having any knowledge about the effects certain SCV sets possibly have in the network, the objective manager has to choose SCV set combinations randomly for the duration of the collection phase. One major disadvantage of such a procedure is that the adaptation process is significantly slowed down and therefore should only be an option if one of the mentioned situations occur.

A situation that also affects the choice of an appropriate effect model is the deployment of a new SON function. In such a case, no measurements are available in the real network effect model about a combination of already deployed SCV sets and SCV sets of the new SON function. A solution for this problem could be to combine the knowledge of both types of effect models, i.e., taking KPI value predictions from the real network effect model and combining them with the predictions in the manufacturer-provided effect model of the new SON function according to one of the options described in Section 5.2.3.1.

An enhancement to the presented methodology could be to apply an aging process when generating or updating the real network effect model based on KPI measurements in the network measurements database. Such an aging process rates older measurements less strongly than newer ones. This is reasonable in two ways: on the one hand, the amount of data gets very huge over time and therefore has to be reduced since it makes the configuration process slower. On the other hand, older measurements reflect the situation in the network worse than newer ones since the network underlies permanent changes and these changes are not captured in too old measurements. This is also the reason why measurements should be tagged with the date they were taken (as depicted in Figure 6.5). Such an aging process using the weighted moving average could look as follows: Arrange raw measurements $\mu(\cdot)$ in descending order based on the measurement date, such that $\mu_1(\cdot)$ is the newest and $\mu_N(\cdot)$ is the oldest measurement. Note that each measurement $\mu(\cdot)$ is actually a vector containing a KPI value for each KPI $k \in K$. Further note that, as described in Section 6.2.3.3, this aging has to be done for each context class and, within the context class, each combined SCV set individually. An aggregated measurement is then defined as:

$$\tilde{\mu}_{\text{aging}}(\cdot) = \frac{\sum_{n=1}^{N} \frac{1}{n} \mu_n(\cdot)}{\sum_{n=1}^{N} \frac{1}{n}} \tag{6.40}$$

6.3. Example

The generation of a real network effect model as well as the methodology for the selection of optimal SCV sets is exemplified in this section. Since the biggest difference to previous approaches is the existence of a new effect model, the methodology part will focus on the usage of the real network effect model, i.e., imagine that the collection phase is already finished and that the measurements effect model is trustworthy enough for an operator to be used. However, note that the SCV set selection works accordingly for the combined effect model. It is furthermore assumed that an operator wants to rank measurements based on their age, i.e., the aging process is applied as described in Equation 6.40.

The MNO delivers two models as input: First, a manual context classes model is created by means of the context attributes model, is needed. The context classes

model which this example is based on, is already depicted in Listing 6.2. The objective model defines targets for each context class and each KPI in the network and is illustrated in Listing 6.4. For this example, three KPIs, namely DCR, HOSR and CL are observed.

During operation of the mobile network, measurements have been gathered reflecting the behavior of the network. An excerpt of these measurements, i.e., the measurements for context class CLASS_3, are shown in Table 6.1. Each measurement consists of a unique ID, the date it was taken, the context class to which it belongs, the deployed SCV sets for MLB and MRO and, most important, the values that have been measured for the three KPIs mentioned above.

The first part of the example describes the derivation of a real network effect model based on Table 6.1.

Table 6.1.: Exemplary raw measurements for context class CLASS_3

ID	DATE	CLASS	MLB	MRO	DCR	HOSR	CL
...	...	...	...	...	...	...	...
168	19-12-2018 07:11:19	CLASS_3	1	2	0.013	0.944	0.495
169	19-12-2018 07:24:33	CLASS_3	1	2	0.017	0.932	0.510
170	19-12-2018 07:07:26	CLASS_3	3	2	0.024	0.971	0.529
171	19-12-2018 08:03:01	CLASS_3	3	2	0.032	0.983	0.432
172	19-12-2018 07:58:45	CLASS_3	3	1	0.025	0.893	0.566
173	19-12-2018 08:01:51	CLASS_3	2	3	0.041	0.920	0.544
174	19-12-2018 07:46:18	CLASS_3	1	1	0.021	0.918	0.470
175	19-12-2018 07:15:52	CLASS_3	1	2	0.016	0.988	0.458
176	19-12-2018 07:32:12	CLASS_3	1	2	0.014	0.977	0.498
177	19-12-2018 08:00:43	CLASS_3	1	2	0.012	0.941	0.502
...	...	...	...	...	...	...	...

6.3.1. Derivation of a Real Network Effect Model

The first step of generating a real network effect model comprises the selection of appropriate measurements for each context class and each tested SCV set combination. Thereby, the focus is on selecting SCV sets for all cells in class CLASS_3. Since the aging process shall be applied, measurements also have to be sorted for each of the combined SCV sets. In Table 6.1, five different sets can be identified: σ_{12}, σ_{32}, σ_{31}, σ_{23} and σ_{11} where the first number represents the applied MLB function SCV set and the second one MRO respectively. Sorting these measurements according to their age, i.e., the DATE, leads to Table 6.2.

Note that the date is illustrated in a shortened way and each measurement is assigned a formalized name where the lower index indicates the age of a measurement compared to other measurements within the same SCV set (the lower

the number, the newer the measurement). Following the formalization in Section 6.2.3.3, the set of measurements for CLASS_3 is divided into five sorted subsets, one for each SCV set: $M_{\sigma_{11}}^{\text{CLASS\_3}}$, $M_{\sigma_{12}}^{\text{CLASS\_3}}$, $M_{\sigma_{23}}^{\text{CLASS\_3}}$, $M_{\sigma_{31}}^{\text{CLASS\_3}}$ and $M_{\sigma_{32}}^{\text{CLASS\_3}}$.

Table 6.2.: Sorted raw measurements for context class CLASS_3

ID	DATE	CLASS	MLB	MRO	DCR	HOSR	CL	NAME
				σ_{11}				
174	07:46:18	CLASS_3	1	1	0.021	0.918	0.470	$\mu_1^{\sigma_{11}}(\cdot)$
				σ_{12}				
177	08:00:43	CLASS_3	1	2	0.012	0.941	0.502	$\mu_1^{\sigma_{12}}(\cdot)$
176	07:32:12	CLASS_3	1	2	0.014	0.977	0.498	$\mu_2^{\sigma_{12}}(\cdot)$
169	07:24:33	CLASS_3	1	2	0.017	0.932	0.510	$\mu_3^{\sigma_{12}}(\cdot)$
175	07:15:52	CLASS_3	1	2	0.016	0.988	0.458	$\mu_4^{\sigma_{12}}(\cdot)$
168	07:11:19	CLASS_3	1	2	0.013	0.944	0.495	$\mu_5^{\sigma_{12}}(\cdot)$
				σ_{23}				
173	08:01:51	CLASS_3	2	3	0.041	0.920	0.544	$\mu_1^{\sigma_{23}}(\cdot)$
				σ_{31}				
172	07:58:45	CLASS_3	3	1	0.025	0.893	0.566	$\mu_1^{\sigma_{31}}(\cdot)$
				σ_{32}				
171	08:03:01	CLASS_3	3	2	0.032	0.983	0.432	$\mu_1^{\sigma_{32}}(\cdot)$
170	07:07:26	CLASS_3	3	2	0.024	0.971	0.529	$\mu_2^{\sigma_{32}}(\cdot)$

In the second step, statistics for the measurements can be generated applying the aging process. For $M_{\sigma_{11}}^{\text{CLASS\_3}}$, $M_{\sigma_{23}}^{\text{CLASS\_3}}$ and $M_{\sigma_{31}}^{\text{CLASS\_3}}$, this is quite obvious since these sets only contain one measurement and hence, build the statistic for the respective combined SCV sets σ_{11}, σ_{23} and σ_{31}. For σ_{12} and σ_{32}, Equation 6.40 is applied for each of the three KPIs. For instance, the aggregated value $\tilde{\mu}_{\sigma_{12}}^{\text{CLASS\_3}}(\text{DCR})$ for SCV set σ_{12} and KPI DCR is calculated as follows:

$$\tilde{\mu}_{\sigma_{12}}^{\text{CLASS\_3}}(\text{DCR}) = \frac{0.012 + \frac{1}{2}\cdot 0.014 + \frac{1}{3}\cdot 0.017 + \frac{1}{4}\cdot 0.016 + \frac{1}{5}\cdot 0.013}{1 + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \frac{1}{5}} \approx 0,0137 \quad (6.41)$$

Doing this for all SCV sets and all KPIs leads to the SCV set statistics, i.e., the real network effect model for CLASS_3 as depicted in Listing 6.7.

6.3.2. SCV Set Calculation

When having the real network effect model, the SCV set policy can be calculated applying the five steps described in Section 6.2.4.1. The first step is the delta determination and therefor, it is assumed that the currently active combined SCV

```
{
  ...
  IF CLASS_3 AND MLB_1 AND MRO_1
    THEN DCR = 0.021 AND HOSR = 0.918 AND CL = 0.470
  IF CLASS_3 AND MLB_1 AND MRO_2
    THEN DCR = 0.014 AND HOSR = 0.953 AND CL = 0.497
  IF CLASS_3 AND MLB_2 AND MRO_3
    THEN DCR = 0.041 AND HOSR = 0.920 AND CL = 0.544
  IF CLASS_3 AND MLB_3 AND MRO_1
    THEN DCR = 0.025 AND HOSR = 0.893 AND CL = 0.566
  IF CLASS_3 AND MLB_3 AND MRO_2
    THEN DCR = 0.029 AND HOSR = 0.979 AND CL = 0.464
  ...
}
```

Listing 6.7: Excerpt of the exemplary real network effect model for CLASS_3

set in CLASS_3 is σ_{32}. For the illustration of KPI values within a measurement, the representation as vector with the same order as Section 6.2.4.1 is chosen, i.e., the DCR value on top, the HOSR value in the middle and the CL value on the bottom. Table 6.3 shows the results for the delta determination. In the first line, the deltas between the currently active set σ_{32} and all other sets is shown. Note that also $\delta(\sigma_{32}, \sigma_{32})$ is calculated since otherwise, it is excluded from further computations and can not be selected even though it probably produces the best network behavior. The table also contains the results for multiplying the deltas with the scaling vector in the second line, i.e., the $\delta^*(\sigma_i, \sigma_{32})$. Finally, the results for applying the filter() function are represented in the bottom line.

Table 6.3.: Deltas, scaled deltas and filtered deltas for all combined SCV sets in context class CLASS_3

	σ_{11}	σ_{12}	σ_{23}	σ_{31}	σ_{32}
$\delta(\sigma_i, \sigma_{32})$	$\begin{pmatrix} -0.008 \\ -0.061 \\ 0.006 \end{pmatrix}$	$\begin{pmatrix} -0.015 \\ -0.024 \\ 0.033 \end{pmatrix}$	$\begin{pmatrix} 0.012 \\ -0.059 \\ 0.080 \end{pmatrix}$	$\begin{pmatrix} -0.004 \\ -0.086 \\ 0.102 \end{pmatrix}$	$\begin{pmatrix} 0.0 \\ 0.0 \\ 0.0 \end{pmatrix}$
$\delta^*(\sigma_i, \sigma_{32})$	$\begin{pmatrix} 0.008 \\ -0.061 \\ -0.006 \end{pmatrix}$	$\begin{pmatrix} 0.015 \\ -0.024 \\ -0.033 \end{pmatrix}$	$\begin{pmatrix} -0.012 \\ -0.059 \\ -0.080 \end{pmatrix}$	$\begin{pmatrix} 0.004 \\ -0.086 \\ -0.102 \end{pmatrix}$	$\begin{pmatrix} 0.0 \\ 0.0 \\ 0.0 \end{pmatrix}$
filter($\delta^*(\sigma_i, \sigma_{32})$)	1	1	0	1	1

As can be seen in the scaled deltas δ^*, when changing from σ_{32} to any other SCV set, the KPIs HOSR and CL would probably be worsened due to the negative δ^*

values. However, the aim of ASM and SON management in general is not the identification of the SCV set with the best KPI values, but the optimization of operator targets. Even though two of three KPI get worse for most of the SCV sets, they may still fulfill operator targets and hence, should not be discarded for further calculations. The only SCV set influencing the behavior of all three KPIs in a bad way, is combined SCV set σ_{23} which should be neglected accordingly. This is displayed in Table 6.3 by the gray shaded cell: When applying the filter($\delta^*(\sigma_{23}, \sigma_{32})$) function, it returns 0 due to negative values for all observed KPIs.

In the next step of the algorithm, the current network performance is taken into account by comparing it with the KPI targets for context class CLASS_3. For this calculation, it is assumed that the current performance in terms of DCR is 0.027, for HOSR it is 0.921 and for CL it is 0.458. Hence, the KPI target vs. network performance comparison $\lambda_{\text{CLASS\_3}}$ is calculated as follows:

$$\lambda_{\text{CLASS\_3}} = \left(\begin{pmatrix} 0.027 \\ 0.921 \\ 0.458 \end{pmatrix} - \begin{pmatrix} 0.020 \\ 0.950 \\ 0.500 \end{pmatrix} \right) \cdot \begin{pmatrix} -1 \\ +1 \\ -1 \end{pmatrix} = \begin{pmatrix} -0.007 \\ -0.029 \\ 0.042 \end{pmatrix} \tag{6.42}$$

That is, with the current SCV set combination, only KPI CL fulfills the operator target. Also, even an SCV set combination that downgrades CL by an amount of 0.042 still fulfills the respective target which is a crucial ascertainment.

Using the $\lambda_{\text{CLASS\_3}}$ and the set of filtered δ^*s, an indication about the expected performance can be made by adding the values. Afterwards, the satisfaction of the respective KPI target can be calculated by using the sat() function. Generating the weighted sum for each σ_i by means of the KPI target weight w_k delivers the overall utilities. Note that in Table 6.4 not the whole indication vector $\iota(\sigma_i, \sigma_{32})$ is displayed, but the projections on indication values for the respective KPIs.

Table 6.4.: Scoring of the exemplary combined SCV sets

	$o_{\text{DCR}} = (0.02,\ 0.5)$			$o_{\text{HOSR}} = (0.95,\ ,0.2)$			$o_{\text{CL}} = (0.5,\ 0.3)$			$U(\sigma_i)$
	$\text{proj}_{\text{DCR}}(\iota(\sigma_i, \sigma_{32}))$	$\text{sat}(\sigma_i, DCR)$	w_{DCR}	$\text{proj}_{\text{HOSR}}(\iota(\sigma_i, \sigma_{32}))$	$\text{sat}(\sigma_i, HOSR)$	w_{HOSR}	$\text{proj}_{\text{CL}}(\iota(\sigma_i, \sigma_{32}))$	$\text{sat}(\sigma_i, CL)$	w_{CL}	
σ_{11}	0.001	1	0.5	−0.090	0	0.2	0.036	1	0.3	0.8
σ_{12}	0.008	1		−0.053	0		0.009	1		0.8
σ_{31}	−0.003	0		−0.115	0		−0.060	0		0.0
σ_{32}	−0.007	0		−0.029	0		0.042	1		0.3

As can be seen in this table, none of the SCV sets can fulfill the HOSR target which may be surprising since, according to the effect model, two of the SCV sets, namely σ_{12} and σ_{32} actually should fulfill this target. Here, one of the major advantages of the ASM approach (compared to PBSM and ODSM) can be seen: By taking the current network performance into account which gives an indication about the actual network behavior of currently deployed SCV sets, estimations are more accurate since they do not only rely on predictions in the effect models.

Furthermore, two SCV set combinations, namely σ_{11} and σ_{12}, achieve the same utility $U(\sigma_i) = 0.8$ since they both fulfill the DCR and CL target. The fulfillment of the CL target may be surprising the other way round: Both SCV sets worsen the CL compared to the active SCV set σ_{32}. However, with the currently deployed SCV set, the target for CL is over-fulfilled and hence, a little degradation does not affect the KPI target fulfillment. In order to make a decision for one of the equally rated SCV sets, the preferences of the MNO come into play. One of the two sets can be either chosen randomly or, the distance to the KPI targets can be determined in order to find the best possible solution. This can be done using Equation 6.39 and is exemplary shown for σ_{12}:

$$D(\sigma_{12}) = (0.008 \cdot 0.5) + (-0.053 \cdot 0.3) + (0.009 \cdot 0.2) = -0.0039 \tag{6.43}$$

$D(\sigma_{11})$ can be determined in the same way, resulting in −0.0067. Consequently, σ_{12} is the SCV set to be selected for context class CLASS_3 since $D(\sigma_{12}) > D(\sigma_{11})$.

6.3.3. Generation of SCV Set Policy

An excerpt of the final SCV set policy is depicted in Listing 6.8. According to the exemplary context classes model, CLASS_3 is defined as *(location = rural OR location = highway) AND (cell_type = micro OR cell_type = macro)*. Hence, for all cells in this operational context, the SCV sets *MLB_1* and *MRO_2* are applied.

```
{
  ...
  IF CLASS_3 THEN MLB = MLB_1
  IF CLASS_3 THEN MRO = MRO_2
  ...
}
```

Listing 6.8: Excerpt of SCV set policy based on the scoring of the combined SCV sets

6.4. Related Work

Since the presented ASM approach can be seen as an enhancement to ODSM, the related work that has been presented in Section 3.4, Section 4.4 and Section 5.4 is

also relevant and applicable for this approach. The delta to previous approaches is the definition of a real network effect model that takes measurements from the real network into account, therefore creating a model with more realistic effect predictions.

In [HK14], the authors present an approach to generate so-called SON function performance models that can be used to estimate the SON functions' performance. Therefore, they also developed a mobile network simulator called SiMoNe [RHK16]. However, these models are only simulation based and can be compared to the manufacturer-provided effect models whose disadvantages have also been discussed in this chapter, leading to the presented approach.

The authors of [HSK18] have also investigated how a set of concurrently acting SON functions impact the behavior of the network. This can be compared to the information which is stored in the combined effect model. While this approach reflects the real situation in mobile networks, meaning that a variety of SON functions is deployed on the cells, it is still simulation based. Furthermore, none of the approaches provides a solution for managing several SON functions and a methodology for picking optimal SCV sets.

7

Cognitive SON Management

The final stage of development is CSM. While in the ASM approach, a simple method for learning an effect model over time based on KPI measurements from the network has been presented, CSM uses real machine learning algorithms to predict network behavior. Therefore, mainly two new sub-models are presented in this chapter: A KPI-based context classes model and a learned effect model. The focus in this chapter is on the generation of these two sub-models. Furthermore, the methodology for the selection of appropriate SCV sets is similar to the one presented in Section 6.2.4 and hence, the focus in the methodology part is on the interaction of different models. Consequently, the example presented in Section 7.3 also focuses on the new models and their combination.

7.1. Motivation

The context model as presented in the previous chapter, has one significant disadvantage: The definitions of context classes in the manual context classes model are based on experience and hence, may be not applicable. Thereby, one has to distinguish between two types of application. On the one hand, these classes are highly relevant for the partitioning of the huge context state space into manageable regions and the definition of technical objectives for each of these regions. On the other hand, for context-dependent predictions in the effect model, these context classes imply the assumption that cells within the same class have similar behavior. However, due to the complexity of modern cellular networks even experienced operators may not capture all network details as well as the behavior of cells in the network and thus, their classification may be incorrect for some cells, leading to a big range of dispersion in matters of their behavior for certain SCV sets. Hence, a method needs to be found which guarantees a classification of cells into similarly behaving cell clusters such that predictions in the effect model can be more accurate.

Furthermore, the ASM approach comprises a second problem. While the real network effect model is a great step to reflect the actual behavior of cells under certain SCV sets, there is still the problem that this model is incomplete and only contains predictions for already tested SCV sets. That is, SCV sets that could never achieve the highest utility during the operation of the network, will never

be part of the real network effect model. This fact reduces the opportunity to run the network in an *optimal* instead of just the *best possible* way. However, testing all different SCV sets in the real network is simply not feasible. In addition, a situation may occur where none of the already tested SCV sets do well in achieving given operator objectives. To overcome this problem, it would be helpful to predict the behavior of untested sets which possibly do better in fulfilling KPIs targets.

The above mentioned shortcomings motivate the application of machine learning algorithms in order to further improve the input models processes by the objective manager. Hence, the objective that is mainly addressed in this chapter, is Objective 3 which refers to the generation of a learned, i.e., KPI-based context classes model and a learned effect model:

Objective 3 *Reliably estimate the performance of untested SCV sets dependent on automatically derived context information which has been reduced to a manageable level.*

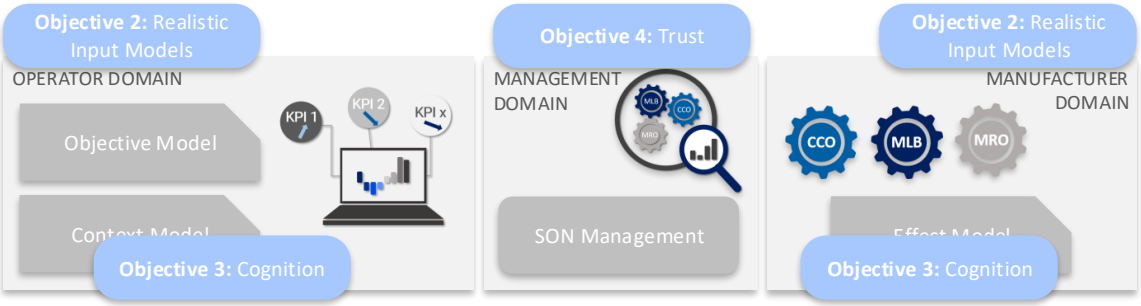


Figure 7.1.: Objectives of CSM

The usage of two new sub-models also marks an important step towards generating more realistic input models, making the initial manufacturer-provided effect models and the manually operator-defined context classes model more and more redundant.

Objective 2 *Automatically generate realistic and complete input models which are continuously updated according to the current network state such that they optimally support the SON management system in finding the best possible network configuration.*

Not least, also Objective 4 is important for the development of a CSM approach. More than ever, SON management underlies highly automated activities, especially when creating learned models. Hence, techniques need to be identified to win the confidence of MNOs with respect to these models and their application.

Objective 4 *Develop a fully automated SON management system which allows MNOs to comprehend, restrict and influence automated actions at any time.*

7.2. Approach

As for the previously introduced ASM approach, three models serve as initial input to CSM. The objective model which describes targets and preferences in terms of network KPIs, equals the one presented in Section 6.2.2 and is not described in more detail here. Second, two types of context models are provided by the MNO: A context attributes model that comprises network properties and their values and a context classes model which partitions the resulting context state space into reasonable context regions that all pursue the same KPI targets. Third, an effect model is provided by the manufacturer of a SON function for each of the particular SON functions describing the effects that certain SCV sets have in the manufacturers' simulation environment.

The initial effect models are brought together by the objective manager in order to make predictions about the effect a combined SCV set has in the network. This information is stored in the combined effect model. In the ASM approach real network measurements are taken into account, building a new sub-model that aggregates these measurements for each operator-defined context class, resulting in the real network effect model. These models have already been presented in detail in Chapter 6 and only the deltas to these models will be introduced in the following. Figure 7.2 illustrates the already known models as well as the enhancements in CSM.

What is new in the CSM approach, is a learning engine within the objective manager adopting techniques in the field of machine learning for the generation of two new models: The learned context classes model which is the result of a KPI-based classification of network cells into cell clusters with similar behavior according to the observed KPIs, and a learned effect model which completes the real network effect model such that for each and every possible SCV set combination, an effect prediction is available. These two models serve as further input to the objective manager which then selects appropriate SCV sets while considering technical objectives and the operator's issues regarding the trustworthiness of the externally provided and internally generated models. It is crucial for the achievement of optimal results that the objective manager picks the models which are most suitable for the current state of the network.

The decisions of the objective manager are summed up in an SCV set policy which is passed to the well-known policy system that deploys SCV sets in the network. Network measurements are conducted at any time, making the generation of models within the management domain a permanently running adaptation and optimization process.

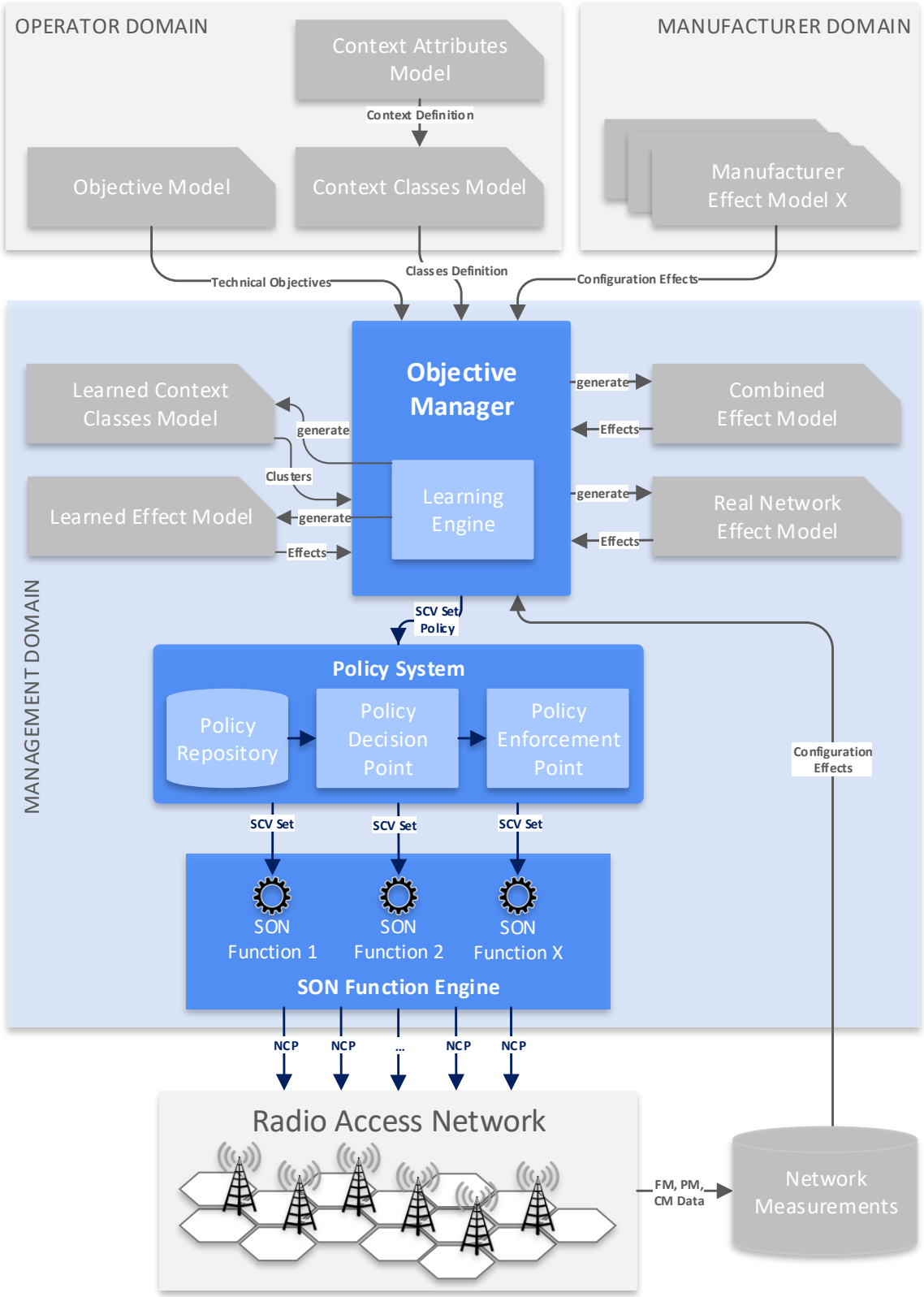


Figure 7.2.: Overview of CSM

7.2.1. Structured Description of Models

The arrangement of sections in this chapter is a bit different than for the previous chapters. Before explaining the derivation of learned input models, it is first necessary to describe the structure of both the context model and effect model such that their purpose gets intelligible. This is due to the fact that for a full understanding of the context model, the effect model needs to be understood and vice versa.

7.2.1.1. Context Model

The context model in CSM is split into three sub-models. The context attributes model is used for the definition of manual context classes, i.e., the context classes model which is the second sub-model. The third model is a machine-generated learned context classes model which is independent of the other context sub-models.

Note that in this section, the learned context model is only described as a result of its derivation process, but not the derivation process itself. This is part of Section 7.2.3. Further note that, besides the learned context classes model, also the other context sub-models are used in CSM. However, they do not vary from those described in the ASM approach and hence, are not presented here in detail.

Learned Context Classes Model
In contrast to the manual context classes model, this model is not based on context attributes and their values, but on the KPI effects that are measured for the respective cells. Since network behavior is defined in terms of KPIs and their values, the learned context classes model truly reflects clusters of cells with similar behavior. The assumption is that these groups might show a more homogeneous behavior in regard to the relationship between SCV sets and the respective KPI. Additionally, the resulting clusters might support an MNO in finding descriptive features of cells and hence, configure newly deployed cells in a suitable way. On a technical level, clustering should characterize cells based on their average KPI under various SCV sets and group cells with similar characteristics into the same group, allowing for more tailored KPI effect predictions per group in the effect model.

Figure 7.3 illustrates the idea behind learning context classes by means of two KPIs CL and DCR. Each region in the context state space is defined by a number of predicates π consisting of a KPI $k \in K$ and a subset $\mathrm{Dom}_{\mathrm{sub}}(k) \subseteq \mathrm{Dom}(k)$, i.e.:

$$\pi_k = (k, \mathrm{Dom}_{\mathrm{sub}}(k)) \tag{7.1}$$

$$\rho_i = \pi_{k_1} \times \pi_{k_2} \times \cdots \times \pi_{k_{|K|}} \tag{7.2}$$

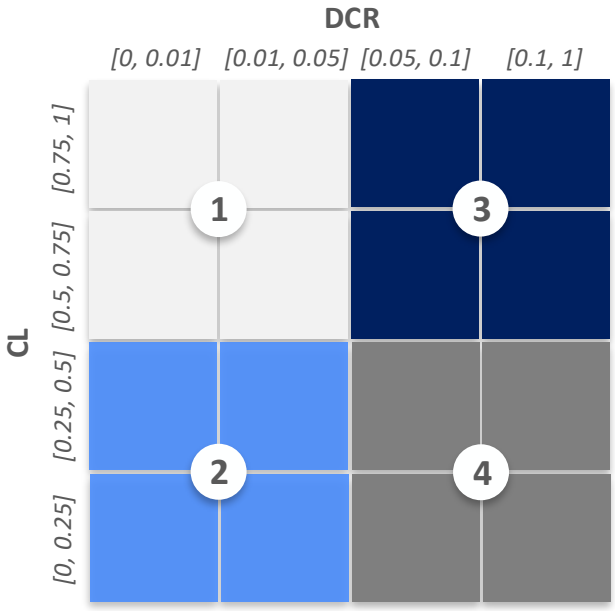


Figure 7.3.: Exemplary KPI-based classification of cells into cell clusters

$$R = \{\rho_1, \rho_2, ..., \rho_{|P|}\} \tag{7.3}$$

Using this definition of a region, a context class is the union of cells in several regions with specific KPI properties. For instance, *LEARNED_CLASS_1* in Figure 7.3 specifies cells with a high load, i.e., ≥ 0.5 and with a low DCR value, i.e., ≤ 0.05, while *LEARNED_CLASS_4* is defined by cells with a low CL, i.e., < 0.5 and a high DCR, i.e., > 0.05.

Formally, the learned context model $\text{CM}_{\text{learned}}$ defines a mapping function from a set of cells Ω to a set of learned context classes $\tilde{C}$, i.e.:

$$\text{CM}_{\text{learned}} : \Omega \mapsto \tilde{C} \tag{7.4}$$

such that

$$\Omega = \{\omega_1, \omega_2, ..., \omega_{|\Omega|}\} \tag{7.5}$$

and

$$\tilde{C} = \{\tilde{c}_1, \tilde{c}_2, ..., \tilde{c}_{|\tilde{C}|}\} \tag{7.6}$$

$|\Omega|$ defines the number of cells in the network and $|\tilde{C}|$ refers to the number of learned context classes, i.e., the number of clusters. The definition of the number of clusters is part of and will be described in Section 7.2.3.

Each cell cluster consists of an arbitrary number of cells $\omega \in \Omega$. Note that each cell can only belong to exactly one learned context class since otherwise the objective

manager is struggling with the selection of appropriate SCV sets for a context class. Consequently, a learned context class c_i^l is defined as follows:

$$\tilde{c}_i = \{\omega_j | 1 \leq j \leq |\Omega| \wedge \forall \tilde{c}_k, i \neq k : \omega_j \notin \tilde{c}_k\} \tag{7.7}$$

According to the formal definitions, the learned context model can be described as a set of rules with the general form depicted in Listing 7.1. That is, the condition parts of the rules refer to the cell and the action parts to the respective context class to which the cells belongs.

```
IF cell THEN learned context class
```

Listing 7.1: Classes definition rule in CSM

An exemplary learned context classes model in CSM is shown in Listing 7.2. Here, only an excerpt is shown for a network with at least 35 network cells and at least five learned context classes. A cell thereby is defined as one of three antennas of a base station, i.e., a site. For instance, site35[0] refers the first cell of a base station, having the ID 35. Note that each of the $|\Omega|$ cells must be assigned to a learned context class.

```
1 {
    ...
3   IF site34[0] THEN LEARNED_CLASS_2
    IF site34[1] THEN LEARNED_CLASS_1
5   IF site34[2] THEN LEARNED_CLASS_4
    IF site35[0] THEN LEARNED_CLASS_5
7   IF site35[1] THEN LEARNED_CLASS_5
    IF site35[2] THEN LEARNED_CLASS_1
9   ...
  }
```

Listing 7.2: Excerpt of an exemplary learned context classes model in CSM

7.2.1.2. Effect Model

The effect model in CSM is split into four sub-models. First of all, manufacturers provide descriptions of their SON functions in terms of mappings from possible SCV sets to KPI effects. Out of these initial effect models, a combined effect model can be generated (cf. Section 5.2.3.1) which is still based on the manufacturers' simulation environments. A real network effect model (cf. Section 6.2.3.3) reflects the actual impact of specific SCV sets in the environment where they are applied. However, some SCV sets will never be selected by the objective manager. This is the case if – according to the simulation-based combined effect model – they do not fulfill technical objectives sufficiently. This is exactly the problem: Since the combined effect model is based on an unrealistic environment, it is highly probable that the KPI predictions made in there do not hold true in a real network

environment. Some SCV sets that would actually perform well, are not selected, possibly preventing the SON functions from fulfilling operator objectives. Hence, a fourth model is needed that predicts the KPI effects of up-to-now untested SCV set combinations based on the effects of already tested SCV set combinations, i.e., the learned effect model.

Analogously to the context model, the two effect sub-models mentioned first are not described here in more detail since their structure and derivation is the same as depicted in previous approaches. However, the real network effect model is subject to little changes compared to the ASM approach. The structure and importance of the real network effect model and the learned effect model are described in the following, while the derivation of a learned effect model is part of Section 7.2.2.

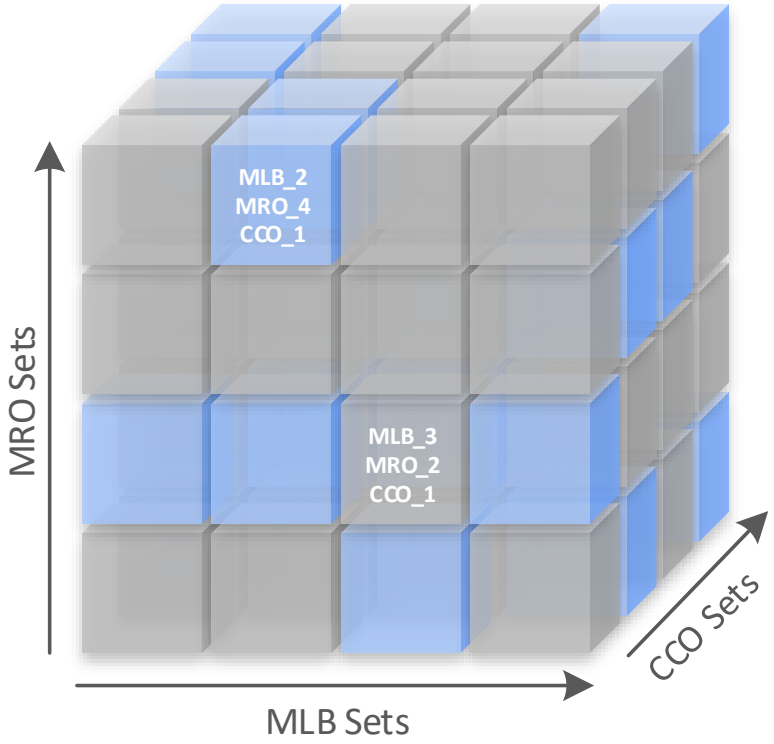


Figure 7.4.: Relationship between real network effect model and learned effect model in CSM, gray cubes representing KPI effect indications for tested combined SCV sets, blue cubes representing indications for untested combined SCV sets

In contrast to the combined effect model, these two models are strongly related to each other and hence, need to be examined together in the first place. Basically, the learned effect model fills the blanks in the real network effect model. While the real network effect model lacks of measurements for (usually) a lot of combined SCV sets, the learned effect model uses machine learning techniques to predict the KPI effects of these blanks. In order to make the aim of the learned

effect model more clear, Figure 7.4 visualizes the relationship between these two sub-models.

In Figure 7.4, each cube represents KPI effect indications for a particular combination of SCV sets. That is, three SON functions are considered with four SCV sets each. Note that this SCV set state space only visualizes the effects for one particular context class since the effect model in CSM is context-dependent as in ASM. The gray and blue shaded cubes together imply having measurements for each and every possible combined SON function combination. It has been already mentioned that it is highly improbable that all combinations are tested during operation and hence, optimization potential gets lost. The gray shaded cubes represent measurements that have been already taken, i.e., that are contained in the real network effect model, e.g., for σ_{321} in Figure 7.4. For SCV set combinations where no measurements are available in the real network effect model, e.g., σ_{241} in Figure 7.4, the learned effect model is able to provide effect predictions such that in the end, an effect model exists that is complete and that is entirely based on the real network environment.

Since both the real network effect model as well as the learned effect model can be seen as a union, they need to be based on the same context and have the same general structure.

```
IF learned context class THEN KPI effects
```

Listing 7.3: Effect model rule in CSM for real network and learned effect model

As can be seen, the IF part contains a learned context class which is a difference to the previous approach, the ASM. While this is quite obvious for the learned effect model, it needs further explanation with respect to the real network effect model.

Real Network Effect Model
First of all, it is worth mentioning that the real network effect model is still based on KPI measurements coming from a database that collects network measurements in terms of CM, PM and FM data. This database can be updated, e.g., at fixed time intervals or when a reconfiguration of SON functions takes place. It is important to know that these measurements are stored in their raw form, i.e., no aggregation per context class or SCV set is done within the database. This is crucial since each measurement can be assigned the respective cell by means of its operational context which is not possible any more when an aggregation of measurements has been performed. This way, measurements can be reused at any time and new aggregation methods can be applied which is necessary in CSM.

Before a learning algorithm can be reasonably applied, enough data about the network needs to be gathered and hence, a management approach such as ASM is adopted first. When enough data is collected, the MNO can apply learning

techniques, making a recalculation of the ASM's real network effect model inevitable due to a different definition of context classes. From that point on, effect predictions in the real network effect model also need to be based on the learned context classes model.

The process of deriving a real network effect model in CSM is similar to the one presented in Section 6.2.3.3 and is only described shortly here. In Equation 6.14, the set of taken measurements MM is defined as the set of measurements μ_σ^c for all applied conflict-free combined SCV sets $\sigma \in \Sigma_{\text{app}}^\Gamma$. These measurements are categorized per (manually defined) context class $c \in C$ and per combined SCV set $\sigma \in \Sigma_{\text{app}}^\Gamma$ such that they can be aggregated into one effect prediction $\mu_\sigma^c(\cdot)$ per context class and combined SCV set. This definition can be easily mapped to learned context classes.

In general, the model can be seen as a mapping function from SCV set combinations per learned context class $\tilde{\Psi}^{\tilde{c}} \in \tilde{\Psi}$ to aggregated measurements (per combined SCV set and learned context class) $\tilde{M}^{\tilde{c}} \in \tilde{M}$ with $\tilde{c} \in \tilde{C}$.

$$\text{EM}_{\text{real}_{\text{CSM}}} : \tilde{\Psi} \mapsto \tilde{M} \tag{7.8}$$

Thereby, $\tilde{\Psi}^{\tilde{c}}$ is defined as the set of already applied combined SCV sets under a certain learned context class and $\tilde{M}^{\tilde{c}}$ is defined as the set of aggregated measurements $\tilde{\mu}_\sigma^{\tilde{c}}$, i.e.:

$$\tilde{\Psi}^{\tilde{c}} = \left\{(\tilde{c}, \sigma) | \sigma \in \Sigma_{\text{app}}^\Gamma, \tilde{c} \in \tilde{C}\right\} \tag{7.9}$$

$$\tilde{M}^{\tilde{c}} = \left\{\tilde{\mu}_\sigma^{\tilde{c}}(\cdot) | \sigma \in \Sigma_{\text{app}}^\Gamma, \tilde{c} \in \tilde{C}\right\} \tag{7.10}$$

Note that each mapping, i.e., $(\tilde{c}, \sigma) \mapsto \tilde{\mu}_\sigma^{\tilde{c}}(\cdot)$ is exactly what is represented by the gray cubes in Figure 7.4.

Learned Effect Model

Referring back to the illustration in Figure 7.4, the blue cubes, i.e., learned effect indications, still need to be explained. In general, the learned effect model can be seen as a function mapping the non-tested combined SCV sets to the set of effect indicators that are not part of the real network effect model, i.e.:

$$\text{EM}_{\text{learned}} : \Psi' \mapsto \Phi' \tag{7.11}$$

Thereby, Ψ' consists of $\Psi^{\tilde{c}'}$, i.e., $\Psi^{\tilde{c}'} \in \Psi'$ and $\Phi^{\tilde{c}'} \in \Phi'$ for all $\tilde{c} \in \tilde{C}$.

$$\Psi^{\tilde{c}'} = \left\{(\tilde{c}, \sigma) | \sigma \in \Sigma^\Gamma \setminus \Sigma_{\text{app}}^\Gamma, \tilde{c} \in \tilde{C}\right\} \tag{7.12}$$

$$\Phi^{\tilde{c}'} = \left\{\tilde{\varphi}_\sigma^{\tilde{c}}(\cdot)' | \sigma \in \Sigma^\Gamma \setminus \Sigma_{\text{app}}^\Gamma, \tilde{c} \in \tilde{C}\right\} \tag{7.13}$$

In other words, each blue cube defines a mapping $(\tilde{c}, \sigma) \mapsto \tilde{\varphi}_{\sigma}^{\tilde{c}}(\cdot)'$ with $\tilde{\varphi}_{\sigma}^{\tilde{c}}(\cdot)'$ being the prediction in terms of all observed KPIs for a certain combined SCV set σ and a learned context class $\tilde{c}$.

Combining the real network effect model and the learned effect model leads to an effect prediction for each combination of SCV sets in each learned context class, facilitating the elicitation of the best of all possible SCV sets. While the definition of a learned effect model is quite simple, the derivation of $\tilde{\varphi}_{\sigma}^{\tilde{c}}(\cdot)'$ is the actual crux, since this is where machine learning algorithms come into play.

7.2.2. Derivation of Learned Models

For the derivation of learned input models it is essential to understand the models' structure first which has been explained in the previous section. This is due to the fact that the learned context classes are determined based on the effect predictions in the real network effect model as well as the learned effect model, and the real network effect model is generated by aggregating KPI measurements for each combined SCV set and, more important, for each learned context class. One may think that this is a cyclic dependency, but it is crucial to know that the time step at which the respective model has been generated, is essential.

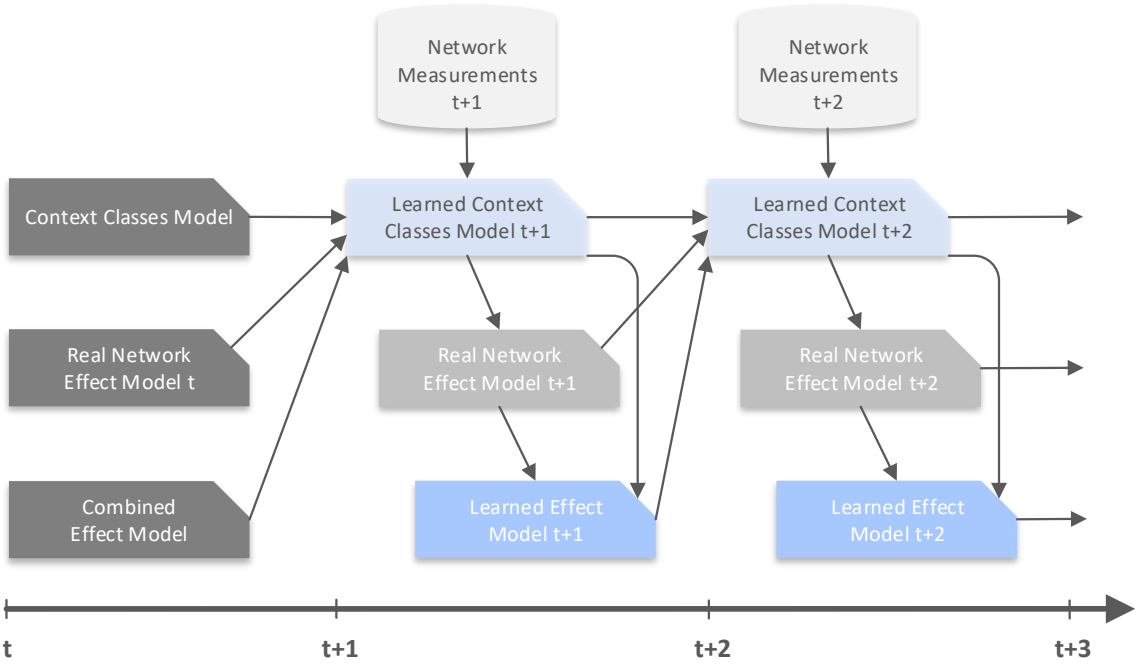


Figure 7.5.: Dependencies between learned context model, real network effect model and learned effect model in different time steps

Figure 7.5 shows the dependencies between the models for three time steps t, t+1 and t+2 whereby a time step is a point in time where the recalculation of

models becomes necessary. This may be at fixed time intervals or at any time new measurements are available due to a reconfiguration of SON functions. In this illustration, the CSM is activated at step t+1 and hence, the learned models are generated here for the first time. It has already been said that the models in CSM are dependent from each other. Since t+1 is the first time step with an active CSM, no real network effect model based on learned context classes and no learned effect model are available. Therefore, besides using measurements from the database, the clustering of cells is done using effect predictions in the combined effect model and in the real network effect model based on manual context classes. The necessity of using the combined effect model is further described in Section 7.2.2.1. Descriptions of effect predictions in the combined effect model are based on the (manual) context classes model and consequently, this model also serves as initial input to the learned context model at t+1 in order to be able to relate KPI effects to the respective cells.

From t+1 on, the generation of models is a sequential process according to the directions of the arrows in Figure 7.5. That is, based on the measurements database, a learned context classes model is generated first, since the real network effect model as well as the learned effect model make use of the learned context classes. The second iteration comprises the aggregation of KPI measurements according to these learned context classes, i.e., the generation of a real network effect model. Finally, the learned effect model can be determined using resulting models of the previous iterations. In t+2, the learned context classes model is then generated by predominantly clustering cells according to measurements in the database and in addition, effects in real network effect model t+1 and learned effect model t+1 while the second and third iteration are performed in the same way as described above.

Given this declaration, it can be said that the clustering of cells and the KPI predictions for untested SCV set combinations are always done using the most current models. With this background information, the details of deriving learned models can be demonstrated.

7.2.2.1. Learned Context Classes Model

In Section 7.2.1.1 it has been described that the learned context classes model describes mappings from cells to classes in the end. This section aims at describing the assignment to a particular context class. Therefore, the set of cells $\omega \in \Omega$ is split into groups of cells, i.e., clusters, with similar behavior in terms of network KPIs. In the end, each cluster becomes what is called a learned context class and hence, all clusters together build the learned context classes model. This is a typical problem for unsupervised learning algorithms such as K-Means clustering. In order to not confuse the K in K-Means with the set of KPIs K, it is denoted as k_{means} in the following.

Data Preparation

Before utilizing a clustering algorithm, the available data needs to be prepared first. Since the clustering shall be done based on KPI values, the effect predictions for all combined SCV sets are required. This opens up two problems: First, the measurements database is not complete usually, i.e., not all combined SCV sets have already been tested in the network yet. Second, since cells are assigned to different context classes, the set of applied combined SCV sets $\Sigma_{\text{app}}^{c_i}$ (or $\Sigma_{\text{app}}^{\tilde{c}_i}$ respectively) is usually different for all manual context classes $c_i \in C$ (or learned context classes $\tilde{c}_i \in \tilde{C}$).

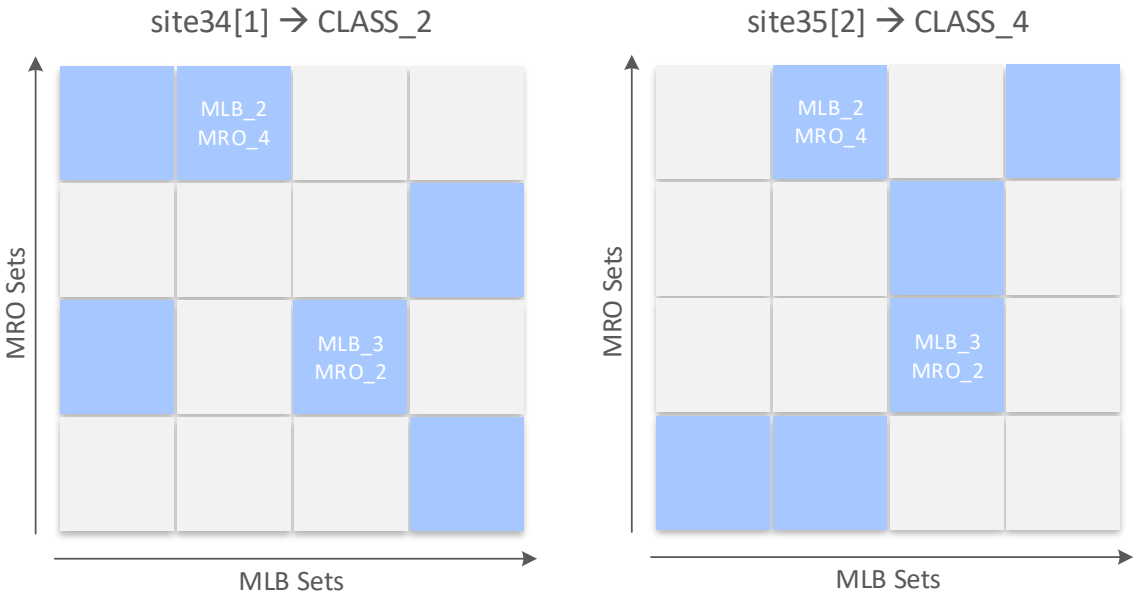


Figure 7.6.: Exemplary illustration of available data for two cells in different context classes

In Figure 7.6, this problem is exemplary illustrated by means of two cells *site34[1]* and *site35[2]* which are assigned to different context classes, and two SON functions MLB and MRO. Blue squares represent already tested combined SCV sets, i.e., where measurements are available, whereas gray squares represent untested SCV sets. One can see that there is only an overlap of two SCV sets, namely *(MLB_2, MRO_4)* and *(MLB_3, MRO_2)*. It is obvious that this overlap gets smaller with an increasing amount of network cells and context classes. As a consequence, the data basis for each of the cells diverges immensely making the application of a clustering algorithm impossible.

This is where the real network effect model and (depending on the time step, cf. Figure 7.5) either the combined effect model or the learned effect model comes into play. In Figure 7.4 it could be seen that the real network effect model in combination with the learned effect model builds a complete effect model with predictions for all possible combined SCV sets. The same applies for the real network effect model in combination with the combined effect model. However, in

the second case, only those predictions are taken from the combined effect model (note that it is already complete itself) where no measurements are available in the real network effect model. The result is a complete set of effect predictions for each cell in the network which serves as input for the application of the K-Means algorithm.

In order to make this more clear, the data set $\tilde{M}^{\omega}$ for each cell ω can be represented as a mapping from a multi-dimensional matrix of combined SCV sets $\sigma \in \Sigma^{\Gamma}$ to KPI effects $\tilde{\mu}^{\omega}_{\sigma}$. For two SON function, i.e., two dimensions, this matrix looks as follows:

$$\tilde{M}^{\omega} = \begin{bmatrix} \sigma_{1j} & \sigma_{2j} & \cdots & \sigma_{ij} \\ \vdots & \vdots & \iddots & \vdots \\ \sigma_{12} & \sigma_{22} & \cdots & \sigma_{i2} \\ \sigma_{11} & \sigma_{21} & \cdots & \sigma_{i1} \end{bmatrix} \mapsto \begin{bmatrix} \tilde{\mu}^{\omega}_{\sigma_{1j}}(\cdot) & \tilde{\mu}^{\omega}_{\sigma_{2j}}(\cdot) & \cdots & \tilde{\mu}^{\omega}_{\sigma_{ij}}(\cdot) \\ \vdots & \vdots & \iddots & \vdots \\ \tilde{\mu}^{\omega}_{\sigma_{12}}(\cdot) & \tilde{\mu}^{\omega}_{\sigma_{22}}(\cdot) & \cdots & \tilde{\mu}^{\omega}_{\sigma_{2j}}(\cdot) \\ \tilde{\mu}^{\omega}_{\sigma_{11}}(\cdot) & \tilde{\mu}^{\omega}_{\sigma_{21}}(\cdot) & \cdots & \tilde{\mu}^{\omega}_{\sigma_{1j}}(\cdot) \end{bmatrix} \tag{7.14}$$

Thereby, i is the number of possible SCV sets for the first SON function and j the number of SCV sets for the second SON function. Consequently, the whole set of measurements for all cells can be described as:

$$\tilde{\mathbf{M}}^{\Omega} = \{\tilde{M}^{\omega} | \omega \in \Omega\} \tag{7.15}$$

Clustering of Cells

Now that a common data base is available for all cells, they can be clustered into similarly behaving cells. Therefore, the k_{means} needs to be chosen first. In Section 2.3.2, it has been said that the k_{means} needs to be determined manually, which, in case of CSM, is the MNO. Since there are $|\Omega|$ cells in the network, the maximum value is $k_{\text{means}} = |\Omega|$. Both, choosing k_{means} too low and too high, has several advantages and disadvantages: The lower the k_{means} value, the more cell effects are aggregated which may lead to a very inaccurate effect prediction in the effect model and thus, to a quite uniform configuration of SON function instances in the network. On the other side, the computational effort is quite low since only combined SCV sets for a small number of context classes need to be calculated. Vice versa, the higher the k_{means} value, the more accurate are effect predictions since they are done more cell-based. It is obvious that the disadvantage of a higher k_{means} value is the increasing computational effort. Hence, one has to balance the k_{means} in a meaningful way which is discussed in Chapter 8.

Having chosen the k_{means}, the data set $\tilde{\mathbf{M}}^{\Omega}$ can be fed into a K-Means algorithm such as the one presented in Algorithm 1. Note that each combined SCV set thereby represents an own feature, resulting in a multi-dimensional state space to be clustered. At this point, it becomes clear that it is impossible for a human operator to capture the behavior of all cells in the network under each and every SCV set, resulting in an erroneous manual context classes model with the utmost probability and making the automated generation of a learned context classes model inevitable. Based on the features and the related values for these features,

the algorithm tries to find k_{means} clusters where cells behave similar. That is, taking all combined SCV sets into account, cells in one cluster show a more homogeneous behavior in terms of KPI effects than all cells outside of this cluster. This learned context model then serves as input for the generation of a real network effect model in CSM, i.e., measurements are aggregated within each learned cell cluster. Both models are used for the generation of a learned effect model which is explained in the following.

7.2.2.2. Learned Effect Model

The learned effect model mainly has the purpose to fill the blanks in the real network effect model as described in Section 7.2.1.2. That is, KPI effects should be reliably estimated for combined SCV sets where no measurements are available in the real network effect model. However, for a variety of combined SCV sets, measurements exist in the real network effect model. Since this is labeled data, i.e., the true output for a given input is known, this is a problem for supervised learning algorithms such as LR, GPR, KNN or ANN.

Data Preparation
Before beginning with the creation of machine learning models, the data need to be in a certain shape such that a learning algorithm can be applied in a meaningful way. Therefore, the KPI measurements database and the learned context classes model serve as input for the creation of a real network effect model which makes up the data basis for training and testing.

Table 7.1.: Exemplary raw measurements in KPI measurements database in CSM

ID	DATE	CELL	MLB	MRO	DCR	HOSR	CL
...	...	...	...	...	...	...	...
345	19-12-2018 07:11:19	site34[1]	1	2	0.013	0.944	0.495
346	19-12-2018 07:24:33	site07[0]	1	2	0.017	0.932	0.510
347	19-12-2018 07:07:26	site14[1]	3	2	0.024	0.971	0.529
348	19-12-2018 08:03:01	site28[2]	3	2	0.032	0.983	0.432
349	19-12-2018 07:58:45	site34[1]	3	1	0.025	0.893	0.566
...	...	...	...	...	...	...	...

In general, measurements in the database consist of a set of attributes as depicted in Table 7.1. The DATE at which they were measured is important to apply an aging process as explained in the ASM approach. By means of the CELL, measurements can be assigned to a learned context class and aggregated within this class. The table further contains one column for each SON function, storing the SCV set number at the moment of taking the measurement. Measurements are aggregated for each single combination of these SCV sets, i.e., the KPI values stored in the last three columns are averaged according to a certain metric (e.g.,

the aging process). Note that an arbitrary number of KPIs can be observed and an arbitrary number of SON functions can be deployed in the network. Further note that the SCV set numbers represent placeholders for a set of actual SCP values.

It may be not obvious why it is necessary to average the measurements within the learned context classes and, within each class, for each combined SCV set. This has mainly two reasons: First of all, the computational effort for doing cell-based predictions would be fairly high. Imagining a network with hundreds and thousands of cells, the calculations would last too long for SON functions acting in short time scales, i.e., a few minutes or even seconds, such as, e.g., MLB. Hence, the learned context classes model reduces this complexity by defining a manageable number of context classes. However, note that the number of learned classes can be set by the operator itself, meaning it can also be defined $k_{\text{means}} = |\Omega|$ in case the MNO does not want to average over more than one cell. The second reason is that the KPI values for a single cell may differ substantially, i.e., they have a large variance within single combined SCV sets. Due to a possibly wide interval of measured KPI values, the irreducible error would always be very large, therefore making accurate predictions for a single cell hard. However, it is not the goal of the learned effect model to predict the exact values for a certain point in time, but to estimate the overall, i.e., the average performance of up-to-now untested combined SCV sets.

Construction of a Regression Model
After dealing with the data preparation in terms of KPI measurements in the previous paragraph, the goal now is to model and describe the causal relationship between the SCV sets and the observed KPIs. That is, the input (or features) of the model consists of a set of SCVs for each deployed SON function. The output variables are the observed KPIs. One model needs to be constructed for each particular KPI since the algorithms that are in the focus of this thesis can only predict one output variable and optimal solutions may differ for different KPIs. However, there are several multivariate regression approaches being able to predict outputs for more than one variable. Note that in this paragraph, the general approach how to construct a regression model is described, applicable for all types of regression algorithms. Four algorithms are undertaken a closer look, namely LR, GPR, KNN and ANN. Their possibilities with respect to the learned effect model are described in this paragraph while the selection of an appropriate algorithm is part of Chapter 8.

Before building the actual model it is worth investigating the premises defined by the SON management approach, in order to avoid common pitfalls and misinterpretations of the results [Lee15]. A fundamental premise is the assumption of *correlation implying causality*. As a mobile network is a controlled environment and the only factors changed are the independent variables of the combined SCV sets, an actual causal relationship between SCV sets and observed average KPIs can be assumed. The next pitfall is presented through *over-fitting*. To avoid overfitting, the data samples, i.e., the combined SCV sets with related KPI effects in

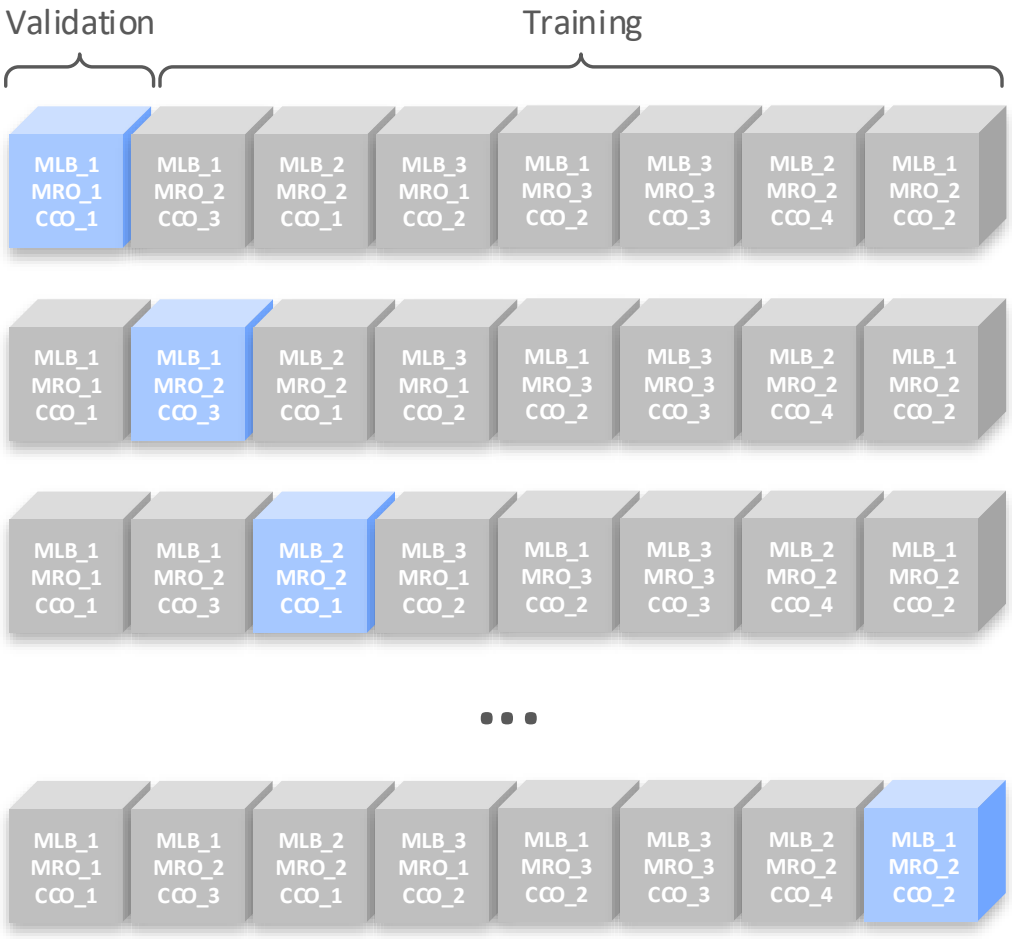


Figure 7.7.: Exemplary LOO cross validation on eight different combined SCV sets

the real network effect model, are split into a train and a test set with a relation of 2 : 1 for each learned context class, thereby ensuring that the model selection happens on a different set of data than the model evaluation. In order to answer the question how well the performance of untested SCV sets can be predicted, it is necessary to split the larger group of combined SCV sets into a training and a validation set. Therefore, LOO cross validation is employed which is illustrated in Figure 7.7.

Let n be the amount of combined SCV sets (i.e, $n = 8$ in Figure 7.7). Then, the model is trained on $n - 1$ SCV sets (the gray cubes in Figure 7.7) and asked to predict the respective KPI of the nth SCV set (the blue cubes in Figure 7.7). This step is repeated for each combined SCV set, therefore leading to n slightly different models and n residuals. On these residuals the error metrics, e.g., RMSE, can be calculated describing how well one type of model is able to predict the KPI of unseen SCV sets. Note that LOO cross validation is only reasonably applicable when having a relatively small dataset, i.e., when having only a few tested distinct SCV sets. Assuming a larger dataset, a method such as k-fold cross vali-

dation with a lower k is more meaningful. The advantages and disadvantages of different resampling methods have already been discussed in Section 2.3.1.

Comparing the error values of the different resulting models, the model with the lowest error can be selected since this is the one which best represents the behavior of the network over all applied SCV sets and hence, is likely the best to predict the effects of untested combined SCV set combinations. While this error value is important for the selection of a regression model, it is also valuable feedback for an MNO. Based on this value, which perfectly represents the quality of the calculated model, an MNO can decide whether the resulting learned effect model is trustworthy enough to be applied in the real network or not.

For the four algorithms mentioned above, the construction of a regression model is presented in the following. Thereby, it starts with LR which is investigated in case the SCV sets and their effects show a linear relation. GPR and KNN have been selected in case that the KPI values follow a more chaotic pattern. Finally, ANNs are investigated due to their wide scope of application.

Linear Regression The general form of an LR model has been shown in Equation 2.9. As explained in Section 2.3.1.1, the LR model can be made increasingly more flexible by extending the input vector with polynomials and combinations of the existing features. The potential drawbacks of an overly flexible model are illustrated in Section 2.3.1. The SON functions can then be fitted with different models mapping an s^f to the respective KPI. The flexibility can be stepwise increased from a simple linear model up to a more flexible model with higher polynomials. For instance, Equation 7.16 shows a linear model for estimating the value for a KPI DCR k^{DCR} based on the SCV sets of two SON functions MLB and MRO. In contrast, Equation 7.17 shows a cubic model with cubic interaction terms.

$$k^{\mathrm{DCR}} = \beta_0 + \beta_1 * s^{\mathrm{MLB}} + \beta_2 * s^{\mathrm{MRO}} \tag{7.16}$$

$$\begin{aligned} k^{\mathrm{DCR}} = {} & \beta_0 + \beta_1 * s^{\mathrm{MLB}} + \beta_2 * s^{\mathrm{MRO}} + \beta_3 * s^{\mathrm{MLB}} * s^{\mathrm{MRO}} \\ & + \beta_4 * (s^{\mathrm{MLB}})^2 + \beta_5 * (s^{\mathrm{MRO}})^2 + \beta_6 * (s^{\mathrm{MLB}} * s^{\mathrm{MRO}})^2 \\ & + \beta_7 * (s^{\mathrm{MLB}})^3 + \beta_8 * (s^{\mathrm{MRO}})^3 + \beta_9 * (s^{\mathrm{MLB}} * s^{\mathrm{MRO}})^3 \end{aligned} \tag{7.17}$$

Gaussian Process Regression Within this method, the crux lies in choosing the kernel function that measures the similarity between two instances x_i and x_j. Only kernel functions that can be used to predict real-valued data are relevant in this thesis since each KPI takes on a real value. Hence, the RBF kernel, the polynomial kernel and the PUK kernel are further investigated. For instance, measuring the similarity between two combined SCV sets in the RBF kernel is calculated as follows for two SON functions, their combined SCV set number represented by i and j:

$$\kappa_{RBF}(\sigma_i, \sigma_j) = exp\left(-\frac{||\sigma_i - \sigma_j||^2}{2\lambda}\right) \tag{7.18}$$

with σ_i and σ_j being vectors of SCV sets, one value for each SON function.

k-nearest Neighbors Regression In the KNN algorithm, the crucial part is the choice of an appropriate k, i.e., the k closest instances to a given instance *inst* approximate the output for *inst* as defined in Equation 2.15. Therefore, the similarity is calculated using a distance measure such as the euclidean distance. Thereby, the difference between two distances is defined by comparing the different features, i.e., the particular SCV sets, with each other. Equation 7.19 shows the distance calculation for two combined SCV sets σ_{ij} and σ_{mn}, the first variable (i, m) representing the MLB set and the second one (j, n) the MRO set:

$$d_{euc}(\sigma_{ij}, \sigma_{mn}) = \sqrt{\left(s_i^{\text{MLB}} - s_m^{\text{MLB}}\right)^2 + \left(s_j^{\text{MRO}} - s_n^{\text{MRO}}\right)^2} \tag{7.19}$$

with s_i^{MLB}, s_m^{MLB}, s_j^{MRO} and s_n^{MRO} representing the respective SCV set number.

Artifical Neural Networks In neural networks, choosing the number of hidden layers and the number of neurons per layer is the crucial part. In Section 2.3.1.4, some guidelines are presented how to decide about these two points without confronting the problems of under- or over-fitting. Since it is proposed to never choose the number of hidden layers > 2, a suitable model should have one or two hidden layers. For the number of hidden neurons, one should have a look at the features (= number of neurons on the input layer) and the values to be predicted (= number of neurons on the output layer). Since each SCP of each SON function $f \in F$ marks a feature, the number of hidden neurons on each layer should be $\leq 2 * \sum_{f \in F} |\text{par}^{\text{f}}|$ following the third rule in Section 2.3.1.4 that determines the highest number of neurons. Since the prediction is done for each KPI individually, there is always one parameter to be predicted, i.e., one neuron on the output layer. Following the first rule in Section 2.3.1.4, each hidden layer must have at least two hidden neurons then. Summarized, the number of hidden neurons n for the presented problem should be $2 \leq n \leq 2 * \sum_{f \in F} |\text{par}^{\text{f}}|$. For instance, having three SON functions deployed in the network with two SCPs each, the number of hidden neurons on each hidden layer should be between 2 and 12.

Prediction of KPI Effects

The crucial part in the process of deriving a learned effect model is the construction of a regression model that reflects the SON functions' behavior good enough to be satisfactory for the estimation of KPI effects of untested SCV sets. This step has been described in the previous paragraph. Possibilities of determining a suitable model for four algorithms are presented. The selected model then represents the analyzed data in the best possible way and can be used for the prediction of unseen, i.e., untested combined SCV sets. Therefore, the set of untested combined

SCV sets needs to be determined first which is defined as all combinations that do not lead to a conflict and that have never been applied in the real network, i.e.:

$$\Sigma_{\text{untested}} = \Sigma^{\Gamma} \setminus \Sigma^{\Gamma}_{\text{app}} \tag{7.20}$$

For each $\sigma \in \Sigma_{\text{untested}}$, the respective regression model is applied for each of the observed KPIs. Note that thereby, the selected model may be different for different KPIs. Further note that this is done for each of the learned context classes individually. Hence, a mapping is defined for each combined SCV set under a certain learned context class to a value for a KPI $k \in K$, i.e., $(\tilde{c}, \sigma) \mapsto \varphi^{\tilde{c}}_{\sigma}(k)'$.

Referring back to Section 7.2.1.2, the goal was to define the $(\tilde{c}, \sigma) \mapsto \varphi^{\tilde{c}}_{\sigma}(\cdot)'$ which then can be easily achieved by combining the $(\tilde{c}, \sigma) \mapsto \varphi^{\tilde{c}}_{\sigma}(k)'$s for all $k \in K$. The set of all these mappings for all learned context classes and all combined SCV sets then results in the learned effect model $\text{EM}_{\text{learned}}$.

7.2.3. Methodology

Before beginning with an explanation of details of the developed methodology, an overview is provided about all models that are relevant for a CSM system. The relationship between these models and their usage in CSM is explained afterwards. This overview does not only include input models which are externally provided, i.e., by an operator or SON function manufacturers, but also models that are internally generated by the objective manager.

Objective Model In this model, an MNO expresses his or her targets in terms of network KPIs. A KPI target is a concrete value that should be achieved and a weight is assigned to each target such that it is able to trade them off against each other. Together, they form an objective which is context-dependent, meaning there needs to be an objective for each KPI in each (manually defined) context class.

Context Attributes Model The cells and the environment where the cells operate in, is described by context properties. These context properties, together with their possible value range, are stored in the context attributes model and serve as input for the definition of manual context classes.

Manual Context Classes Model This model is manually defined by the operator of a mobile network for the purpose of reducing the huge context state space to manageable context classes. That is, a context class summarizes context states where it is assumed that cells have a similar behavior and hence, there is exactly one objective per KPI per manual context class in the objective model.

Manufacturer Effect Model SON functions are usually delivered as black boxes by the manufacturers and hence, a description of their behavior needs to be provided in the first place. The manufacturer effect model maps a set of SCV sets for a SON function to KPI effects, i.e., concrete KPI values, expected to be achieved in the network.

Combined Effect Model Manufacturer effect models are delivered per SON function and hence, do not indicate the effect of having a variety of SON functions concurrently being active in the network. The objective manager performs a reasoning process to combine the KPI effect prediction provided by different SON function manufacturers. The result is an effect model that maps a combined SCV set to the expected effects on all observed KPIs.

Real Network Effect Model Both, the manufacturer and the combined effect model, are based on simulations done in the network environment of the SON function manufacturer. When deploying SON functions in the real network, their behavior usually differs from what is predicted in these models. Therefore, measurements are taken and aggregated to generate a real network effect model that better reflects the actual SON function behavior. While these measurements are always cell-based, the aggregated effects in the model are based on either manual or learned context classes.

Learned Context Classes Model Since the classification of cells in the manual context classes model is done by the operator on inspection, it is highly probable that some of the cells are wrongly classified. The learned context classes model clusters cells based on their behavior, i.e., based on the KPI values they produce under certain SCV sets, resulting in mappings from cells to learned context classes.

Learned Effect Model It is very unlikely that all available SCV set combinations can be tested in the real network since the objective manager always selects combinations with a promising expected utility. However, effect predictions in the effect models may be inaccurate and hence, some untested combinations of SCV sets may lead to an unexpectedly well performance. Effect predictions in the learned effect model complete the real network effect model by using machine learning techniques to estimate the performance of untested SCV set combinations based on the real network effect model.

Three models have been explained in more detail in this chapter: The learned context classes model, the real network effect model based on learned context classes and the learned effect model. However, these models can only be generated when enough knowledge about the network in terms of KPI measurements is gathered and as a consequence, all of these models are necessary for an effective CSM approach. More precisely, when enabling SON management for the first time in a mobile network, the ODSM approach serves as a starting point. Within this *configuration testing phase*, as much different combined SCV sets as possible should be tested. In the second phase, i.e, the *data collection phase*, the ASM approach is applied. Hereby, measurements are continuously taken to gain knowledge about

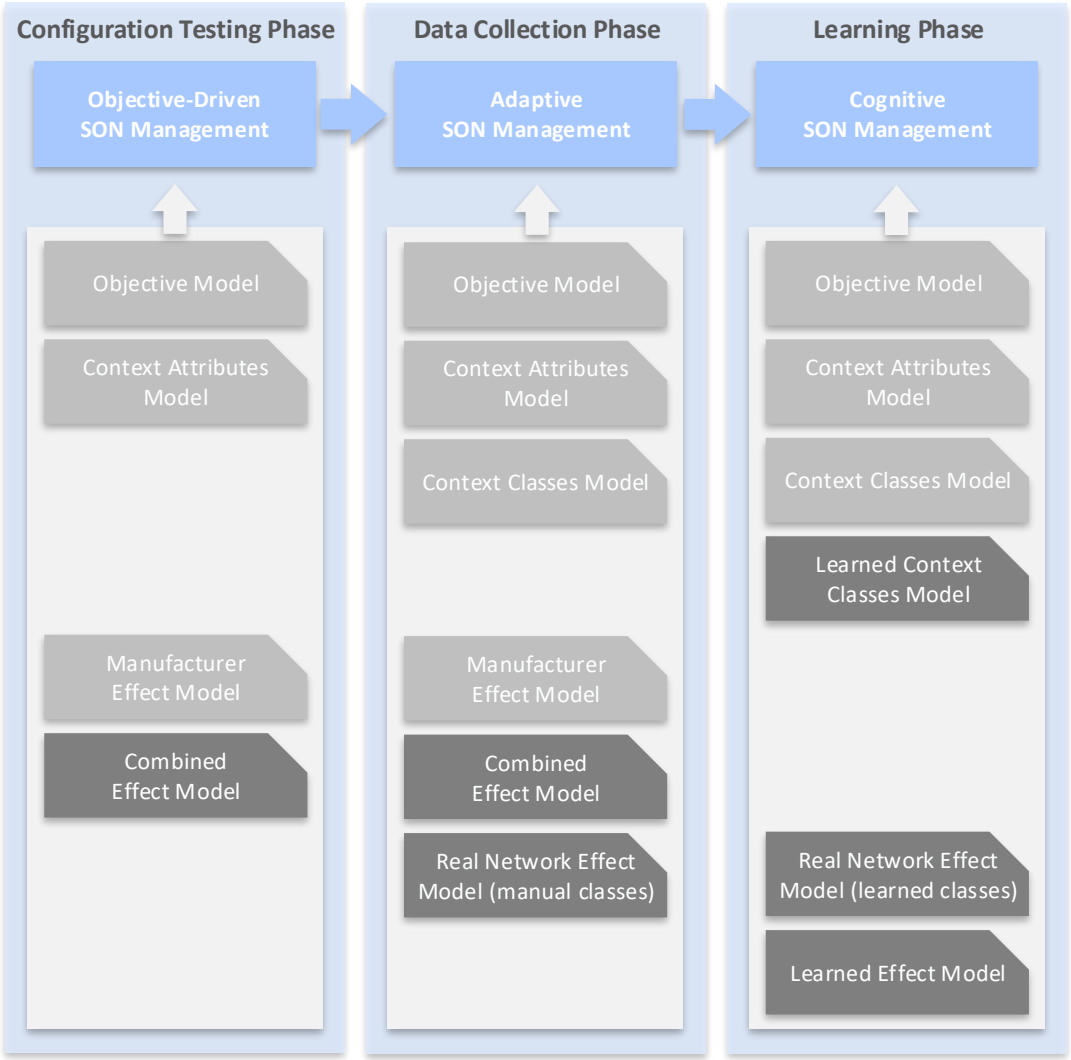


Figure 7.8.: Overview of CSM phases and models used in them

already tested SCV sets. In the third and last phase, the optimum in terms of KPI target fulfillment can be achieved by using the CSM approach. This *learning phase* helps to understand the behavior of all possible SCV set combinations, even the undocumented ones. An overview of these three phases, the used SON management approach and used models is depicted in Figure 7.8 and will be further explained in the following, thereby focusing on the learning phase and the methodology of the CSM approach.

Externally provided models are shaded in light gray, models generated by the objective manager are shaded in dark gray. Note that the real network effect model is based on different context classes in the ASM and CSM approach.

7.2.3.1. Configuration Testing Phase

It is not the goal of this thesis to describe a SON management approach that is already installed in the mobile network for a while, but also give an idea how SON management could deal with initial network conditions. That is, no measurements or any other knowledge about the network is existent when deploying a SON management system for the first time. The only available knowledge is the description of technical objectives, the network's attributes and the manufacturer-provided effect models. ODSM uses this knowledge to first combine the effect models for the different SON functions into a combined effect model. Afterwards, the combined SCV set with the highest overall utility is selected and deployed in the network. Various SCV set combinations usually provide the same (optimal) utility such that they should be chosen randomly to test as much SCV sets as possible. In doing so, information about several SCV set combinations can be gathered. One may think that the random choice of SCV sets might be problematic for an MNO. However, all these combined SCV sets provide the same utility, i.e., it is expected that they all fulfill operator objectives to the same degree. The selection of SCV sets according to the ODSM approach has already been described in Chapter 5 and is not illustrated in further detail here.

It remains the operator's decision to stop the configuration testing phase and to turn into the data collection phase. This may have two reasons: First, the number of tested SCV sets is sufficient for an operator and for each situation there is an SCV set where KPI targets are sufficiently fulfilled. However, this is highly unrealistic due to the fact that all calculations are based on unrealistic input effect models. Hence, a second reason may be that some of the SCV sets do not achieve the expected results and hence, the network is jumping between good and bad SCV sets.

7.2.3.2. Data Collection Phase

To overcome the above mentioned problem, ASM collects data about the network in terms of KPI measurements and builds a real network effect model that better reflects the actual behavior of SCV sets in the network. This is done by permanently integrating the most current KPI measurements in the existing real network effect model and weighting newer measurements higher than older ones. This way, the effect model gets closer to reality step by step. For the selection of optimal SCV sets it is now meaningful to not only choose the combination with the highest utility, but the one which provides the best absolute KPI target fulfillment. This can be achieved by measuring the distance as described in Section 6.2.4.1. In general, the selection of combined SCV sets works as presented in Chapter 6 and is not further explained here.

It is again the operator's decision to stop the data collection phase and turn into the learning phase. This may have two reasons: First, enough data has been collected such that the real network effect model perfectly describes the SON functions' behavior and, as a consequence, applying machine learning techniques becomes meaningful. Second, the network may be in a state where none of the already tested SCV sets achieves the operator's objectives. This can be the case when the effect predictions in manufacturer-provided effect models do not prove true at all in the real network.

Note that it is also possible to switch back to the configuration testing phase at any time in case that more SCV sets should be tested in the real network.

7.2.3.3. Learning Phase

In the learning phase, CSM is activated, meaning that a complete effect model is available at any time. The learning process is initiated every time new measurements are taken such that the learned effect model calculations are always based on the most current information about the mobile network, i.e., the real network effect model is permanently adapted. Both, the real network effect model as well as the learned effect model, are thereby dependent on learned context classes. This is a difference to the ASM approach where the effect model depends on the manually defined context classes. However, the learned context classes describe similarities between cells in a better way since the clustering is done based on the actual KPI values produced by the cells. The manual context classes model is still used for the definition of operator objectives since the MNO best knows what goals should be achieved in the network. The general CSM methodology for selecting appropriate SCV sets is still based on the ASM approach with slight changes in the different calculation steps which is presented in the following.

Figure 7.9 shows the algorithm for deriving an SCV set policy in CSM. The first step includes the generation of a learned context classes model which has been already described in Section 7.2.2.1. The second step is the generation of a real network effect model based on the learned context classes. The details of creating this model can be found in Section 7.2.1.2. Based on these two models, the learned effect model can be generated which is explained in detail in Section 7.2.2.2. The selection of suitable SCV sets is based on these three models as well as the objective model and the manual context classes model.

SCV Set Calculation
The calculation of suitable combined SCV sets is similar to the approach presented in the ASM chapter. Five consecutive steps need to be executed, namely the *delta determination*, the *SCV set filtering*, the *KPI target vs. network performance comparison*, the *performance indication* and finally, the *SCV set selection*. One of the main differences to ASM is that this is done for each cell (instead of manual context class) in the network which has mainly two reasons: First of all, this allows

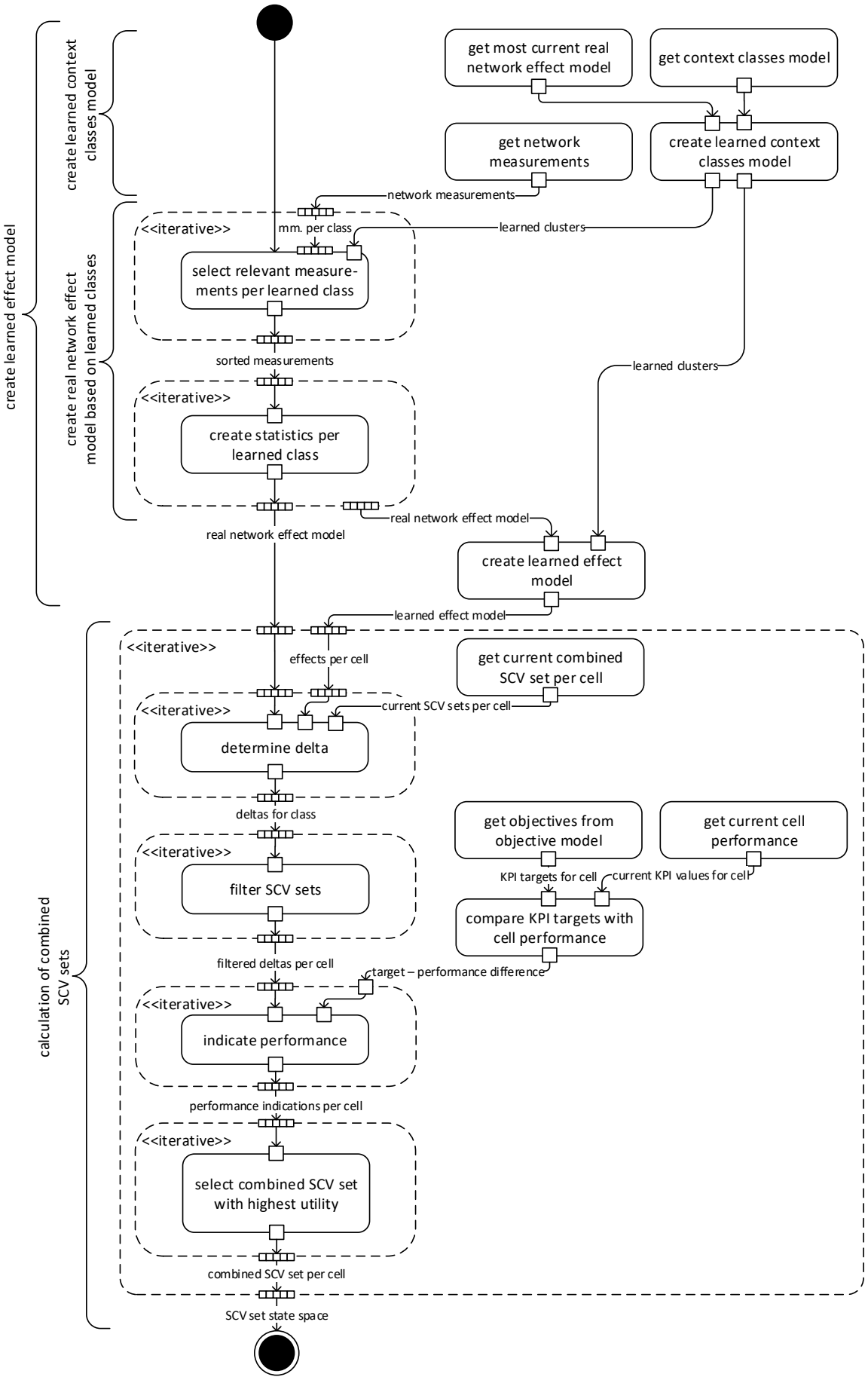


Figure 7.9.: SCV set policy derivation algorithm in CSM (noted in UML 2)

for a more precise and tailored configuration of the SON function instances deployed on each cell. Second, since the KPI targets are defined per manual context class and the effect models are dependent on learned context classes, a mapping between these two types of context models would be too complex and erroneous. Note that the cell-based configuration does not have a tremendous impact on the computational effort. The calculations done in here are simple arithmetic operations such that even a thousandfold number of them does not have a noticeable effect with respect to calculation time. The five steps as they appear in CSM are one by one described in the following while focusing on the differences to ASM.

Delta Determination In this step, the delta $\tilde{\delta}_\omega$ between the expected network performance, i.e., the effect prediction for the currently active SCV set combination σ_{current}, and all other effect predictions in the real network effect model EM_{real} as well as learned effect model $\text{EM}_{\text{learned}}$, is calculated. This is done for each cell $\omega \in \Omega$ individually. Note that σ_{current} is definitely in EM_{real} since this SCV set combination has been already deployed in the real network.

$$\tilde{\delta}_\omega(\sigma_i, \sigma_{\text{current}}) = \tilde{\mu}^{\tilde{c}}_{\sigma_i}(\cdot) - \tilde{\mu}^{\tilde{c}}_{\sigma_{\text{current}}}(\cdot) \tag{7.21}$$

with $\tilde{c} \in \tilde{C}$ and $(\omega \mapsto \tilde{c}) \in \text{CM}_{\text{learned}}$. Thereby, $\sigma_i \in \Sigma_{\text{app}} \cup \Sigma_{\text{untested}}$, meaning the effect prediction is either in the real network or learned effect model. By calculating this delta, the objective manager gets an impression about the impact in the network when changing from the current combined SCV set to every other possible SCV set combination. Hence, the result of this step is a set $\tilde{\Delta}$ of $\tilde{\delta}$s for each cell $\omega \in \Omega$, i.e.:

$$\tilde{\Delta}_{\tilde{c}} = \left\{ \tilde{\delta}_\omega(\sigma_i, \sigma_{\text{current}}) | \sigma_{\text{current}} \in \Sigma^\Gamma_{\text{app}}, \sigma_i \in \Sigma^\Gamma_{\text{app}} \cup \Sigma^\Gamma_{\text{untested}}, \tilde{c} \in \tilde{C} \right\} \tag{7.22}$$

with $|\tilde{\Delta}_{\tilde{c}}| = |\Sigma^\Gamma_{\text{app}} \cup \Sigma^\Gamma_{\text{untested}}|$.

SCV Set Filtering The second step of the objective manager's algorithm is very similar to the one in ASM. That is, the list of possible SCV sets and hence, the set of δs is filtered for SCV sets that could have a better effect on the KPIs of a cell than the currently deployed SCV set. Therefore, the $\tilde{\delta}_\omega$s are multiplied with a scaling vector as defined in Equation 6.24, i.e.:

$$\tilde{\delta}^*_\omega(\sigma_i, \sigma_{\text{current}}) = \tilde{\delta}_\omega(\sigma_i, \sigma_{\text{current}}) \cdot \varsigma \tag{7.23}$$

The $\tilde{\delta}^*_\omega$s that contain negative normalized delta values for all KPIs, can be discarded since they would definitely worsen the performance of that cell. One can easily translate the function defined in Equation 6.27 into a function $\text{filter}(\delta^*_\omega(\sigma_i, \sigma_{\text{current}}))$ that filters the cell-specific normalized deltas. Using this function, the combined SCV sets with an overall negative effect can be defined as:

$$\tilde{Z}_\omega = \left\{ \sigma_i | \sigma_i \in \Sigma^\Gamma_{\text{app}} \cup \Sigma^\Gamma_{\text{untested}} \wedge \text{filter}(\tilde{\delta}^*_\omega(\sigma_i, \sigma_{\text{current}})) = 0 \right\} \tag{7.24}$$

Accordingly, the set of filtered deltas $\tilde{\Delta}_{\omega}^{\tilde{Z}}$ for a cell $\omega \in \Omega$ is defined as:

$$\tilde{\Delta}_{\omega}^{\tilde{Z}} = \left\{ \tilde{\delta}_{\omega}^{*}(\sigma_i, \sigma_{\text{current}}) | \sigma_i, \sigma_{\text{current}} \in \Sigma_{\omega}^{\tilde{Z}}, \tilde{c} \in \tilde{C} \right\} \tag{7.25}$$

with $\Sigma_{\omega}^{\tilde{Z}} = (\Sigma_{\text{app}}^{\Gamma} \cup \Sigma_{\text{untested}}^{\Gamma}) \setminus \tilde{Z}_{\omega}$. These filtered deltas and the results of comparing the KPI targets with the current network performance can be used to indicate the performance of possible SCV set combinations. The latter is explained in the next step.

KPI Target vs. Network Performance Comparison The result of this step indicates how far a KPI is away from fulfilling its target with the current combined SCV set. Technical objectives are taken into account and the respective targets are compared with the current performance of the KPIs. Again, this step needs to be performed for each individual cell.

$$\tilde{\lambda}_{\omega} = (\xi_{\omega} - \vartheta_c) \cdot \varsigma \tag{7.26}$$

The difference is calculated by subtracting the vector of target values ϑ_c from the vector containing the current KPI values ξ_{ω} and multiplying the result with the scaling vector ς. Note that the KPI targets depend on a manual context class $c \in C$ with the operational context of the cell being in c, i.e., $\chi_{\omega} \in c$. Further note that manual context classes are unambiguous, i.e., each cell can be clearly assigned to exactly one manual context class. In the end, the set of comparison vectors $\tilde{\Lambda}$ is defined as

$$\tilde{\Lambda} = \{\tilde{\lambda}_{\omega_1}, \tilde{\lambda}_{\omega_2}, ..., \tilde{\lambda}_{\omega_{|\Omega|}}\} \tag{7.27}$$

These $\tilde{\lambda}_{\omega} \in \tilde{\Lambda}$ are combined with the respective set of filtered deltas $\tilde{\Delta}_{\omega}^{\tilde{Z}}$ in the next step.

Performance Indication For each of the SCV sets where a delta exists in $\tilde{\Delta}_{\omega}^{\tilde{Z}}$, a prediction is calculated indicating the effect when changing from the currently active SCV set combination to this set. This is done for each individual cell by adding the afore-calculated difference between the cell's KPI targets and current KPI performance to the respective delta values.

$$\tilde{\iota}_{\omega}(\sigma_i, \sigma_{\text{current}}) = \tilde{\delta}_{\omega}^{*}(\sigma_i, \sigma_{\text{current}}) + \tilde{\lambda}_{\omega} \tag{7.28}$$

Doing this for all remaining SCV sets σ_i in $\tilde{\delta}_{\omega}^{*}(\sigma_i, \sigma_{\text{current}})$ for a given cell ω, the result is a set $\tilde{I}_{\omega}$ of $\tilde{\iota}_{\omega}(\sigma_i, \sigma_{\text{current}})$s, i.e.:

$$\tilde{I}_{\omega} = \left\{ \tilde{\iota}_{\omega}(\sigma_i, \sigma_{\text{current}}) | \sigma_i, \sigma_{\text{current}} \in \Sigma_{\omega}^{\tilde{Z}}, \omega \in \Omega \right\} \tag{7.29}$$

Note that the $\tilde{\iota}_{\omega}(\sigma_i, \sigma_{\text{current}})$s are again vectors containing one value for each of the observed KPIs, a negative value indicating that a KPI target can not be fulfilled and a positive value representing a KPI fulfillment. Here, a big advantage compared to ASM can be seen: Instead of having this indication

per manual context class, it is available for each individual cell, meaning that cells can be differently configured while still being in the same manual context class. In a final step, the most suitable SCV set combination, i.e., the one which best fulfills the KPI targets, can be selected.

SCV Set Selection With the results of the previous step, it is easy to calculate a utility for each possible SCV set combination. Remember that a positive value in the indication vector represents the fulfillment of a KPI target and vice versa. Thus, a function is needed similar to Equation 6.37, returning either 1 or 0 depending on a positive or negative value, i.e.:

$$\text{sat}(\sigma_i, k_j) = \begin{cases} 1 & \text{if } \text{proj}_j(\tilde{\iota}_\omega(\sigma_i, \sigma_{\text{current}})) \geq 0 \\ 0 & \text{otherwise} \end{cases} \tag{7.30}$$

with $\sigma_i \in \Sigma_\omega^{\tilde{Z}}$, $k_j \in K$ and $\text{proj}_j(\tilde{\iota}_\omega(\sigma_i, \sigma_{\text{current}})$ being a projection on the jth element in the vector $\tilde{\iota}_\omega(\sigma_i, \sigma_{\text{current}})$. For the calculation of utilities, the weights w_{k_j} of the KPI targets come into play such that a weighted sum can be determined for each SCV set combination and each cell, i.e.:

$$U(\sigma_i) = \sum_{1 \leq j \leq |K|} \text{sat}(\sigma_i, k_j) \frac{w_{k_j}}{\bar{w}} \tag{7.31}$$

with $\bar{w} = \sum_{i \in K} w_i$. Equal to the ASM process, the distance to the KPI targets can be additionally determined in order to not only find the SCV set with the best utility, but the SCV set with the overall best performance in terms of KPIs. This distance is calculated as follows:

$$D(\sigma_i) = \left(\text{proj}_1(\tilde{\iota}_\omega(\sigma_i, \sigma_{\text{current}}) \cdot w_{k_1}\right) + \cdots + \left(\text{proj}_{|K|}(\tilde{\iota}_\omega(\sigma_i, \sigma_{\text{current}}) \cdot w_{k_{|K|}}\right) \tag{7.32}$$

where $\text{proj}_j(\tilde{\iota}_\omega)$ refers to the same projection function as defined above. The KPIs' weights are again considered to best reflect the operator's requirements in terms of KPIs. The combined SCV set with the biggest (positive) distance then does best in fulfilling the targets and hence, is the one to be selected for the examined cell.

7.2.4. Policy System

The policy system in CSM possesses one main difference to the policy systems in previous approaches: Since all calculations in the objective manager are done per cell, one rule is generated for each of them and hence, the policy repository contains $|\Omega|$ rules.

Thus, the number of rules has significantly increased compared to the ASM approach, in which one rule exists per manually defined context class in the end. However, the number of rules is still manageable, even when the mobile network

```
1 IF cell THEN SCV set combination
```

Listing 7.4: SCV set policy rule in CSM

consists of hundreds or a few thousand cells. Also, the rules are less complex as in the PBSM and ODSM approaches where conditions were based on context attributes and there values, laboriously combined in the end. Furthermore, the decision process in the PDP is less complex: Instead of mapping operational context to the definition of classes in the manual context classes model, the only thing that is needed as input is the ID of the cell. A rule can then be unambiguously related to that cell and the action can be executed, i.e., the SCV sets can be deployed to the SON function instances installed on the cell.

7.3. Example

The CSM approach as it is used in the learning phase, is exemplified in this section. Therefore, this section is divided into three subsections: First, the derivation of a learned context classes model is shown. Second, this learned context classes model and a real network effect model are used to generate a learned effect model. And finally, SCV sets can be selected according to the real network and learned effect model. However, before starting with the calculations, some premises in terms of the network and the required models must be constituted.

Since doing calculations manually becomes very complex with an increasing size of the mobile network, for this example a very simple network is assumed with three cells and two SON functions operating in the network, namely MLB and MRO. Each of these SON functions has two possible SCV sets whereby an SCV set only consists of one single SCV, i.e., the SON functions only have one SCP that can be adapted. Two KPIs are observed in the network, namely DCR and CL. Summarized, the following assumptions apply for the presented mobile network.

The set of cells Ω is defined as follows:

$$\Omega = \{\text{site01}[0], \text{site01}[1], \text{site01}[2]\} \tag{7.33}$$

The set of SON functions F is defined as follows:

$$F = \{\text{MLB}, \text{MRO}\} \tag{7.34}$$

The set of KPIs K is defined as follows:

$$K = \{\text{DCR}, \text{CL}\} \tag{7.35}$$

The set of SCV sets for MLB S^{MLB} is defined as follows:

$$S^{\text{MLB}} = \{\text{MLB\_1}, \text{MLB\_2}\} \tag{7.36}$$

The set of SCV sets for MRO S^{MLB} is defined as follows:

$$S^{\text{MRO}} = \{\text{MRO\_1}, \text{MRO\_2}\} \tag{7.37}$$

It is further assumed that all SCV set combinations are non-conflicting, i.e.:

$$\Sigma^{\Gamma} = \{\sigma_{11}, \sigma_{12}, \sigma_{21}, \sigma_{22}\} \tag{7.38}$$

with the first number representing the MLB set number and the second one representing the MRO set number.

A few models are needed as input for further calculations which are depicted below.

1. An **Objective Model** containing the targets to be achieved for manual context classes.

```
{
  IF CLASS_1 THEN DCR ≤ 0.03 WITH 0.7
  IF CLASS_1 THEN CL ≤ 0.5 WITH 0.3
  IF CLASS_2 THEN DCR ≤ 0.02 WITH 0.4
  IF CLASS_2 THEN CL ≤ 0.6 WITH 0.6
}
```

Listing 7.5: Exemplary objective model for two manual context classes in CSM

2. A **Manual Context Classes Model** such that these targets can be assigned to the respective cells. The operational context of site01[0] and site01[2] is assumed to be $\chi_{\text{site01[0]}} = \chi_{\text{site01[2]}} = \{\text{location} = \text{urban}\}$ whereas for site01[1] it is $\chi_{\text{site01[1]}} = \{\text{location} = \text{suburban}\}$. Consequently, site01[0], site01[2] $\in$ CLASS_1 and site01[1] $\in$ CLASS_2.

```
{
  IF location = urban THEN CLASS_1
  IF location = suburban THEN CLASS_2
}
```

Listing 7.6: Exemplary manual context classes model in CSM

3. A **Learned Context Classes Model** defining clusters of cells where cells have a similar behavior in terms of KPI values. Thereby, $k_{\text{means}} = 2$, i.e., two learned clusters of cells exist. As can be seen in Listing 7.7, cells are assigned to different classes as in the manual context classes model, i.e., site01[0] $\in$ LEARNED_CLASS_1 and site01[1], site01[2] $\in$ LEARNED_CLASS_2.
4. A **Real Network Effect Model** reflecting the actual impact of combined SCV sets in the network. Thereby, for LEARNED_CLASS_1 effect predictions exist for σ_{11} and σ_{12}, i.e., these combinations have already been applied in the real network. For LEARNED_CLASS_2, SCV set combinations σ_{12}, σ_{21} and σ_{22} have already been applied.

```
{
  IF site01[0] THEN LEARNED_CLASS_1
  IF site01[1] THEN LEARNED_CLASS_2
  IF site01[2] THEN LEARNED_CLASS_2
}
```

Listing 7.7: Exemplary learned context classes model in CSM

```
{
  IF LEARNED_CLASS_1 AND σ11 THEN DCR = 0.013 AND CL = 0.495
  IF LEARNED_CLASS_1 AND σ12 THEN DCR = 0.032 AND CL = 0.432
  IF LEARNED_CLASS_2 AND σ12 THEN DCR = 0.041 AND CL = 0.644
  IF LEARNED_CLASS_2 AND σ21 THEN DCR = 0.024 AND CL = 0.629
  IF LEARNED_CLASS_2 AND σ22 THEN DCR = 0.022 AND CL = 0.577
}
```

Listing 7.8: Exemplary real network effect model in CSM

5. A **Learned Effect Model** complementing the real network effect model with effect predictions for untested SCV set combinations. This model contains up-to-now untested, i.e., not applied combinations of SCV sets, i.e., σ_{21} and σ_{22} for LEARNED_CLASS_1 and σ_{11} for LEARNED_CLASS_2.

```
{
  IF LEARNED_CLASS_1 AND σ21 THEN DCR = 0.027 AND CL = 0.515
  IF LEARNED_CLASS_1 AND σ22 THEN DCR = 0.024 AND CL = 0.489
  IF LEARNED_CLASS_2 AND σ11 THEN DCR = 0.015 AND CL = 0.573
}
```

Listing 7.9: Exemplary learned effect model in CSM

Note that it is assumed that CSM is already running since a few time steps (cf. Figure 7.5), meaning that the combined effect model is not needed anymore and the real network effect model, the learned effect model and the learned context classes model represent the most current version of these models. It is assumed that these models have been generated at 8 o'clock am. In the following, these models are recalculated and adapted according to new measurements that have been taken at 9 o'clock am. The respective measurements database is shown in Table 7.2.

7.3.1. Derivation of Learned Context Classes Model

For the clustering of the cells, it is first necessary that a common data basis is given, i.e., the same SCV set effects need to be existent for all the cells. When analyzing the measurements database, it can be seen that the set of already applied combinations strongly differs for all three cells. This is illustrated in Figure 7.10 where blue squares denote existing measurements and gray ones are referring to missing measurements.

Table 7.2.: Exemplary raw measurements for all cells in the network

ID	DATE	CELL	MLB	MRO	DCR	CL
1	19-12-2018 07:00:00	site01[0]	1	1	0.013	0.495
2	19-12-2018 07:00:00	site01[1]	2	2	0.017	0.598
3	19-12-2018 07:00:00	site01[2]	2	1	0.024	0.629
4	19-12-2018 08:00:00	site01[0]	1	2	0.032	0.432
5	19-12-2018 08:00:00	site01[1]	2	2	0.025	0.566
6	19-12-2018 08:00:00	site01[2]	1	2	0.041	0.644
7	19-12-2018 09:00:00	site01[0]	2	2	0.021	0.470
8	19-12-2018 09:00:00	site01[1]	1	1	0.016	0.558
9	19-12-2018 09:00:00	site01[2]	2	2	0.014	0.607

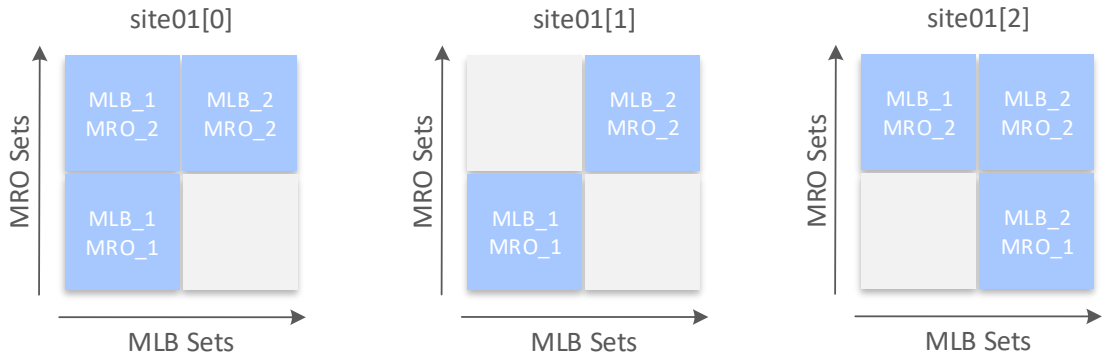


Figure 7.10.: Existing measurements for all three cells in the network according to measurements database

These blanks need to be filled in order to enable a meaningful clustering of the cells. According to the most current learned context classes model, site01[0] is assigned to LEARNED_CLASS_1 and hence, the rule in the real network effect model or the learned effect model needs to be found where the condition part exists of LEARNED_CLASS_1 and the missing SCV set combination σ_{21}. Since this combined SCV set obviously has not been applied yet for none of the cells in LEARNED_CLASS_1, this rule is part of the learned effect model. For cell site01[1], a measurement is missing for σ_{12} resulting in a gap that can be filled by the real network effect model where the effect for LEARNED_CLASS_2 in combination with σ_{12} is predicted. Doing this for all gray squares in Figure 7.10 results in the following table that serves as input to the clustering algorithm.

Since having eight values for each of the cells, i.e., two KPI values for each of the four possible SCV set combinations, this spans up an eight-dimensional state space which can be hardly illustrated. Hence, it is worth having a closer look to the data table. While all cells provide a fairly low DCR, more precisely, $0.013 \leq \text{DCR} \leq 0.041$, this value does not say a lot about the affiliation of cells to certain clusters. However, it can be seen that in three of the four cases, site01[1]

Table 7.3.: Completed set of measurements for all cells in the network

CELL	σ_{11}		σ_{12}		σ_{21}		σ_{22}	
	DCR	**CL**	**DCR**	**CL**	**DCR**	**CL**	**DCR**	**CL**
site01[0]	0.013	0.495	0.032	0.432	0.027	0.515	0.021	0.470
site01[1]	0.016	0.558	0.041	0.644	0.024	0.629	0.025	0.566
site01[2]	0.015	0.573	0.041	0.644	0.024	0.629	0.014	0.607

and site01[2] have a very similar DCR probably qualifying them for being in the same cluster. When having a look at the CL, this impression is confirmed. For these two cells, the CL value is noticeably higher for all of the four SCV set combinations compared to site01[0] and hence, these two cells should be in the same cluster while site01[0] builds the other cluster. This is exactly what is already expressed in the previous learned context classes model and hence, the model does not change in this time step.

7.3.2. Derivation of Learned Effect Model

This learned context classes model is then used for the derivation of a learned effect model. In addition, a real network effect model is needed that aggregates measurements in the database (i.e., Table 7.2) according to learned context classes in Listing 7.7. The latter has been already elaborated and exemplified in full detail in Chapter 6 and will not be explained step by step here. Hence, the real network effect model depicted in Listing 7.10 is assumed for further calculations.

```
1 {
    IF LEARNED_CLASS_1 AND σ11 THEN DCR = 0.013 AND CL = 0.495
3   IF LEARNED_CLASS_1 AND σ12 THEN DCR = 0.032 AND CL = 0.432
    IF LEARNED_CLASS_1 AND σ22 THEN DCR = 0.021 AND CL = 0.470
5   IF LEARNED_CLASS_2 AND σ11 THEN DCR = 0.016 AND CL = 0.558
    IF LEARNED_CLASS_2 AND σ12 THEN DCR = 0.041 AND CL = 0.644
7   IF LEARNED_CLASS_2 AND σ21 THEN DCR = 0.024 AND CL = 0.629
    IF LEARNED_CLASS_2 AND σ22 THEN DCR = 0.018 AND CL = 0.592
9 }
```

Listing 7.10: Exemplary real network effect model in CSM

Based on this real network effect model, the effects of untested σs can be calculated. Have in mind that this needs to be separately done for each learned context class. The real network effect model already contains predictions for all possible σs for LEARNED_CLASS_2 and hence, nothing needs to be learned for this class. However, in case of LEARNED_CLASS_1, a prediction for σ_{21} is missing and a learning algorithm can be applied to estimate the performance when applying this combined SCV set on all cells within LEARNED_CLASS_1 in the network.

When estimating the effect of a combined SCV set, the actual SCVs of the sets need to be taken into account. For the sake of simplicity, it is assumed that both

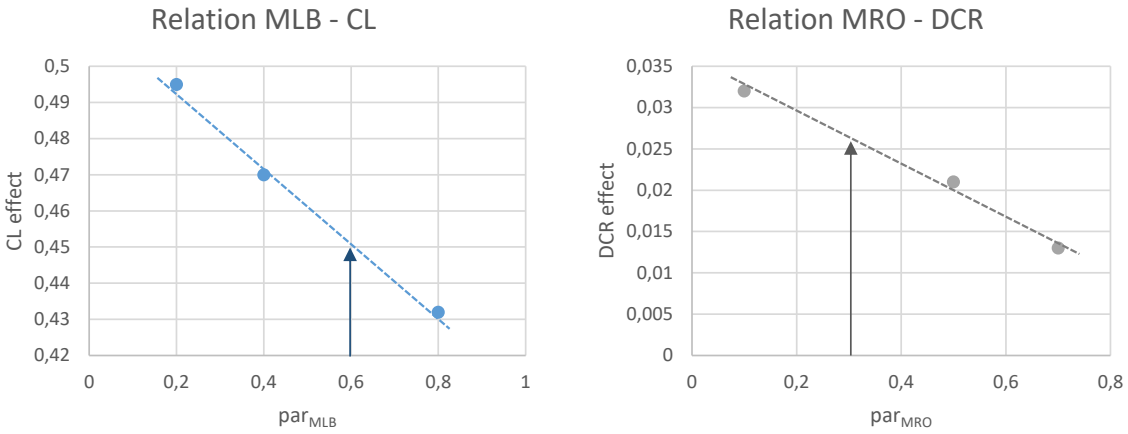
SON functions only have one SCP, called par_{MLB} and par_{MRO} in the following. For both SCPs, a domain $[0,1]$ is assumed and in addition, the unrealistic assumption is made that each KPI is only influenced by one SON function, i.e., DCR by MRO and CL by MLB. This is an assumption that generally argues against one of the main benefits of CSM: Finding optimal *combined* SCV sets for all cells in the network. However, this significantly facilitates further explanations.

The following table shows the respective SCP values for the four combined SCV sets.

Table 7.4.: SCPs and respective SCVs for all combined SCV sets

	par_{MLB}	par_{MRO}
σ_{11}	0.2	0.7
σ_{12}	0.8	0.1
σ_{21}	0.6	0.3
σ_{22}	0.4	0.5

Having these values, a graph can be drawn for each of the KPIs showing the relationship between the SON function's input parameter and the resulting KPI value. These two graphs are depicted in Figure 7.11a and Figure 7.11b. When applying learning algorithms to these datasets, the selected regression model could look as illustrated by the dotted lines, i.e., a simple linear relation between the SCPs and the respective KPI values exists. Using this linear model, the effects on the KPIs are predicted as indicated by the arrows, i.e., par_{MLB} maps to CL = 0,451 and par_{MRO} maps to DCR = 0,026.



(a) Relation between MLB parameter and CL value

(b) Relation between MRO parameter and DCR value

This information is then stored in the learned effect model, the result can be seen in Listing 7.11.

Note that the derivation of an effect model is exemplified in a very simplified form for demonstration purposes such that it gets easily understandable.

```
1 {
    IF LEARNED_CLASS_1 AND σ21 THEN DCR = 0.026 AND CL = 0.451
3 }
```

Listing 7.11: Resulting learned effect model

7.3.3. Selection of Combined SCV Sets

Now that all models are available, optimal SCV sets can be calculated and an SCV set policy can be generated. Therefore, the five steps mentioned in Section 7.2.3.3 are executed consecutively. First, the delta between the currently deployed SCV set combination and all other possible combinations is defined for each cell in the network. Looking at Table 7.2 it can be seen that $\sigma_{\text{current}}^{\text{site01[0]}} = \sigma_{22}$, $\sigma_{\text{current}}^{\text{site01[1]}} = \sigma_{11}$ and $\sigma_{\text{current}}^{\text{site01[2]}} = \sigma_{22}$. By means of the learned context class that is assigned to each of the cells, the expected performance can be read out from the real network effect model. For instance, since site01[0] is in context class LEARNED_CLASS_1, the expected performance of the currently deployed SCV set is DCR = 0.021 and CL = 0.470. These values are then subtracted from the effect predictions of all other combinations resulting in the deltas.

In a second step, the delta vectors are multiplied with a scaling vector ς, thereby enabling a filtering of the SCV set combinations. In this example, ς for the two observed KPIs is defined as follows:

$$\varsigma = \begin{pmatrix} -1 \\ -1 \end{pmatrix} \text{ for } \begin{pmatrix} \text{DCR} \\ \text{CL} \end{pmatrix} \tag{7.39}$$

In doing so, SCV sets with a negative effect on all KPIs can be discarded.

Table 7.5 shows the results in terms of deltas $\tilde{\delta}$, scaled deltas $\tilde{\delta}*$ and the result of filtering the scaled deltas. As can be seen, for site01[0], none of the combined SCV sets could be filtered out, meaning that they all have the potential to fulfill respective operator objectives in the best possible way. For site01[1], changing from the current SCV set combination to any other combination would definitely worsen the KPI fulfillment and hence, the selection is easy: σ_{11} still provides the best KPI effect. In case of site01[2], one more combination besides the currently deployed σ_{22} can possibly lead to a better KPI effect. Having a closer look, one can see that both observed KPIs are improved by σ_{11} due to the positive values. That is, while both options σ_{11} and σ_{22} possibly provide the same utility, σ_{11} will definitely be selected independent of the cell's targets, when applying a distance function, which is recommended in the learning phase.

As a consequence, due to the obvious results for site01[1] and site01[2], only site01[0] will be viewed in detail in the following.

In the next step of the objective manager, the current network performance is taken into account and compared with the targets that belong to the manual context class of site01[0]. The current network performance of site01[0] can again be

Table 7.5.: Deltas, scaled deltas and filtered deltas for all cells and all combined SCV sets

	σ_{11}	σ_{12}	σ_{21}	σ_{22}
site01[0]				
$\tilde{\delta}_{\text{site01[0]}}(\sigma_i, \sigma_{22})$	$\begin{pmatrix} -0.008 \\ 0.025 \end{pmatrix}$	$\begin{pmatrix} 0.011 \\ -0.038 \end{pmatrix}$	$\begin{pmatrix} 0.005 \\ -0.019 \end{pmatrix}$	$\begin{pmatrix} 0.0 \\ 0.0 \end{pmatrix}$
$\tilde{\delta}^*_{\text{site01[0]}}(\sigma_i, \sigma_{22})$	$\begin{pmatrix} 0.008 \\ -0.025 \end{pmatrix}$	$\begin{pmatrix} -0.011 \\ 0.038 \end{pmatrix}$	$\begin{pmatrix} -0.005 \\ 0.019 \end{pmatrix}$	$\begin{pmatrix} 0.0 \\ 0.0 \end{pmatrix}$
$\text{filter}(\tilde{\delta}^*_{\text{site01[0]}}(\sigma_i, \sigma_{22}))$	1	1	1	1
site01[1]				
$\tilde{\delta}_{\text{site01[1]}}(\sigma_i, \sigma_{11})$	$\begin{pmatrix} 0.0 \\ 0.0 \end{pmatrix}$	$\begin{pmatrix} 0.025 \\ 0.086 \end{pmatrix}$	$\begin{pmatrix} 0.008 \\ 0.071 \end{pmatrix}$	$\begin{pmatrix} 0.002 \\ 0.034 \end{pmatrix}$
$\tilde{\delta}^*_{\text{site01[1]}}(\sigma_i, \sigma_{11})$	$\begin{pmatrix} 0.0 \\ 0.0 \end{pmatrix}$	$\begin{pmatrix} -0.025 \\ -0.086 \end{pmatrix}$	$\begin{pmatrix} -0.008 \\ -0.071 \end{pmatrix}$	$\begin{pmatrix} -0.002 \\ -0.034 \end{pmatrix}$
$\text{filter}(\tilde{\delta}^*_{\text{site01[1]}}(\sigma_i, \sigma_{11}))$	1	0	0	0
site01[2]				
$\tilde{\delta}_{\text{site01[2]}}(\sigma_i, \sigma_{22})$	$\begin{pmatrix} -0.002 \\ -0.034 \end{pmatrix}$	$\begin{pmatrix} 0.023 \\ 0.052 \end{pmatrix}$	$\begin{pmatrix} 0.006 \\ 0.037 \end{pmatrix}$	$\begin{pmatrix} 0.0 \\ 0.0 \end{pmatrix}$
$\tilde{\delta}^*_{\text{site01[2]}}(\sigma_i, \sigma_{22})$	$\begin{pmatrix} 0.002 \\ 0.034 \end{pmatrix}$	$\begin{pmatrix} -0.023 \\ -0.052 \end{pmatrix}$	$\begin{pmatrix} -0.006 \\ -0.037 \end{pmatrix}$	$\begin{pmatrix} 0.0 \\ 0.0 \end{pmatrix}$
$\text{filter}(\tilde{\delta}^*_{\text{site01[2]}}(\sigma_i, \sigma_{22}))$	1	0	0	1

read from Table 7.2 which is DCR = 0.021 and CL = 0.470. Subtracting the target vector from the current performance vector and multiplying it with the scaling vector delivers the current distance to the KPI targets, positive values representing a fulfillment and vice versa.

$$\tilde{\lambda}_{\text{site01[0]}} = \left(\begin{pmatrix} 0.021 \\ 0.470 \end{pmatrix} - \begin{pmatrix} 0.03 \\ 0.50 \end{pmatrix} \right) \cdot \begin{pmatrix} -1 \\ -1 \end{pmatrix} = \begin{pmatrix} 0.009 \\ 0.030 \end{pmatrix} \tag{7.40}$$

It can be seen in Equation 7.40 that the SON functions currently perform well for cell site01[0]. However, it is still possible that there is a better SCV set combination than the currently active one.

This can be found out in the next two steps where first, the performance indication is calculated for each of the four SCV set combinations based on the result of Equation 7.40 and the filtered deltas. Based on these indication vectors, the

utility as well as the distance can be calculated leading to a clear result which combination performs best for site01[0]. This is illustrated in Table 7.6.

Table 7.6.: Scoring of the exemplary combined SCV sets

	$o_{\text{DCR}} = (0.03,\ 0.7)$			$o_{\text{CL}} = (0.5,\ 0.3)$			$D(\sigma_i)$	$U(\sigma_i)$
	$proj_{\text{DCR}}(\vec{i}(\sigma_i, \sigma_{22}))$	$sat(\sigma_i, DCR)$	w_{DCR}	$proj_{\text{CL}}(\vec{i}(\sigma_i, \sigma_{22}))$	$sat(\sigma_i, CL)$	w_{CL}		
σ_{11}	0.017	1	0.7	0.005	1	0.3	0.0134	1.0
σ_{12}	−0.002	0		0.068	1		0.0190	0.3
σ_{21}	0.004	1		0.049	1		0.0175	1.0
σ_{22}	0.009	1		0.030	1		0.0153	1.0

Table 7.6 shows that the three combined SCV sets σ_{11}, σ_{21} and σ_{22} fulfill all KPI targets, indicated by a utility of 1.0. Hence, the decision for one of the sets is taken by considering the distance. Here, an interesting fact can be detected: While σ_{12} provides the highest, i.e., the best distance, it is the only one that does not fulfill the target for DCR. This is due to the fact that it only falls slightly below the DCR target but on the other side, provides by far the best performance for CL. Since it is the most important requirement of an operator to fulfill objectives at all, σ_{21} is selected because $D(\sigma_{21}) \geq D(\sigma_{22}) \geq D(\sigma_{11})$.

7.3.4. Generation of SCV Set Policy

One main difference to the ASM approach is that in CSM, the resulting policy does not depend on manual context classes, but cells instead. Following the presented results, a policy for the example looks as depicted in Listing 7.12.

```
1 {
    IF site01[0] THEN MLB = MLB_2 AND MRO = MRO_1
3   IF site01[1] THEN MLB = MLB_1 AND MRO = MRO_1
    IF site01[2] THEN MLB = MLB_1 AND MRO = MRO_1
5 }
```

Listing 7.12: Resulting cell-based SCV set policy

Note that cells site01[1] and site01[2] belong to the same learned context class while site01[0] and site01[2] belong to the same manual context class. Since site01[1] and site01[2] are equally configured, the similar behavior of these two cells is confirmed. One of the main advantages of CSM can be seen here: Cells

within the same manual context class, i.e., cells where the same operator objectives apply, can still be configured differently, leading to an overall better network performance.

7.4. Related Work

Applying techniques in the field of machine learning currently is a hot topic in mobile network optimization. Several reinforcement learning approaches exist to learn optimal parameters for specific single SON functions. In [MM13a] and [MM13b], an approach is presented which adapts the CIO values of a cell's MLB function instance and the instances of all its neighbors by using Q-Learning depending on the current load values of these cells. In [Qin+13], an algorithm is presented optimizing the HO parameters of an MRO function through Q-Learning. For the CCO function, a lot of approaches exist dealing with the optimization of the function's parameters. Most of them also use reinforcement learning, e.g., [IM12b] and [IM12a] propose an approach for the initial configuration and optimization during operation of the vertical antenna tilt using fuzzy Q-Learning. Furthermore, the authors of [SBG15] present a solution to learn the optimal range of macro and pico cells in a small cell network, thereby improving the UEs' average throughput. In contrast, deep learning is used in [Yan+18] to find an optimal trade-off between coverage and capacity in massive multiple-input multiple-output (MIMO) wireless systems. [Gaz+18] aims at improving KPIs such as average throughput, fairness and coverage probability by adopting unsupervised learning techniques. However, what all these approaches have in common is, that they only investigate one single SON function. Conflicts between SON functions are not considered and they all assume that SON functions only influence one single or a limited set of KPIs. In addition, some of them only take specific cell types into account. However, mobile networks are strongly heterogeneous, i.e., a lot of SON functions are running concurrently and influencing a variety of KPIs in a multi-RAT network. Thereby, it is often not obvious which KPIs are affected by which SON functions. CSM is able to abstract from these restrictions and overcome these shortcomings by taking all type of cells and all SON functions into account and learning the joint effect of combined SCV sets on a set of KPIs.

An approach that coordinates a set of SON functions, is presented in [Iac+14a], [Iac+14b], [Iac+14c] and [Iac+16]. Reinforcement learning is used to coordinate actions of concurrently running and possibly conflicting SON functions. Also, the authors of [DK13] use a reinforcement learning algorithm, i.e., fuzzy Q-Learning, to solve conflicts between a handover optimization SON function and a load balancing function. While conflicts can be avoided this way, no optimal configuration can be found that fulfills operator objectives in the best possible way.

The approaches that are closest to the CSM approach in this chapter, are [MSM18] and [MSM16] as well as [DJD17a], [DJD17b], [DJD18a] and [DJD18b]. The authors of the first two publications have developed a Q-Learning framework for cognitive cellular networks where SON function instances independently act as reinforcement learning agents in the network with the goal to optimally configure the SON functions. The authors of the latter approaches also adopt different reinforcement learning algorithms such as LinUCB and a multi-armed bandit algorithm to learn an SCV set policy with combinations of SCV sets. While [MSM18] and [MSM16] are not policy-based, the other approaches are based on the general SON management approach developed by the author of this thesis and presented in Chapter 3, Chapter 4 and Chapter 5, and hence, come closest to the approach presented in this chapter. Even though all of them aim at the same objective, i.e., finding an optimal configuration for all cells in the network, they do not allow the network to be controlled by the MNO any more. SON functions act completely autonomously, always aiming at optimizing specific KPIs without considering operator objectives and the joint fulfillment of them. In doing so, it is hard to gain the trust of an MNO which is one of the main objectives of this thesis. In CSM, important actions must be always authorized by the MNO first while still providing a high degree of automation.

Part III.

Evaluation and Conclusion

8 Evaluation

Several approaches on SON management have been described in detail in the previous chapters. In this chapter, these approaches are evaluated and the benefit of SON management in general as well as the improvement from the first to the fourth solution should be evaluated. Therefore, an LTE mobile network simulator is introduced utilized for all kind of simulations. This includes the simulator itself, the implementation of the SON functions, the SON management and the simulation scenarios that are used. This section is followed by a description of the input models and their derivation as they are employed by SON management. In the end, a comprehensive analysis of each SON management approach is performed and the results are amplified in detail before finishing the chapter with a discussion about the achieved results.

8.1. Simulation Setup

In order to implement the SON management approaches and to generate the necessary data for running it, an LTE network simulator is needed for the evaluation of the elaborated concepts. In 2009, the former Nokia Siemens Networks published a white paper about their first simulator of a SON, namely System Experience of Advanced SON (SEASON) [SON09]. In cooperation with NOMOR research GmbH, the present Nokia Bell Labs has further developed the initial SEASON simulator to the current version SEASON II on which this evaluation is based on. On top of SEASON II a SON function engine is installed developed by Nokia Bell Labs and its predecessors which enables to run several SON functions concurrently in SEASON II. Furthermore, a SON management component has been developed by the author of this thesis using the SON functions provided by the SON function engine for the configuration of the SEASON II-provided LTE network.

8.1.1. LTE Network Simulator

SEASON II is a mobile network simulator developed in C++. Figure 8.1 provides a brief overview of the functional components being active in SEASON II and

the relationship between them. This figure and the corresponding description is based on the SEASON II user manual [Buc+18].

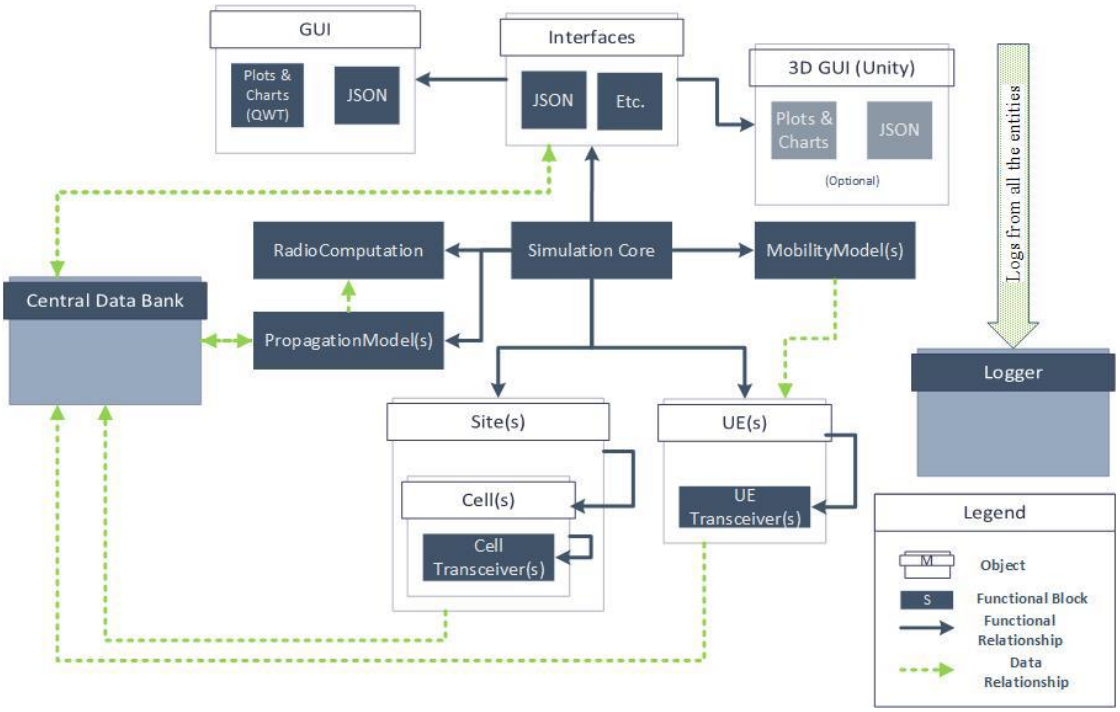


Figure 8.1.: Overview of SEASON II with its functional components and relationships between them [Buc+18]

The main components of SEASON II are briefly presented in the following:

Simulation Core This is the central component which runs the actual simulation and coordinates all other components. Furthermore, the simulation core is responsible for instantiating, for instance, sites and UEs.

Sites Each site consists of several cells. Each cell has a transceiver with the cell's properties such as the type of RAT or the list of neighbors. A site has a unique ID such as "site05" and each cell is identified by a unique ID within its site, such that a cell can be clearly identified by, e.g., "site05[2]". KPIs are measured and aggregated per cell.

UEs A UE can be described as one user within the mobile network simulator, demanding a connection to a specific cell, thereby producing traffic and influencing the behavior of a cell in terms of KPIs. Each UE can have multiple transceivers, one for each deployed RAT. UEs are moving according to certain mobility models, hence they are also changing their affiliation to certain cells.

Propagation Model SEASON II supports different types of radio propagation models. The one which is used for the simulations in this chapter, is Winner+ [Mei+10]. Winner+ simulates a realistic radio propagation and is usu-

ally used for large urban and suburban scenarios. It uses raytracing to realistically model the effect of buildings on radio waves.

Mobility Model Different mobility models are supported by SEASON II: A street-graph model enables that UEs move on streets with a realistic speed. This data is coming from OpenStreetMap XML files and mapped to nodes and edges with associated speed limits. Another type of mobility model is used for further simulations, namely the circle mobility model. As the name suggests, UEs are moving within a defined circle with a defined speed randomly.

Central Data Bank All information about the scenario, the configuration of the mobile network and the produced KPIs is stored in this data base. Hence, it has a data relationship to most of the other components. The central data bank serves as interface to the outside world providing it with data about the running simulation.

Interfaces The JavaScript Object Notation (JSON) format [JSON] is used for configuration files. Within these files, one can define, e.g., the desired propagation model, sites and their properties, UEs groups and their properties, different types of mobility models for different UE groups, or the KPIs that should be observed just to name a few.

GUI A 2D Graphical User Interface (GUI) is attached to the simulation always representing the current state of the network. That is, amongst others, it shows buildings in the observed area, the position of sites, the movement of UEs, and offers the possibility to visualize a set of KPIs.

Another key feature of SEASON II is that it allows to dynamically reconfigure the network in terms of, e.g., the cells' configuration parameters, and that it also enables to dynamically change the deployed scenario in terms of, e.g., the number of UEs which is essential for the presented evaluation. Note that only the key features and those relevant for this chapter are presented here. However, SEASON II offers much more functionality such as other propagation models, mobility models or the ability to support network slicing.

8.1.2. SON Function Engine

A SON function engine has been developed by Nokia Bell Labs which implements the algorithmic details of different SON functions. By connecting the SON function engine with the SEASON II simulator, cells' properties, i.e., NCPs, can be changed which affects the network behavior and hence, the observed KPIs. Using a SON function engine makes the SEASON II simulator a SON-enabled mobile network simulator.

Several SON functions are implemented in the SON function engine, the relevant ones are presented in the following.

Coverage and Capacity Optimization (CCO) has the goal to find a balance between coverage and capacity, i.e., the capacity per UE should be improved while at the same time coverage holes should be avoided. This is achieved by changing the remote electrical tilt of the cell's antenna. Hence, CCO aims at improving the CQI value and can be configured by the following SCPs:

- **active:** Indicates whether the CCO function is active or not.
- **cqi_threshold:** Indicates the threshold for becoming active based on the current CQI value.
- **tilt_stepsize:** Indicates how to change the remote electrical tilt of the cell's antenna. Negative values indicate a decreasing tilt value and vice versa.

Mobility Load Balancing (MLB) has the goal to reduce the load within a cell by shifting UEs to neighboring cells. This is achieved by adapting the CIO value to all neighboring cells. Hence, MLB aims at improving the CL value and can be configured by the following SCPs:

- **active:** Indicates whether the MLB function is active or not.
- **load_threshold:** Indicates the threshold for becoming active based on the current CL value.
- **cio_stepsize:** Indicates how to change the CIO value, i.e., the virtual cell border. Negative values indicate a decreasing CIO value and vice versa.

Mobility Robustness Optimization (MRO) has the goal to reduce unnecessary HOs to neighboring cells. This is achieved by adapting the Time to Trigger (TTT) to a specific neighboring cell. Hence, MRO aims at improving the PiPo value and can be configured by the following SCPs:

- **active:** Indicates whether the MRO function is active or not.
- **pipo_threshold:** Indicates the threshold for becoming active based on the current PiPo value.
- **ttt_stepsize:** Indicates how to change the TTT value, i.e., the time for fulfilling the conditions for a HO. Negative values indicate a decreasing TTT value and vice versa.

8.1.3. SON Management

The LTE network simulator and the SON function engine are required as a basis for the implementation of a SON management system. The SON management is implemented in Java and indirectly uses the SON function engine for the adaptation of NCPs in SEASON II. That is, SON management implements a policy

system that deploys certain SCV sets on the SON function instances which then adapt the NCPs accordingly.

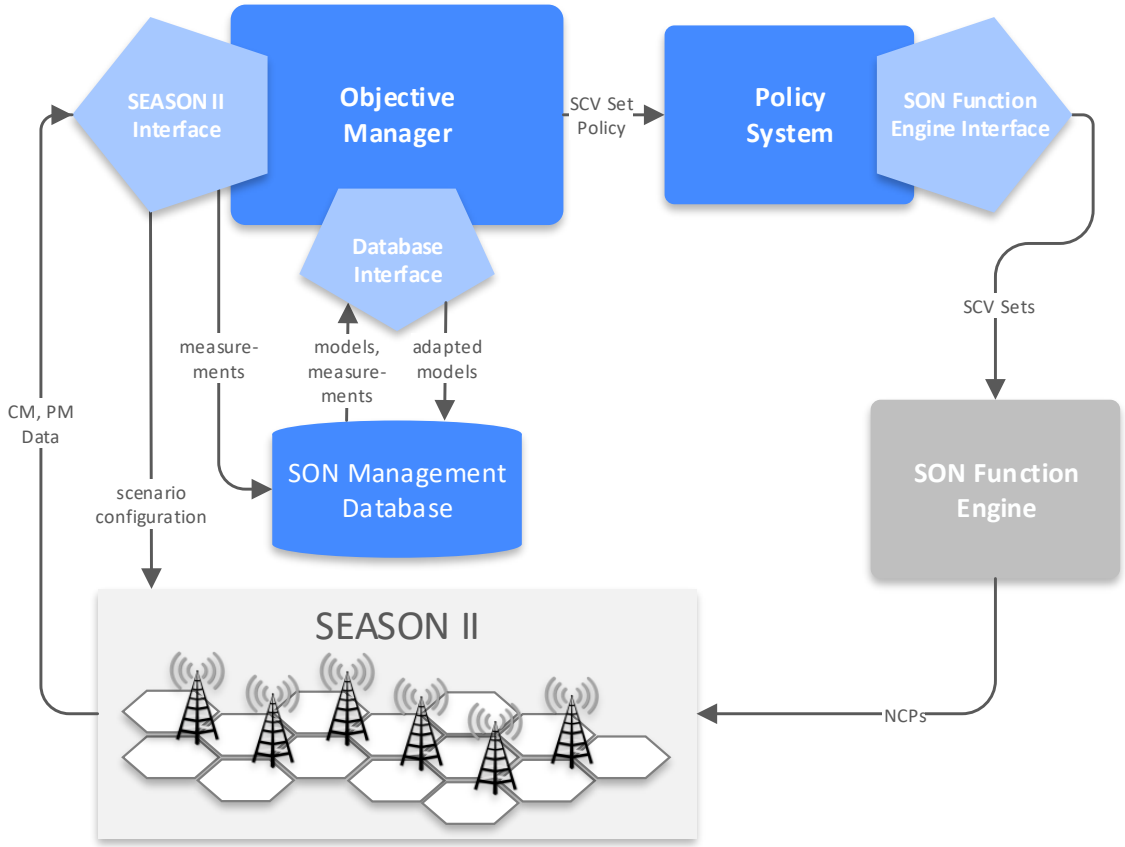


Figure 8.2.: Overview of implemented SON management parts and dependencies between them

In general, the SON management system consists of six parts whose dependencies are depicted in Figure 8.2 and which are described below:

Objective Manager The core component of SON management, responsible for calculating optimal SCV sets. The underlying methodology varies depending on the selected approach, i.e., PBSM, ODSM, ASM or CSM.

SON Management Data Base Input models, i.e., the context model(s), the objective model and effect model(s) are stored in a database in a way that is readable for SON management. Also, KPI measurements are written into the database by the SEASON II interface.

Database Interface An interface to the SON management database for receiving stored input models and SEASON II measurements, and for storing models that have been adapted by the objective manager.

SEASON II Interface An interface to the LTE network simulator in order to provide the SON management system with CM and PM data such as KPI values, and to reconfigure the network scenario deployed in SEASON II.

SON Function Engine Interface An interface to the SON function engine as part of the policy system in order to send SCV set change requests. These configuration changes are then sent to SEASON II by the SON function engine in terms of NCP changes.

Policy System Configuration change requests are given to the SON function engine via the respective interface. That is, the policy system is responsible for choosing the suitable SCV sets for each cell out of the SCV set policy that has been generated by the objective manager.

In addition to the four presented approaches, the objective manager is implemented in a further way. In current SON-enabled mobile networks, SON functions usually run with default SCV sets only. The implementation of this default configuration case serves as baseline for a comparison with the particular SON management approaches. Note that in case of CSM, also a learning engine is implemented within the objective manager.

8.1.4. Scenarios

For the following studies two different scenarios are necessary. Referring back to Chapter 6, a manufacturer-generated model has the major shortcoming that it has been generated in a different environment than the one where it is applied in the end. It is one of the main goals of this thesis to develop a SON management approach that comes as close to reality as possible and hence, this situation needs to be simulated. Therefore, SEASON II is configured in two different ways, meaning that two completely different scenarios are created. The first scenario simulates the environment of a SON function manufacturer and hence, is used for the generation of manufacturer-provided effect models. This scenario is called *Helsinki scenario* and is described in detail in Section 8.1.4.1. The second scenario simulates the real network environment, i.e., the mobile network where the manufacturer-provided effect models should be applied and that should be optimized according to operator objectives. This scenario is called *Hamburg scenario* and is described in detail in Section 8.1.4.2.

8.1.4.1. Helsinki Scenario

The Helsinki scenario covers an area of around 50km^2 in the city center of Helsinki. 35 cells are installed in this scenario, 32 of them macro and 3 micro cells. The exact (initial) configurations of the cells, the radio parameters and the users in the scenario can be seen in Table 8.1.

Table 8.1.: Initial parameter settings in the Helsinki scenario

Category	Parameter	Value
Radio	Radio technology	LTE
	Carrier frequency	2 GHz
	Carrier bandwidth	20 MHz
	Radio propagation model	Winner+
Network	Type	Urban HetNet
	Area	$50km^2$
	Granularity Period	5400 s
	Macro cells	
	Number of cells	32
	Height	25 m
	Antenna gain	14 dB
	Transmission Power	55 dB
	Azimuth	65°
	Elevation	9°
	Micro cells	
	Number of cells	3
	Height	8 m
	Antenna gain	5 dB
	Transmission Power	0 dB/35 dB
	Azimuth	360°
	Elevation	15°
Users	Data rate per user	1.2 mbps
	Speed of streetgraph model users	according to speed limit
	Speed of highway users	120 km/h
	Speed of city center users	6 km/h
	Speed of hot spot users	6 km/h
	High traffic scenario users	
	Number of streetgraph model users	600
	Number of highway users	20
	Number of city center users	100
	Number of hot spot users	75
	Low traffic scenario users	
	Number of streetgraph model users	400
	Number of highway users	13
	Number of city center users	66

The data rate per user is set to 1.2 mbps which is fairly high, but instead the number of users is quite low making it a realistically loaded mobile network. The Helsinki scenario is further subdivided into two scenarios: While the network itself and the radio parameter configuration is the same in both, the number of users differs, thereby simulating a scenario with high traffic and one with low

traffic. In the high traffic scenario there are three hot spot groups with 75 users in total, each 25 of them within the area of one micro cell. In the low traffic scenario, these micro cells are deactivated (indicated by a transmission power of 0 dB) and consequently, there are also no additional users within the scope of these cells. The number of streetgraph users is reduced from 600 to 400 in the low traffic scenario and also highway and city center users are reduced by 1/3.

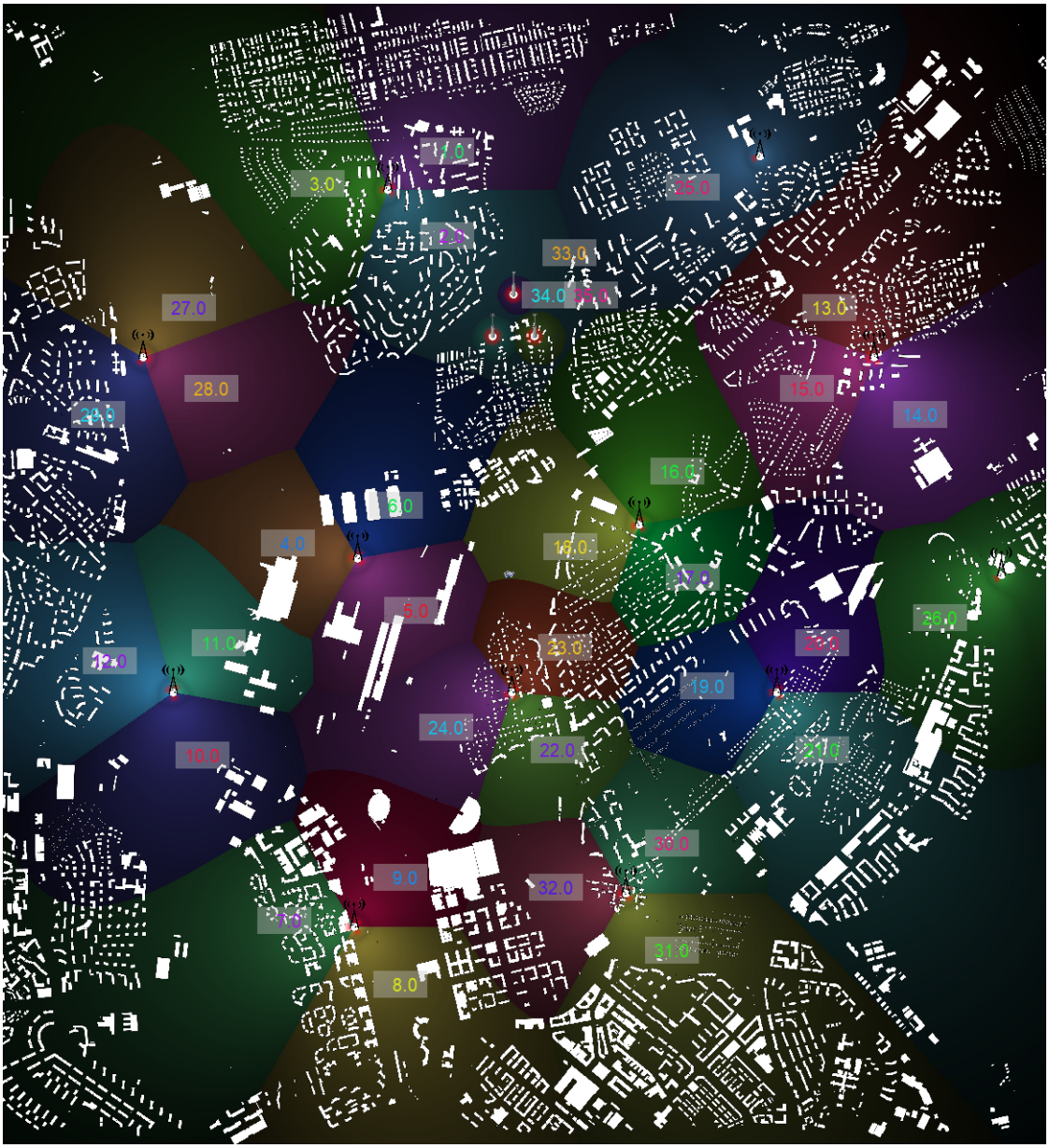


Figure 8.3.: Screenshot of the Helsinki scenario, colored regions represent the respective coverage area, numbers represent the unique ID of a cell

An overview of the resulting network scenario can be seen in Figure 8.3 where the numbers indicate the unique cell IDs. Micro cells have the IDs 33, 34 and 35

and are placed close to the main station since this is a typical hot spot area at high traffic times. It can be seen that most of the macro sites consist of three cells while only two sites have one sector, namely those with cell 25 and 26. The white parts represent buildings while the colors indicate the propagation of the radio waves. Since a realistic propagation model is used, the buildings have a shadowing effect on cells, meaning that areas behind a building partially can not be reached by all of the antennas.

8.1.4.2. Hamburg Scenario

The Hamburg scenario covers an area of around 60 km^2 in the city center of Hamburg with 61 installed cells. Table 8.2 shows the initial configuration of the scenario.

Table 8.2.: Initial parameter settings in the Hamburg scenario

Category	Parameter	Value
Network	Area	60km^2
	Macro cells	
	Number of cells	54
	Height	32 m
	Transmission Power	55 dB
	Micro cells	
	Number of cells	7
	Height	8 m
	Transmission Power	0 dB/40 dB
Users	Speed of streetgraph model users	according to speed limit
	Speed of highway users	120 km/h
	Speed of city center users	6 km/h
	Speed of hot spot users	6 km/h
	Speed of boat users	22 km/h
	High traffic scenario users	
	Number of streetgraph model users	700
	Number of highway users	30
	Number of city center users	100
	Number of hot spot users	155
	Number of boat users	50
	Low traffic scenario users	
	Number of streetgraph model users	466
	Number of highway users	20
	Number of city center users	67
	Number of boat users	34

Note that Table 8.2 only depicts the parameters which are different to the Helsinki

scenario. That is, the radio parameters and most of the cells' parameters are equivalent to Table 8.1.

The Hamburg scenario, compared to the Helsinki scenario, is larger and contains more cells, i.e., 54 macro and 7 micro cells. Thereby, the micro cells are again placed in hot spot areas, i.e., three of them around the main station area, three on the Reeperbahn and one at the Elbphilharmonie. 20-25 users are moving slowly or, in case of the Elbphilharmonie, are static within a predefined circle. Additionally, 100 users are moving randomly at a low speed within the city center and 50 users are moving with 22 km/h on boats within the harbor area. Together with the streetgraph users, the scenario sums up to over 1000 users, each demanding a data rate of 1.2 mbps. Similar to the Helsinki scenario, the Hamburg scenario is configured in two different ways, one simulating high traffic times and one simulating a low traffic situation. In the low traffic scenario, the number of users is again reduced by 1/3 compared to the high traffic scenario and micro cells and the corresponding users within them are deactivated. A screenshot from SEASON II depicting the Hamburg scenario can be seen in Figure 8.4, the numbers again indicating the unique cell IDs.

8.2. Input Models

In this section, the derivation of all input models necessary for the execution of the SON management approaches, are described in detail. Thereby, these models are all generated under realistic conditions, i.e., as they would be generated within the respective domains. It starts with the manual context classes model which is created by inspection of the network. The derivation of the learned context classes model has already been shown in Section 7.2.2.1 and since this is an automated and continuously running process, the derivation can hardly be demonstrated. Based on the manual context classes, an operator can define technical objectives with respect to KPIs for each of the classes. For the effect model, two things need to be evaluated: First, a manufacturer-provided effect model needs to be created in the Helsinki scenario in a realistic way and the results need to be provided to SON management in a reasonable form. Second, for the generation of a learned effect model, an appropriate learning algorithm needs to be selected that performs best in the real network. The data generation in the Hamburg scenario, the data preparation and the evaluation of the results is shown in Section 8.2.3.2.

8.2.1. Manual Context Classes Model

The manual context classes model mainly has two purposes: First, it serves as a basis for the definition of technical objectives and second, it is provided to the manufacturers of the SON functions, such that a context-dependent initial effect

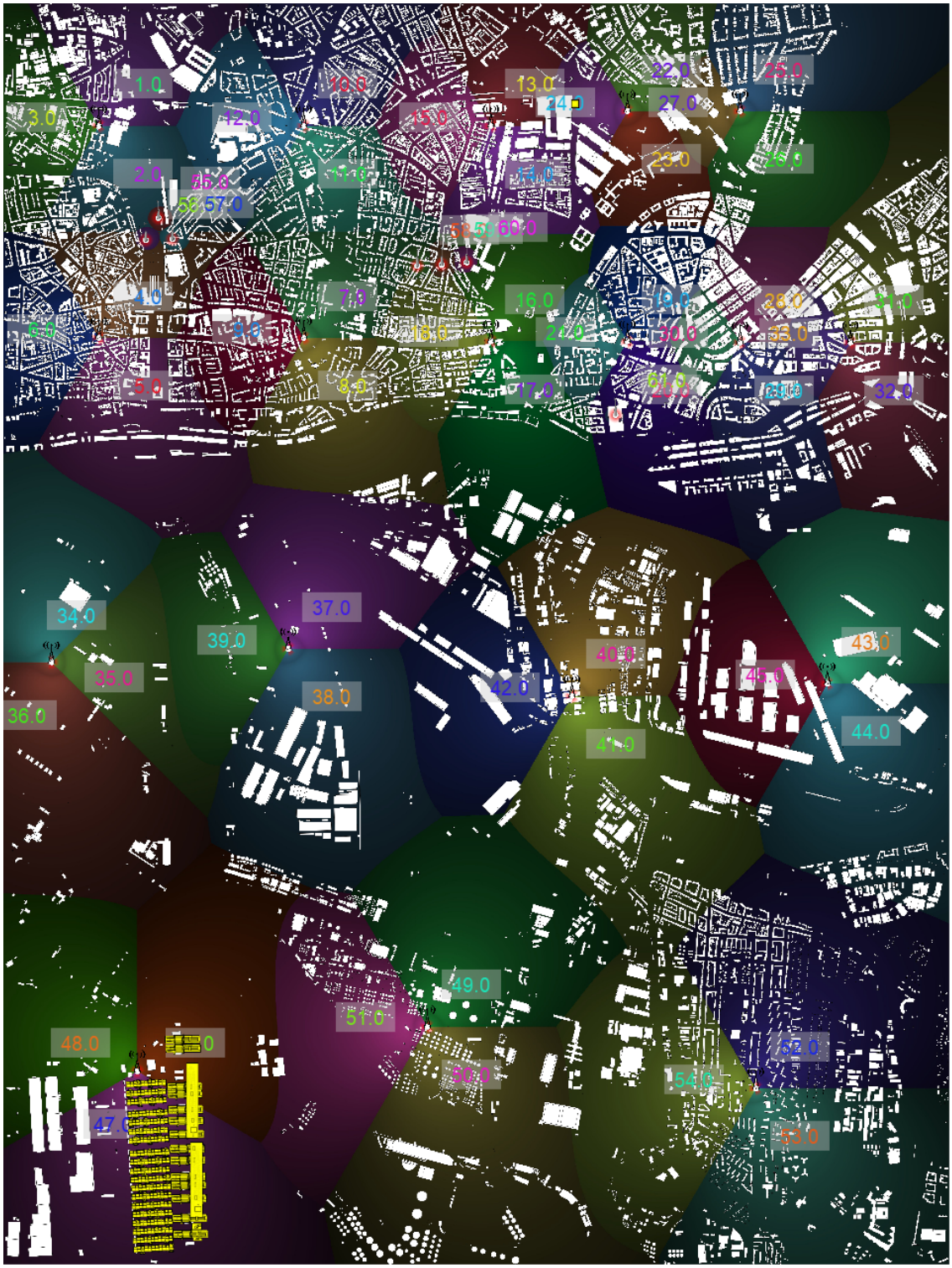


Figure 8.4.: Screenshot of the Hamburg scenario, colored regions represent the respective coverage area, numbers represent the unique ID of a cell

model can be created for each of the functions. Hence, context classes need to be specified for the real network, i.e., the Hamburg scenario. Therefore, an MNO inspects the network to find relevant network and cell attributes to be classified into a context class. The following network properties and behavior can be identified when looking at the Hamburg scenario:

- All cells in the network use LTE as RAT.
- There are macro and micro cells in the network.
- The scenario is separated into a densely populated area in the north, an industrial area in the middle and the southwestern part and a more suburban area in the southeastern part.
- Micro cells are only located in densely populated areas.
- There is a highway with fast moving users in the western part of the observed network.
- Users in the city center predominantly do not move a lot.
- Users in the industrial harbor area are mainly slowly moving.
- Users in the hot spot areas are mainly slowly moving.
- Users in the suburban area predominantly demand for a connection while being in the car with a moderate speed.
- There is a big difference in traffic between daytime and nighttime.
- Micro cells are deactivated at nighttime.

Bringing together all these detections, the context attributes model looks as follows:

```
1 {
    technology HAS DOMAIN {LTE},
3   location HAS DOMAIN {urban, suburban, industrial},
    cell_type HAS DOMAIN {macro, micro},
5   mobility_type HAS DOMAIN {static, pedestrian, car, highway},
    time HAS DOMAIN {day, night}
7 }
```

Listing 8.1: Context attributes model for the Hamburg scenario

Building the cross product over all attributes and their values for this small set of attributes would already lead to $1 * 3 * 2 * 4 * 2 = 48$ regions in the context state space which is too much to be handled manually in a reasonable way. Hence, only relevant context attribute combinations should be taken into account, each combination defining one manual context class. Therefore, the listed conditions need to be reasonably combined. For instance, since each cell is an LTE cell, micro cells are only placed in urban areas and micro cell users are predominantly slowly moving, this combination builds a reasonable context class occurring in that form

in the network. Since micro cells are deactivated at nighttime, this context class only plays a role at daytime. Doing this for all listed facts, the following manual context classes model arises where above mentioned example is comprised by the last rule.

```
{
  IF time = day AND technology = LTE AND location = urban
    AND cell_type = macro AND mobility_type = static
    THEN CLASS_1_HIGH
  IF time = night AND technology = LTE AND location = urban
    AND cell_type = macro AND mobility_type = static
    THEN CLASS_1_LOW
  IF time = day AND technology = LTE AND location = industrial
    AND cell_type = macro AND mobility_type = pedestrian
    THEN CLASS_2_HIGH
  IF time = night AND technology = LTE AND location = industrial
    AND cell_type = macro AND mobility_type = pedestrian
    THEN CLASS_2_LOW
  IF time = day AND technology = LTE AND location = industrial
    AND cell_type = macro AND mobility_type = highway
    THEN CLASS_3_HIGH
  IF time = night AND technology = LTE AND location = industrial
    AND cell_type = macro AND mobility_type = highway
    THEN CLASS_3_LOW
  IF time = day AND technology = LTE AND location = suburban
    AND cell_type = macro AND mobility_type = car
    THEN CLASS_4_HIGH
  IF time = night AND technology = LTE AND location = suburban
    AND cell_type = macro AND mobility_type = car
    THEN CLASS_4_LOW
  IF time = day AND technology = LTE AND location = urban
    AND cell_type = micro AND mobility_type = pedestrian
    THEN CLASS_5_HIGH
}
```

Listing 8.2: Manual context classes model of the Hamburg scenario

Note that this manual context classes model applies for both scenarios. While it has been defined based on the Hamburg scenario, it is provided to SON function manufacturers to make available an effect model with KPI effect predictions dependent on these classes. However, what is different for the two scenarios, is the assignment of cells to the respective context class. Note that *time* thereby is the only dynamic parameter while all other attributes are static, i.e., their values always remain constant for each cell. Consequently, except for micro cells, there are always two rules comprising the cells' properties, but only one is applicable depending on the current time of the day.

The assignments of cells to the respective manual context classes for the Hamburg scenario is shown in Listing 8.3 where the names of the cells are in accordance with Figure 8.4: site01[0] refers to the cell with ID 1.0, site01[1] to the cell with ID 2.0 and so on.

```
{
  CLASS_1 = {site01[1], site02[0], site02[1], site03[0], site03[1],
    site03[2], site04[1], site04[2], site05[1], site05[2], site06[1],
    site06[2], site07[0], site07[1], site07[2], site08[1], site08[2],
    site09[1], site09[2], site10[0], site10[1], site10[2], site11[0],
    site11[2]}
  CLASS_2 = {site01[0], site01[2], site02[2], site04[0], site05[0],
    site06[0], site08[0], site09[0], site11[1], site14[1], site15[1],
    site17[0], site17[1], site18[0], site18[1], site18[2]}
  CLASS_3 = {site12[0], site12[1], site12[2], site13[2]}
  CLASS_4 = {site13[0], site13[1], site14[0], site14[2], site15[0],
    site15[2], site16[0], site16[1], site16[2], site17[2]}
  CLASS_5 = {site19[0], site20[0], site21[0], site22[0], site23[0],
    site24[0], site25[0]}
}
```

Listing 8.3: Assignment of cells to manual context classes in the Hamburg scenario

8.2.2. Objective Model

For each of the nine context classes in Listing 8.2, an objective needs to be defined by the network operator. The KPIs that are observed in this evaluation, according to the three SON functions that mainly influence them, are: CQI, PiPo and CL. Hence, a target and a weight needs to be defined for each of them with the target between 0 and 1 and the weights summing up to 1.

The objective model as it is used for the evaluation of SON management in Section 8.3, is shown in Listing 8.4. The defined targets and weights shall only be exemplary explained. For instance, the CL target for micro cells is fairly high, since the purpose of these cells is to unload macro cells. However, the load should not exceed 90% since otherwise, the CQI decreases and neighboring cells do not handover UEs to the micro cells any more. Hence, CL is the highest ranked KPI with 0.6. What can also be seen, is that the CL target is always lower during nighttime which is due to the reduced number of UEs in this time. Furthermore, the CQI targets for urban cells are usually higher than for suburban or industrial cells since in these densely populated areas an MNO wants to achieve a high degree of customer satisfaction which is strongly related to CQI. Since UEs are moving quite fast at the highway, cells declared as highway cells usually have a higher PiPo rate and hence, SON management should aim at keeping the PiPo at least within an acceptable range with a high weight.

Note that in the PBSM and ODSM approaches, the objective model usually looks different than in ASM and CSM. In order to get comparable results for the evaluation of the approaches, the same objective model is assumed in each case, meaning that the concept of context classes is also applied for PBSM and ODSM. Furthermore, PBSM can not read an objective model as presented in Listing 8.4. Objectives in PBSM are prioritized instead of weighted and the target is a minimize or maximize indication. However, this model can be easily transformed into an

objective model understandable for PBSM: Each manual context class has three targets which are always the same: *MAXIMIZE_CQI*, *MINIMIZE_PiPo* and *MINIMIZE_CL*. The difference between the classes lies in the priorities assigned to each target. Therefore, the highest weighted target gets the highest priority, the second highest weighted one the second highest priority and so on. In the end, the real KPI targets are taken into account also for PBSM to ensure comparable results in terms of objective fulfillment.

```
1 {
    IF CLASS_1_HIGH THEN CQI ≥ 0.6 WITH 0.6
3   IF CLASS_1_HIGH THEN PiPo ≤ 0.02 WITH 0.1
    IF CLASS_1_HIGH THEN CL ≤ 0.7 WITH 0.3
5   IF CLASS_1_LOW  THEN CQI ≥ 0.55 WITH 0.5
    IF CLASS_1_LOW  THEN PiPo ≤ 0.02 WITH 0.2
7   IF CLASS_1_LOW  THEN CL ≤ 0.4 WITH 0.3
    IF CLASS_2_HIGH THEN CQI ≥ 0.55 WITH 0.5
9   IF CLASS_2_HIGH THEN PiPo ≤ 0.01 WITH 0.2
    IF CLASS_2_HIGH THEN CL ≤ 0.55 WITH 0.3
11  IF CLASS_2_LOW  THEN CQI ≥ 0.5 WITH 0.4
    IF CLASS_2_LOW  THEN PiPo ≤ 0.01 WITH 0.2
13  IF CLASS_2_LOW  THEN CL ≤ 0.3 WITH 0.4
    IF CLASS_3_HIGH THEN CQI ≥ 0.45 WITH 0.3
15  IF CLASS_3_HIGH THEN PiPo ≤ 0.1 WITH 0.6
    IF CLASS_3_HIGH THEN CL ≤ 0.5 WITH 0.1
17  IF CLASS_3_LOW  THEN CQI ≥ 0.45 WITH 0.2
    IF CLASS_3_LOW  THEN PiPo ≤ 0.08 WITH 0.6
19  IF CLASS_3_LOW  THEN CL ≤ 0.3 WITH 0.2
    IF CLASS_4_HIGH THEN CQI ≥ 0.55 WITH 0.6
21  IF CLASS_4_HIGH THEN PiPo ≤ 0.02 WITH 0.1
    IF CLASS_4_HIGH THEN CL ≤ 0.15 WITH 0.3
23  IF CLASS_4_LOW  THEN CQI ≥ 0.55 WITH 0.4
    IF CLASS_4_LOW  THEN PiPo ≤ 0.01 WITH 0.1
25  IF CLASS_4_LOW  THEN CL ≤ 0.1 WITH 0.5
    IF CLASS_5_HIGH THEN CQI ≥ 0.65 WITH 0.2
27  IF CLASS_5_HIGH THEN PiPo ≤ 0.04 WITH 0.2
    IF CLASS_5_HIGH THEN CL ≤ 0.9 WITH 0.6
29 }
```

Listing 8.4: Objective model for manual context classes in the Hamburg scenario

8.2.3. Effect Models

For the effect model, two tasks have to be accomplished: First, manufacturer-provided effect models need to be created in the simulation environment of the SON function manufacturers. Second, an appropriate learning algorithm needs to be selected for each KPI which can then be applied for CSM in the real network. Note that the *generation* of a combined effect model, real network effect model and learned effect model are not part of this chapter. This has already been demonstrated in Chapter 5, Chapter 6 and Chapter 7.

8.2.3.1. Manufacturer Effect Model

In order to generate a manufacturer effect model for each of the three SON functions CCO, MLB and MRO, a variety of SCV sets have been tested per function and in the end, the most relevant ones have been selected, building the respective effect model. Before explaining the results of the simulations, some premises of the experiment need to be clarified which is shown in Table 8.3.

Table 8.3.: Parameter settings for the generation of manufacturer effect models

Category	Parameter	Value(s)
Simulation	Scenario	Helsinki
	Granularity periods	25-100
	Skipped GPs	20-80
SON function	CCO	
	cqi_threshold	0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0
	tilt_stepsize	-10. -5, -2, -1, -0.5, -0.2, -0.1, 0. 0.1, 0.2, 0.5, 1, 2, 5, 10
	MLB	
	load_threshold	0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0
	cio_stepsize	-10. -5, -2, -1, -0.5, -0.2, -0.1, 0. 0.1, 0.2, 0.5, 1, 2, 5, 10
	MRO	
	load_threshold	0.01, 0.02, 0.03, 0.04, 0.05, 0.06, 0.07, 0.08, 0.09, 0.1
	cio_stepsize	-10. -5, -2, -1, -0.5, -0.2, -0.1, 0. 0.1, 0.2, 0.5, 1, 2, 5, 10

The simulations have been performed using the Helsinki scenario which simulates the manufacturers' testing environment. For each SON functions, ten different SCVs for parameter threshold, combined with each of the 15 stepsize SCVs, have been tested, i.e., 150 possible SCV sets per SON function in total. Since the stepsize parameter significantly influences the convergence time, i.e., the time that is needed until the measured KPI values under a certain SCV set do not change any more, the simulated GPs differ dependent on the stepsize. The larger the stepsize (no matter, if negative or positive), the shorter is the convergence time and consequently, the lower is the number of simulated GPs. Since it takes some time until measured KPIs do not change significantly any more, the measurements in the beginning are discarded. The number of discarded measurements again depends on the stepsize. For larger stepsizes, only the measurements of the first 20 GPs are sorted out while for smaller stepsizes the measurements of up to 80 GPs are rejected.

The SCPs of the SON functions are chosen very similar for each of them. On the one hand, for the stepsizes, the same SCVs were tested since this is a normalized

parameter with a value range from -10 to 10. On the other hand, the thresholds for CQI and CL were increased in 0.1 steps from 0.1 to 1.0, all of them offering a meaningful value. For PiPo, the tests started with 0.01 and ended with 0.1. This is due to the fact that everything above 0.1 is a PiPo value that is not desirable, i.e., an operator would never set the target higher than this value.

For each of the tested SCV sets within each SON function, the relevant measurements, i.e., those which were not sorted out, were averaged per KPI, building the effect prediction for this specific set. This has been done for each manual context class individually, i.e., the measurements of cells within each class were aggregated. Therefore, it is first necessary to assign cells to context classes in the Helsinki scenario, as it has been already done for the Hamburg scenario. While this will not be explained in detail here, a meaningful classification is shown in Listing 8.5. This classification is again done by inspection of the network, i.e., the SON function manufacturer divides the set of cells in the manufacturer's simulation environment into reasonable groups where it is assumed that cells have a similar behavior. The only difference to the classification in the Hamburg scenario is that the context classes definitions are already given by the MNO of the real network.

```
{
  CLASS_1 = {site01[0], site01[1], site01[2], site02[2], site05[2],
    site06[0], site06[1], site07[0], site07[2], site08[0], site08[1],
    site09[0], site11[2], site12[1]}
  CLASS_2 = {site03[1], site03[2], site05[0], site05[1], site06[2],
    site07[1], site08[2], site10[0], site12[0], site12[2]}
  CLASS_3 = {site03[0], site04[0], site04[2]}
  CLASS_4 = {site02[0], site02[1], site04[1], site11[0], site11[1]}
  CLASS_5 = {site13[0], site14[0], site15[0]}
}
```

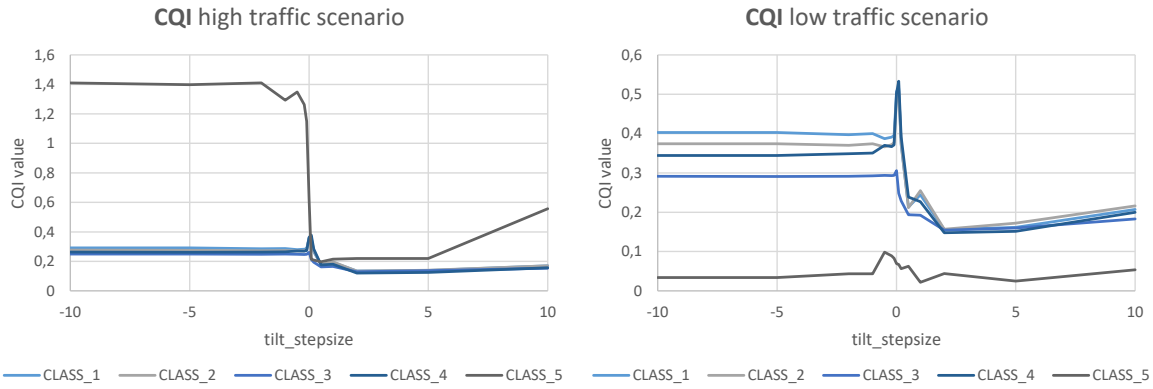
Listing 8.5: Assignment of cells to manual context classes in the Helsinki scenario

The results of these extensive simulations and the selection of SCV sets for the generation of an effect model is shown in the following. Thereby, the focus always is on the KPI that is mainly influenced by the respective SON function, i.e., CQI for CCO, CL for MLB and PiPo for MRO. Note that the resulting initial effect models are not depicted in this section due to its extent. The interested reader can find the detailed results for all context classes in the annex in Table A.1 - Table A.5.

CCO Results

The first thing that can be seen when looking at the simulation data, is that there is nearly no difference at all in all KPIs for different cqi_threshold SCVs. That is, this parameter obviously does not have an effect on the KPI values. The little difference that can be observed, can be treated as random noise. However, the tilt_stepsize has a significant impact on the CQI in both the high traffic and low traffic scenario which is illustrated in Figure 8.5a and Figure 8.5b. Thereby, the

cqi_threshold is always set to 0.3 while another threshold would have shown the same results.



(a) Results for CCO SCV sets in a high traffic scenario

(b) Results for CCO SCV sets in a low traffic scenario

In the high traffic scenario, all macro cells, i.e., CLASS_1, CLASS_2, CLASS_3 and CLASS_4 show a very similar behavior. Only CLASS_5 is far above the CQI values of the context classes containing macro cells. Even though the scale in the y-axis is different in the low traffic scenario, CCO shows a different behavior here: The CQI of micro cells is close to zero which is due to the transmission power set to 0 dB. In contrast to the high traffic scenario, significant differences can be also seen for the macro cell context classes while they all show a similar behavior across the SCV sets.

Another detection is that CCO obviously performs better for negative and very small positive stepsizes which limits the selection of SCV sets to these values. For each SON function, the three best performing SCV sets are chosen, and the results for CCO are depicted in Table 8.4. Since the cqi_threshold does not have a significant impact on the CQI, it can be chosen randomly or, simply the best performing sets when looking at the thousandth . Thereby, *CCO_0* denotes the default SCV set which is relevant for further simulations.

Table 8.4.: Selected SCV sets for CCO

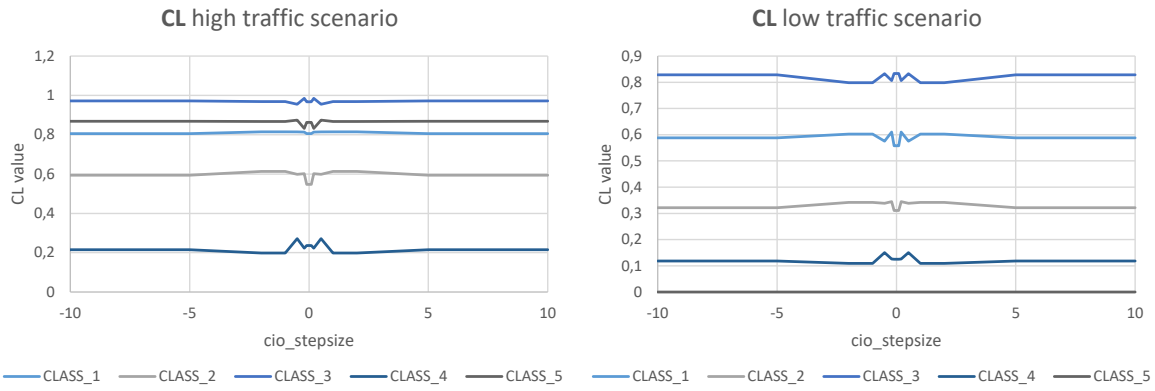
ID	cqi_threshold	tilt_stepsize
CCO_0	0.6	0.1
CCO_1	0.8	0.1
CCO_2	0.8	-0.5

MLB Results

The results for testing different SCV sets in the MLB function are depicted in Figure 8.6a and Figure 8.6b for the high traffic and the low traffic scenario. The underlying load_threshold in both cases is set to 0.3 while, similar to the CCO

function, the effect of this SCP is nearly not noticeable and hence, can be ignored. In contrast to CCO, the effect of MLB on the CL seems quite small. However, MLB always affects only a few UEs and hence, the effect will never be as obvious as in the CCO case where all UEs are affected at the same time by a tilt change.

An obvious detection is the fact, that for small stepsizes, even negative or positive, the load behavior strongly fluctuates and that the behavior is symmetrical for the respective negative and positive value. This applies for both scenarios. Note that in the low traffic scenario, micro cells, i.e., CLASS_5, are again deactivated, resulting in a CL of 0 for all SCV sets.



(a) Results for MLB SCV sets in a high traffic scenario

(b) Results for MLB SCV sets in a low traffic scenario

Another detection is that the CL is lower for all context classes in the low traffic scenario. This is not surprising since the number off UEs is reduced by more than 1/3. Thereby, note that the scale of the y-axis is again different for both scenarios.

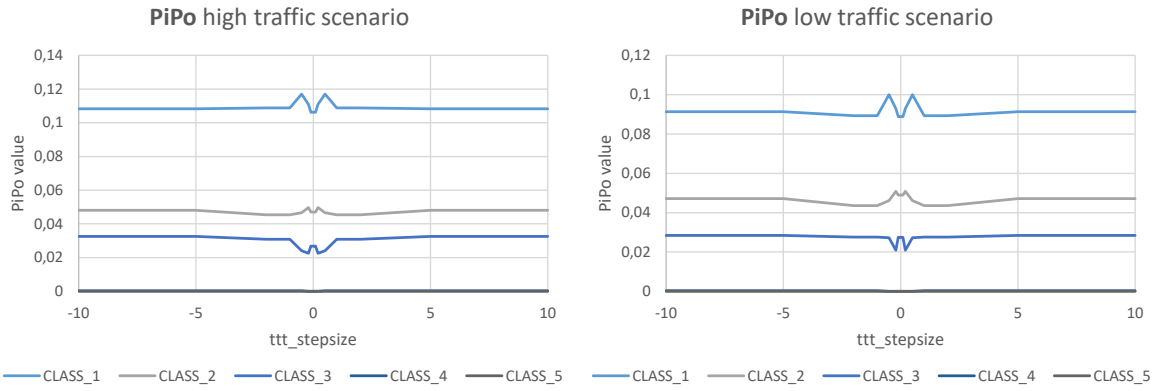
For MLB, the selection of SCV sets is harder than for CCO. This is due to the fact that different context classes show a different behavior for equal stepsizes. For instance, CLASS_4 in the high traffic scenario increases CL for smaller stepsizes and offers a better behavior for larger stepsizes while it is vice versa for CLASS_2. Hence, a balance needs to be found where MLB performs well for, at least, most of the cells. The resulting selection of SCV sets is shown in Table 8.5.

Table 8.5.: Selected SCV sets for MLB

ID	load_threshold	cio_stepsize
MLB_0	0.3	-1
MLB_1	0.3	-0.5
MLB_2	0.3	-2

MRO Results

Finally, MRO is investigated, the results of the simulations can be seen in Figure 8.7a and Figure 8.7b. At first sight, it can be seen that MRO behaves similar to MLB. That is, it is oscillating for smaller stepsizes while being quite constant for larger ones. Again, the pipo_threshold does not have an impact on the PiPo.



(a) Results for MRO SCV sets in a high traffic scenario

(b) Results for MRO SCV sets in a low traffic scenario

In both scenarios, only three of the five context classes actually have ping-pongs. This is an interesting fact for the selection of appropriate SCV sets in the end. Therefor, these two context classes, namely CLASS_4 and CLASS_5, do not to be considered for the selection. Furthermore, both scenarios show a very similar SCV set behavior in terms of the measured PiPo. In the low traffic scenario, the PiPo is nearly as high or low as in the high traffic scenario. This is probably the case since PiPo is a KPI that describes a rate, i.e., it is not influenced by the number of UEs such as CQI and CL.

Since three SCV sets are selected for MRO, the best performing stepsize for each of the three relevant classes can be chosen. For CLASS_1, this is -10. for CLASS_2 it is -2 and -0.2 for CLASS_3. This is shown in Table 8.4, combined with a pipo_-threshold of 0.01 which provides the slighty best performance when comparing the threshold SCVs.

Table 8.6.: Selected SCV sets for MRO

ID	pipo_threshold	ttt_stepsize
MRO_0	0.01	-10
MRO_1	0.01	-2
MRO_2	0.01	-0.2

The results of Section 8.2.3.1, Section 8.2.3.1 and Section 8.2.3.1 can then be used for the generation of a combined effect model which serves as a basis for the real network effect model and hence, also for the learned effect model. Since there

are three possible SCV sets for each SON function and also a deactivated SON function shall be taken into account as a fourth possibility, there are in total $4^3 = 64$ possible combined SCV sets. Note that these combinations have a different effect within each manual context class and hence, the resulting combined effect model consists of $64 * 9 = 576$ rules. Hence, the combined effect model is not part of the annex, but can be easily calculated according to the methodology presented in Section 5.2.3.1.

8.2.3.2. Learned Effect Model

Before beginning with the creation of models and the selection of the best model for learning an effect model, it is worth examining the requirements and premises of the experiment. First of all, it is the goal of the learned effect model to reliably estimate the effect of up-to-now untested combined SCV sets. That is, for the experiment it is assumed that some of the 64 possible SCV set combinations have already been applied in the real network and some have not been tested yet. Second, the effect model shall be learned in the real network, based on real network measurements and hence, the simulations and evaluation is done using the Hamburg scenario. Third, three KPIs are under investigation while it is probable that the different learning algorithms provide different results for them, i.e., the best model is selected for each of the KPIs individually.

Dataset Description
In this paragraph it is explained how the data for learning an effect model are generated and how the raw data look like. Before starting with the gathering of data, one important parameter to determine is the amount of consecutive GPs necessary to be run in order to measure KPIs actually representing the behavior of the SON functions under given SCV sets. In other words: How many GPs are necessary for the SON function to settle?

To start out, the default amount of GPs was set to 100 as a conservative baseline, especially since the SON functions in the experiment were tasked to cope with changing environments within one continuous run of the simulations, therefore having to undergo multiple settling processes. The simulations were run with different combinations of SCV sets and their behavior qualitatively analyzed based on the graphs plotting the KPIs over the whole simulation and based on the measurement database. Generally, the following behavior could be observed: If any settling happens, it does so within the first around 20 rounds of the simulation, and from that point on each cell shows a fairly constant behavior. In certain cases with more aggressive SCV sets, some KPIs never settle at all and instead show a high amount of jitter. In these instances, the SON functions are taking action after each GP, but due to large stepsizes rectify each time, leading to a high variance in the respective KPI. As a consequence, a simulation usually lasted for 50 GPs.

Each of the possible SCV set combinations is then simulated in the Hamburg scenario over these 50 GPs. This has been done for both the low traffic and the high traffic scenario. Thereby, the measurements of the first 30 GPs were sorted out since this is assumed as the time where the SON functions settle. The results are stored in the SON management data base for further calculations. In summary, around 154.000 measurements were gathered where each measurement consists of the values for each of the columns depicted in Table 8.7. The SCV set numbers again refer to the actual SCPs and their values where set number -1 indicates a deactivated SON function. Since each SCV set consists of two SCPs, there are in total six parameters, i.e., features, serving as input for the learning. The KPIs all have a value range of $[0, 1.0]$ whereat a higher CQI and a lower PiPo and CL are preferred.

Table 8.7.: An excerpt of the raw dataframe, each row representing the measurements of a single cell in one GP

GP	Cell Name	CQI	PiPo	CL	CCO	MLB	MRO
31	site01[0]	0.589	0.0115	0.366	0	0	-1
	site01[1]	0.346	0.0129	0.483	0	0	-1
	site01[2]	0.617	0.0333	0.566	0	0	-1
	...	...	...	...	...	...	...
32	...	...	...	...	...	...	...
	...	...	...	...	...	...	...
...	...	...	...	...	...	...	...

When looking at the raw data, a few points can be discovered:

Stability over Time The CQI changes over time, when looking at a particular SCV set, are very low. As shown in Table 8.8 and Table 8.9, for the 20 GPs stored in the database the variance reaches a maximum of 0.0025 for one cell under a specific SCV set in the high traffic scenario while even being less in the low traffic scenario. In most of the cases the variance is even closer to zero. This is expected behavior since the first 30 rounds were used for the SON function to turn into a stable state, i.e., the SON function can not optimize the KPI any more.

For PiPo, the situation is a bit different. Here, it has to be distinguished between three cases: A cell can have no no ping-pongs at all, hence, the PiPo is always 0 and so is the variance. For cells with a low PiPo rate, the interval of PiPos within each SCV set is also quite small, leading to a very low variance. Some cells have a very high PiPo rate > 0.2 and here, the variance can be fairly high, especially in the low traffic scenario. The reason for this has already been described above: Some very aggressive SCV sets with a high stepsize lead to a high amount of jitter. However, this is only

the case for a handful of SCV sets. Over 99% of the SCV set combinations have a quite stable effect on the PiPo with a variance < 0.005.

CL is probably the KPI which fluctuates the most. The difference between minimal and maximal value within one cell between GP 31 and GP 50 can be fairly large, leading to a high variance in both scenarios. This is somehow expected because UEs are permanently moving, thus leaving or joining cells, and hence, the CL permanently changes. Furthermore, even one single UE with a bad Signal to Interference plus Noise Ratio (SINR) value can increase the CL significantly in a certain GP. Since there are also some chronically overloaded cells, the variance sometimes is 0. Referring back to Section 2.3.1, the fit of a model is measured based on the predicted output of $\hat{y}$ compared to the actual output of a sample y. However, any model will only predict a single value $\hat{y}$ based on the input, which in this case would be the combined SCV set. Due to the wide interval of measured $\hat{y}$s for a single combined SCV set the irreducible error will always be very large, therefore making accurate predictions for the CL of a single cell hard.

Difference across SCV Set Combinations This is one of the most crucial detections of investigating the raw data. Since the behavior of untested SCV sets should be predicted in the end, a different behavior of the SON functions under different SCV sets must be presumed. An investigation of the raw data shows that the intervals for all of the KPIs are quite large for the various SCV sets. If this assumption would not have applied, this would have been a show-stopper for the further experiment.

Outliers While for CQI, the number of outliers is quite low, it is in contrast quite high for PiPo and CL. The reasons for this, in case of PiPo, is that there are only a few cells where the PiPo rate is extremely high since only a few UEs determine the KPI value. Similarly, there are some strongly overloaded cells which may have two reasons. First of all, the cell is chronically overloaded due to a very high number of UEs or second, one or a few UEs have an extremely bad SINR value, causing the cell to be overloaded even though the number of UEs is relatively low.

As a consequence, the values for CQI are widely spread without being able to clearly identify outliers. The values for PiPo are clustered between 0 and 0.1 while having a few outliers between 0.1 and 0.5. Finally, the values for CL are widely spread between 0.1 and 0.9 while clearly having a cluster at 1.0.

To sum up the findings to far: Changing SCV set combinations does appear to have a sufficient effect on the performance of the KPIs. Additionally, the measured values show a large variance for each combined SCV set for the KPIs PiPo and CL, making it hard to postulate statements as “under combined SCV set X the cell will have KPI Y”. However, reliably estimating the average performance of a set of KPIs is sufficient since it is the goal of SON management to find the SCV sets for the SON functions that optimally fulfill operator objectives, not to

Table 8.8.: Variances and mean values for simulated SCV sets in the high traffic scenario

	Max. Variance	Min. Variance	Max. Mean	Min. Mean	Difference
CQI	0.0025	0.0000005	0.966	0.109	0.857
PiPo	0.0204	0.0	0.338	0.0	0.338
CL	0.1233	0.0	1.0	0.005	0.995

Table 8.9.: Variances and mean values for simulated SCV sets in the low traffic scenario

	Max. Variance	Min. Variance	Max. Mean	Min. Mean	Difference
CQI	0.0009	0.0	0.959	0.121	0.838
PiPo	0.0592	0.0	1.0	0.0	1.0
CL	0.1370	0.0	0.919	0.0	0.919

predict the exact KPI value under a certain SCV set for a certain GP. Hence, the raw data will be condensed into average KPIs per combined SCV set which is explained in the next paragraph.

Data Preparation
It was shown above that the irreducible error would be fairly large when predicting the performance of a single cell in a single round, leading to bad models due to the large amount of noise. To combat that noise, the high level of detail (KPIs per cell per GP per SCV set) of the data is forgone, and solely the average KPIs per SCV set are aggregated.

Additionally, the learned context model is taken into account. The value of such a model has already been explained in Section 7.2.2.1, i.e., most of all, the reduced computational effort. Instead of calculating a learned effect model on a cell basis, effects are estimated per learned context class. Therefore, the set of cells is clustered into k_{means} clusters where cells behave similar. The k_{means} value thereby can be defined by the MNO. However, for the evaluation of the learned effect model, a variety of possible values shall be investigated. More precisely, all values from $k_{\text{means}} = 1$, i.e., all cells belong to one cluster, to $k_{\text{means}} = 61$, i.e., each cell belongs to its own cluster, are tested.

In Section 7.2.2.2, the generation of a learned effect model has been already described and the premises have been discussed. Accordingly, the set of combined SCV sets is divided into a training and a testing set with a relation of $2:1$. Within the training set, the KPI values are averaged per SCV set combination and per learned context class. In contrast, in the testing set, KPI values are only averaged per SCV set combination and per cell. The latter may look strange at first sight, however becomes meaningful considering the methodology of CSM. In CSM, optimal SCV set combinations are chosen on a cell level, i.e., the effect predictions

must optimally reflect the cells' behavior instead of the behavior of the learned context classes.

Table 8.10 illustrates the averaged data frames and their structure as they are used for the training set, assuming $k_{\text{means}} = 10$. The sample count thereby indicates the number of samples that have been averaged, i.e., 20 (the number of GPs) per cell in a learned context class. For instance, seven cells are assigned to CLASS_1, hence, the sample count is 140.

Table 8.10.: An excerpt of the averaged data frame, each row representing the averaged measurement of a learned context class under a certain SCV set

Learned Class	Sample Count	CCO	MLB	MRO	CQI	PiPo	CL
CLASS_1	140	-1	-1	-1	0.543	0	0.975
	140	-1	-1	1	0.483	0.032	0.943
	140	-1	0	-1	0.327	0.014	0.963
	140	...	...	...	...	...	...
	140	2	2	1	0.517	0.022	0.983
CLASS_2	80	...	...	...	...	...	...
...	...	...	...	...	...	...	...
CLASS_10	120	...	...	...	...	...	...

In the testing set, there are always 20 samples per SCV set, one measurement per GP. These measurements are averaged building the CQI, PiPo and CL values in Table 8.11. Note that the combined SCV sets are split randomly in a relation of $2:1$ and the testing set contains those SCV set combinations which are not available in the training set.

Table 8.11.: An excerpt of the averaged data frame, each row representing the averaged measurement of a cell under a certain SCV set

Testing Cell	Sample Count	CCO	MLB	MRO	CQI	PiPo	CL
site01[0]	20	-1	-1	0	0.453	0.035	0.841
	20	-1	-1	2	0.683	0.033	0.790
	20	-1	0	0	0.581	0.002	0.822
	20	...	...	...	...	...	...
	20	2	2	2	0.444	0.046	0.753
site01[1]	20	...	...	...	...	...	...
...	...	...	...	...	...	...	...
site25[0]	20	...	...	...	...	...	...

For the application of learning algorithms, Weka [WEK19] is used which is an open source collection of machine learning algorithms. In Weka, data is provided

in the Attribute-Relation File Format (ARFF) format consisting of a header which contains the attributes of the data, i.e., the SCPs and the KPI value to be predicted, and the data, i.e., the actual SCVs and their values of the observed KPI. Thus, a huge set of ARFF files is generated, one training and one testing file for each learned context class, each k_{means} between 1 and 61 and for each single KPI. Four different learning algorithms with several configurations are then trained on each of the training files and the resulting models tested on the testing files. Finally, the models can be evaluated in terms of the RMSE in order to find the one which best fits the testing set.

Therefore, each learned context class was fitted with different models of different learning algorithms, mapping the SCVs to the respective KPI. Each model was trained on the training data averages, and its performance measured against the training labels, following the LOO cross validation protocol. RMSE is used as error metric to compare the performances of different models against each other. Based on these findings the best performing model for each case, i.e, each learned context class, each k_{means} and each KPI can be selected.

In the following, the results for these algorithms, i.e., LR, GPR, KNN and ANN are presented. Thereby, the focus will be on evaluating the KPI CQI in both the high and low traffic scenario. The results for the KPIs PiPo and CL will also be described in the following, however, due to their amount and size, the graphs are part of the annex.

Linear Regression Results
As explained in Section 2.3.1.1, the LR model can be made increasingly more flexible by extending the input vector with polynomials and combinations of the already existing features. The potential drawbacks of an overly flexible model were illustrated in Section 2.3.1. The flexibility was increased step by step from a linear model up to a cubic model with cubic interaction terms, such that nine different models were created for each of the k_{means} learned context classes. Figure 8.8 shows how the different models behave with increasing levels of flexibility for CQI in a high traffic scenario and Figure 8.9 shows the respective behavior in the low traffic scenario. Thereby, the RMSE of each k_{means} indicates the average RMSE over all learned context classes. That is, the overall RMSE is calculated by building the weighted average over the RMSEs of the particular learned context classes, taking the number of cells within each learned context class into account.

First of all, it can be detected that the high traffic and the low traffic scenario behave quite similar, i.e., a model that is performing well in the high traffic scenario, also does so in the low traffic scenario, and analogously for bad performing models. This applies for all of the three KPIs. Second, the RMSE decreases more or less constantly for higher k_{means}s which is expected since for $k_{\text{means}} = 1$, i.e., all cells are classified into one cluster, and hence, the difference in their CQI values is fairly large. In contrast, the higher the k_{means}, the more similar are cells within each cluster with respect to their KPI behavior and consequently, the irreducible

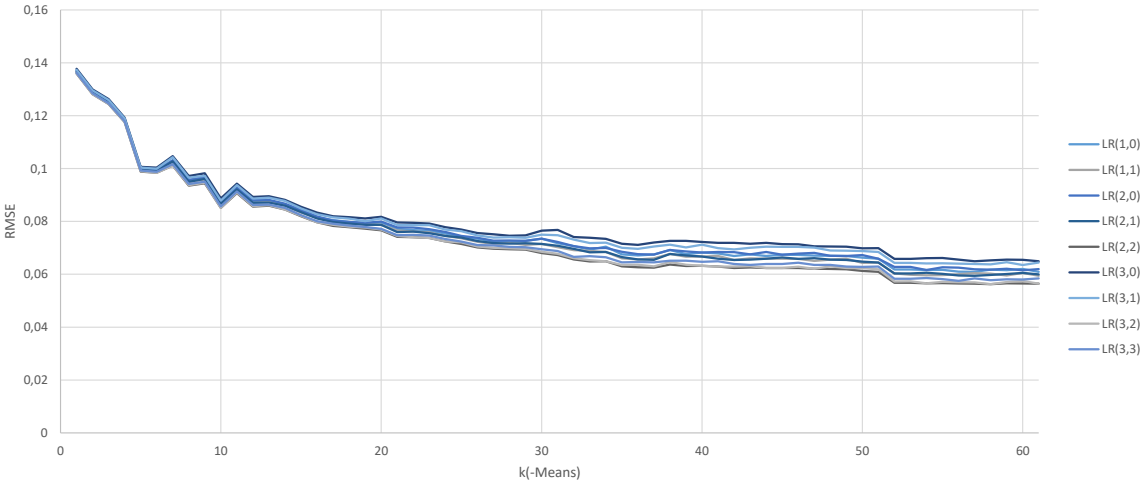


Figure 8.8.: LR: Results in terms of RMSE for KPI CQI in a high traffic scenario

error becomes lower and, accordingly, the average RMSE. Even though it provides the best results, choosing the k_{means} as high as possible is not the right way. The disadvantages with respect to computational effort have already been discussed and hence, a balance needs to be found where both the resulting RMSE and the computational effort are in an acceptable range.

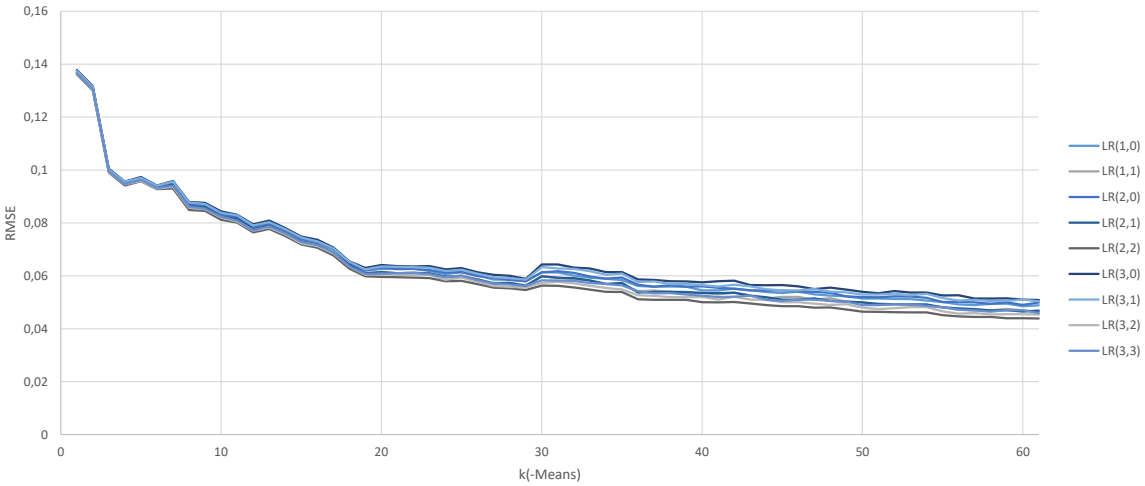


Figure 8.9.: LR: Results in terms of RMSE for KPI CQI in a high traffic scenario

When comparing the different models with each other, it can be detected that the *quadratic model with interaction squared* is the best performing LR model. It is important to remember that a more flexible model with higher polynomials of the input vector does not necessarily provide better results. As a matter of fact, Figure 8.8 and Figure 8.9 show that the more flexible models partially have worse performance on the training data, which is due to the bias-variance trade-off explained in Section 2.3.1. Finally, it is interesting to see that in general, PiPo

and CL show a similar behavior as CQI, with the difference that the *cubed model with interaction squared* performs best for both KPIs.

Gaussian Process Regression Results
In GPR, the performance of the model can be influenced by using different kernel functions. Three kernel functions have been tested, more precisely, the ones that have been presented in Section 2.3.1.3, i.e., a polynomial kernel function, an RBF and a PUK function. The RMSE is thereby calculated in the same way as for the LR models. The result of this process is shown in Figure 8.10 for the high traffic scenario and in Figure 8.11 for the low traffic scenario.

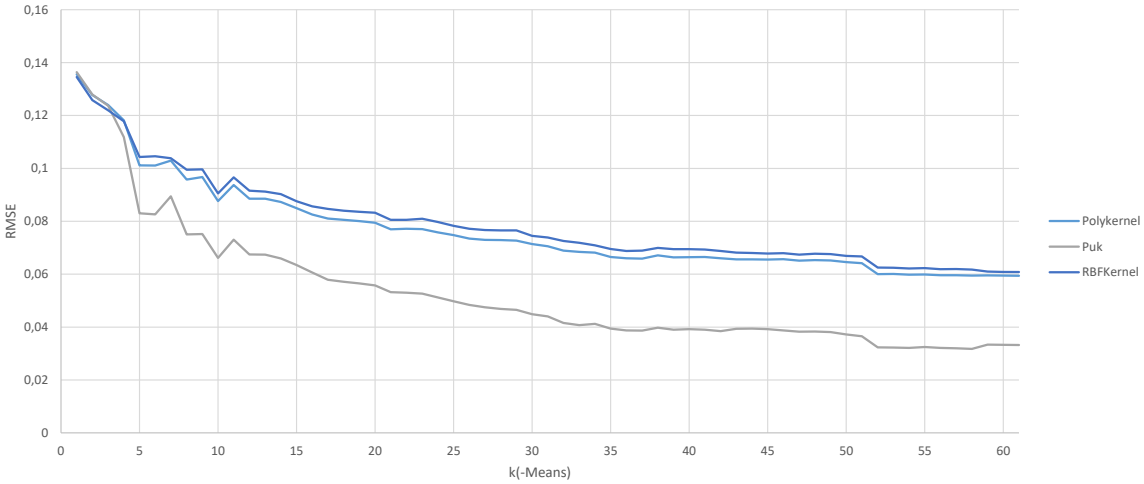


Figure 8.10.: GPR: Results in terms of RMSE for KPI CQI in a high traffic scenario

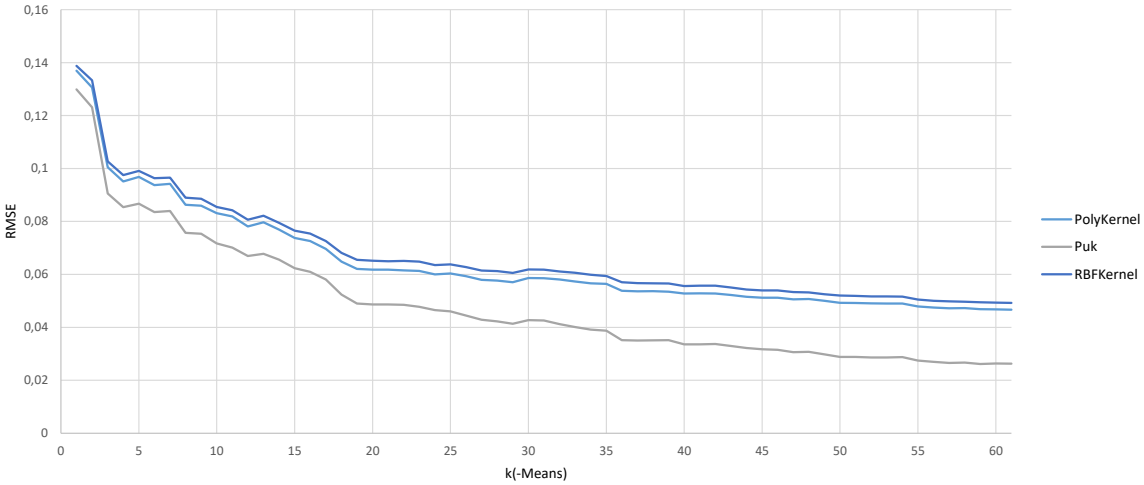


Figure 8.11.: GPR: Results in terms of RMSE for KPI CQI in a low traffic scenario

The first detection is that the *PUK function* obviously performs best in both scenarios while the difference to the polynomial kernel function and the RBF is slightly

smaller in the low traffic scenario. It is also quite clear that the RMSE successively decreases for higher k_{means}s, the reasons for that have already been explained in the previous paragraph. Having a very low k_{means} makes the models unusable for a reliable estimation of KPI effects under untested SCV sets. A meaningful value for k_{means} is discussed in Section 8.2.3.2 where the RMSEs are compared with the differences between the SCV sets, thereby assessing the quality of the models.

For PiPo, the different models perform similar as for CQI, however, the difference between the kernel functions just gets apparent from $k_{\text{means}} = 9$ on, i.e., for lower k_{means}s, the models have a nearly equal performance. For CL, the huge discrepancy between lower and higher k_{means}s is remarkable. This may be caused by the unsteady behavior of the CL which is disproportionately stronger when cells are thrown together in a small number of clusters.

k-Nearest Neighbors Results

The next algorithm to be evaluated is KNN. Thereby, the crucial part is the selection of an appropriate k_{nearest} which is not to be confused with the number of clusters k_{means}. In general, a guideline for choosing the right k_{nearest} is setting it to the square root of the number of training samples which is 2/3 of all possible SCV set combinations, i.e., 42. Consequently, setting $k_{\text{nearest}} \sim 6$ is a good starting point. Since this only serves as a guideline, an interval of $k_{\text{nearest}} \in [4, 8]$ is tested. The results of doing this can be seen in Figure 8.12 for the high traffic scenario and in Figure 8.13 for the low traffic scenario.

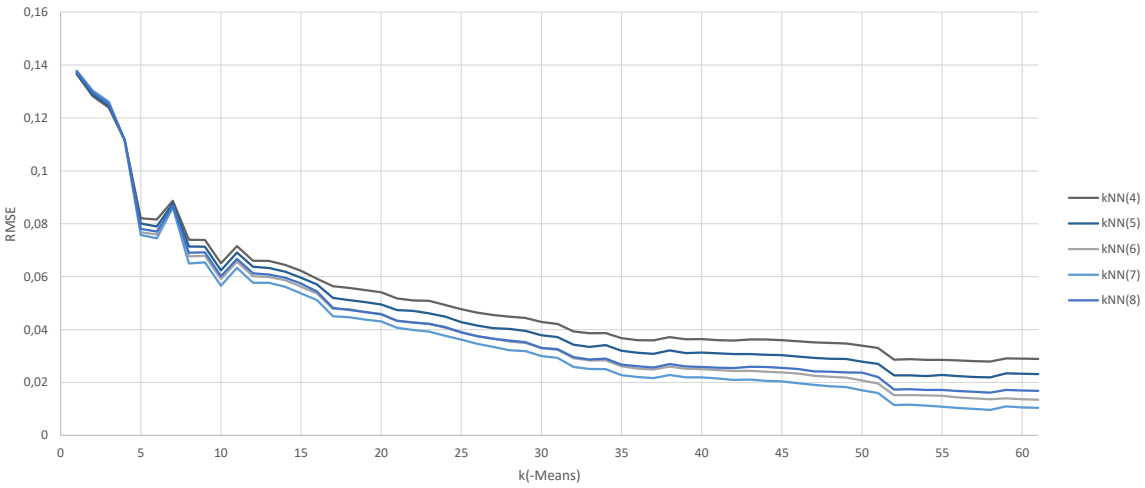


Figure 8.12.: KNN: Results in terms of RMSE for KPI CQI in a high traffic scenario

Interpreting the graphs, the nearly parallel structure of the curves becomes apparent as well as, again, the significant differences between lower and higher k_{means}s. This is the case for both scenarios and all three KPIs. It can be seen that $k_{\text{nearest}} = 7$ performs best and that the performance worsens, the more far away the k_{nearest} is from 7. It is also interesting to see that the performance for PiPo and

CL in the low traffic scenario is very similar up to a fairly high k_{means}. However, the selection of $k_{\text{nearest}} = 7$ is obvious, because this is the best performing model in all observed cases.

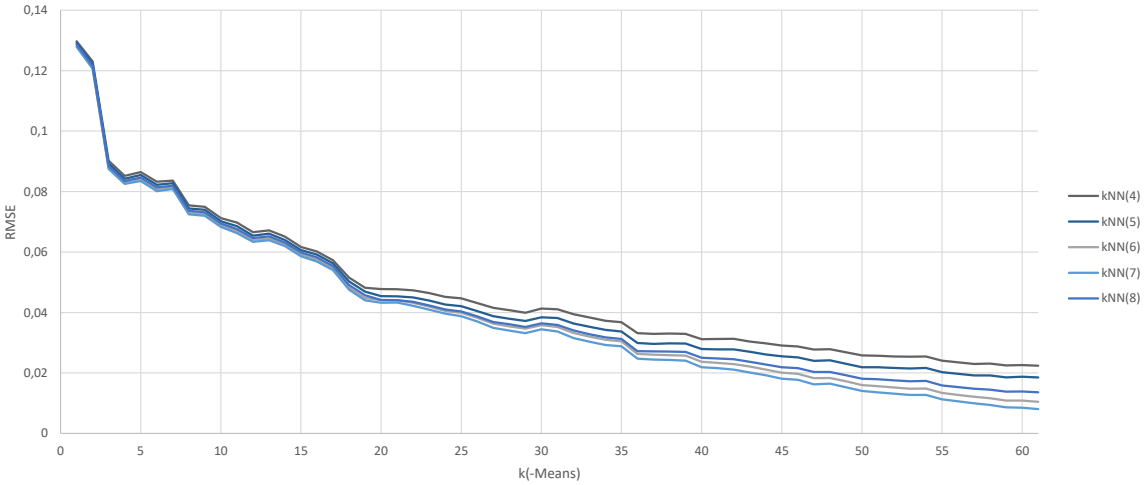


Figure 8.13.: KNN: Results in terms of RMSE for KPI CQI in a low traffic scenario

Artificial Neural Network Results
Finally, different ANN models were evaluated. As explained in Section 2.3.1.4, the crucial part thereby is the choice of the number of hidden layers and the number of hidden neurons on each layer. It has been said that the number of hidden layers usually should not exceed two. Choosing the number of hidden neurons is more complex, since several guidelines exist for that. One of them which is widely used, is to choose the number of neurons on a hidden layer between the size of its input and output layer. Following these rules, the configurations which are depicted in the explanation of Figure 8.14, were tested. Each number thereby represents a hidden layer and the number itself defines the number of hidden neurons on that layer.

The results for the nine tested models are illustrated in Figure 8.14 and Figure 8.15 for the two well-known scenarios. A few things can be seen at first sight: First of all, the results are significantly worse for just one hidden layer, no matter what the number of neurons on this layer is. Second, the performance for models with two hidden layers is more or less the same for all cases. Basically, no difference can be seen at all when looking at the charts for CQI and also CL. A slight difference can be detected for PiPo in the low traffic scenario, making the model with two hidden layers, five hidden neurons on the first and two on the second layer respectively, the one to be preferred.

Selection of Best Algorithm
In this paragraph, the best performing models for each algorithm are compared in order to select the one for each KPI which best reflects the behavior of the

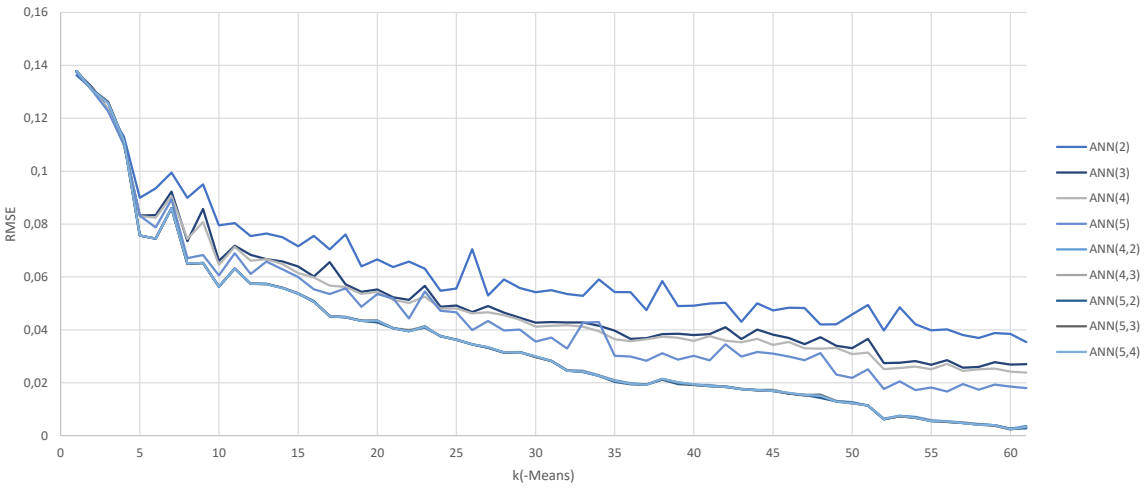


Figure 8.14.: ANN: Results in terms of RMSE for KPI CQI in a high traffic scenario

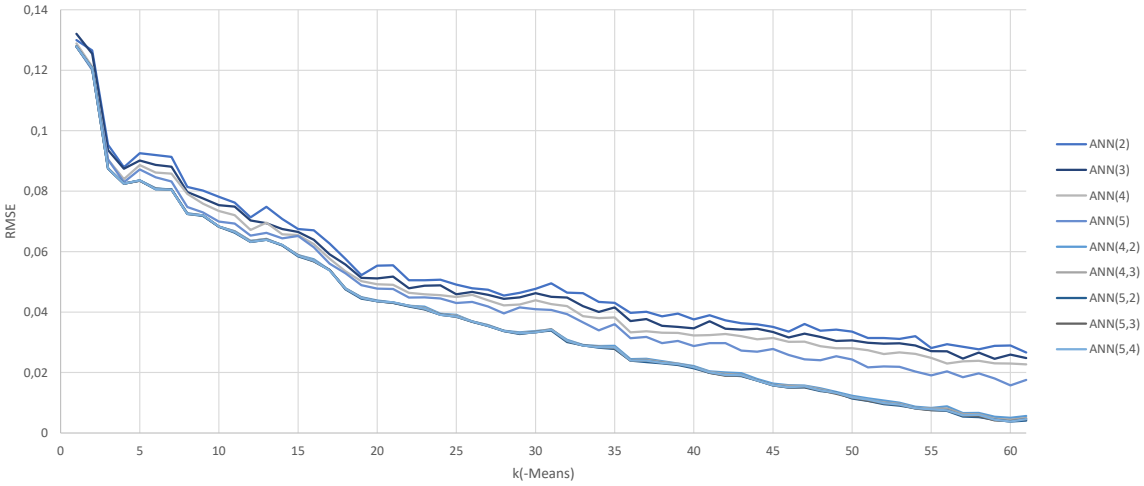


Figure 8.15.: ANN: Results in terms of RMSE for KPI CQI in a low traffic scenario

SON functions. These models are then applied within CSM. Since all algorithms provide the same results for the high traffic scenario and the low traffic one respectively, i.e., the selected model is always the same in both cases, only the high traffic scenario will be considered in the following. The respective graphs of the low traffic scenario can be found in the annex in Figure A.17, Figure A.19 and Figure A.18.

The overall results for CQI are illustrated in Figure 8.16. Thereby, the results of the ANN model and the KNN model look promising while those of GPR and LR provide a fairly high error rate. The decision for one of the models is taken based on the results of higher k_{means}s: Here, the ANN model clearly performs better while only being worse than the KNN model for $k_{\text{means}} = 23$. Consequently, the ANN model with two hidden layers, having five and two hidden neurons respectively, is the preferable one for KPI CQI.

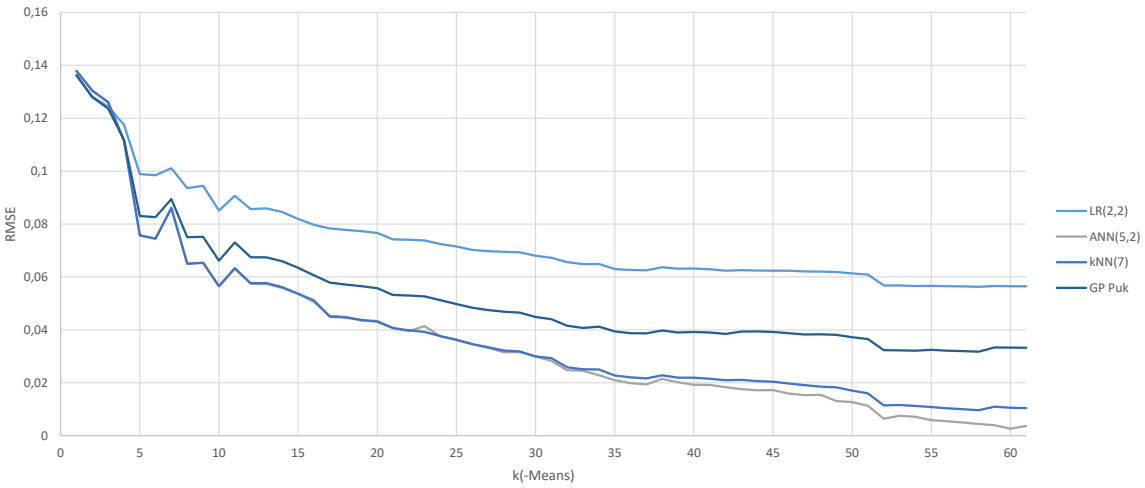


Figure 8.16.: Overall results in terms of RMSE for KPI CQI in a high traffic scenario

The overall results for PiPo are illustrated in Figure 8.17. Here, it is harder to come to a decision. While for $k_{\text{means}} \geq 11$, the ANN model is again the best performing one, for lower k_{means}s the KNN model better estimates the PiPo effects while loosing against the ANN model and the GPR model for higher k_{means}s. Thus, the decision for one the two models must be taken based on the choice of the k_{means}.

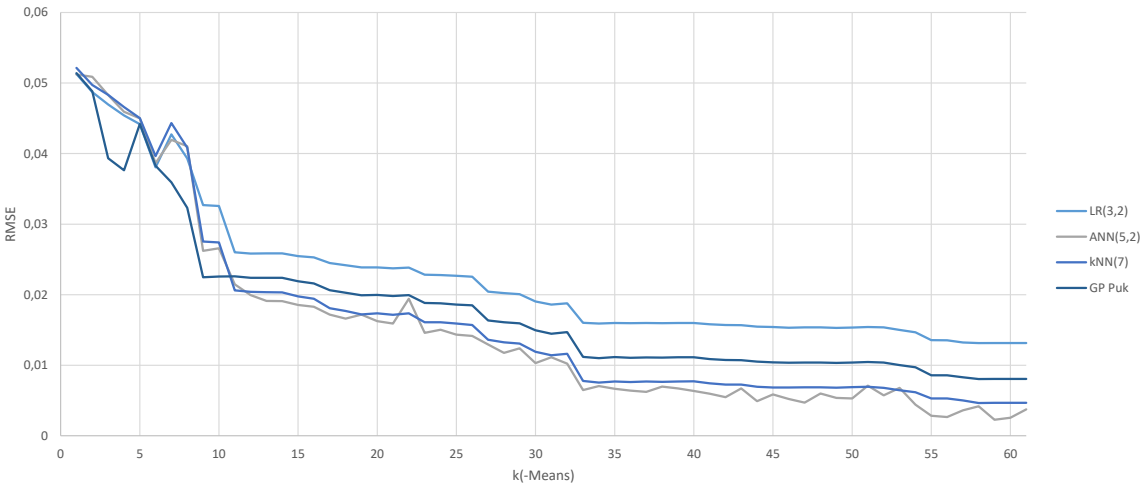


Figure 8.17.: Overall results in terms of RMSE for KPI PiPo in a high traffic scenario

The overall results for CL are illustrated in Figure 8.18. While providing more or less the same results for $k_{\text{means}} \leq 6$, it has to be differentiated for higher k_{means}s: For $7 \leq k_{\text{means}} \leq 28$, the KNN model is the one to be chosen. From $k_{\text{means}} = 28$ to $k_{\text{means}} = 42$, the KNN model and the ANN model have the same RMSEs while

afterwards, i.e., $k_{\text{means}} \geq 43$, ANN is clearly the winning model. Consequently, the decision is again reached by taking the chosen k_{means} into account.

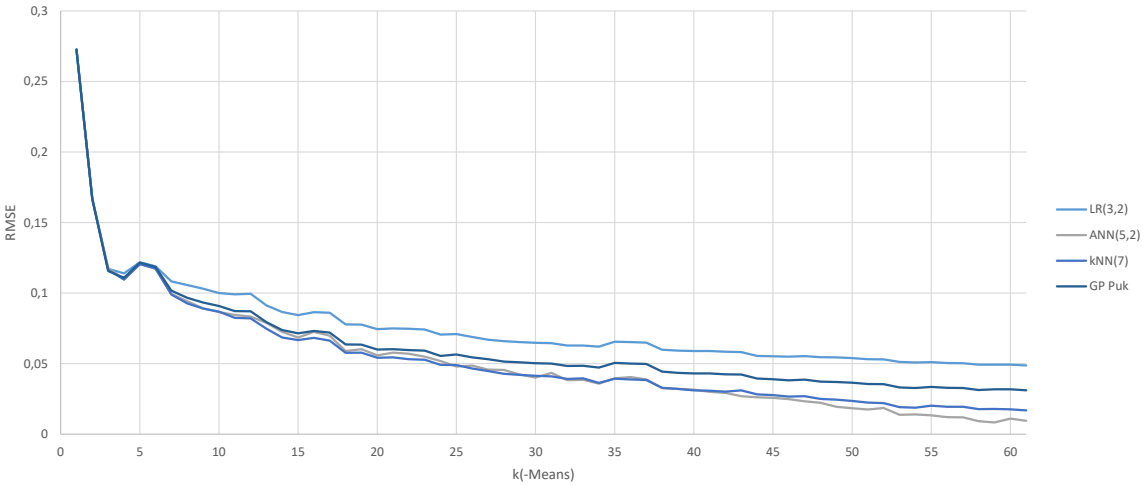


Figure 8.18.: Overall results in terms of RMSE for KPI CL in a high traffic scenario

What all these results have in common, is that LR is the worst performing algorithm in basically every case, i.e., for each k_{means} and every KPI. This implies that the underlying relationship between the thresholds and stepsizes of the different SON functions are by no means a linear combination and instead, follow a more chaotic pattern. This is the reason why non-parametric regression methods such as GPR, KNN and ANN are able to provide a better performance.

Discussion of Results

In the previous paragraphs, the model (or models respectively) for each KPI has been selected that is best in predicting the effects on a specific KPI, i.e., which has the lowest RMSE. However, no investigation has been made, in how far the quality of these models is sufficient, i.e.: Is the prediction of KPI effects under a certain SCV set combination reliable enough to be used in the real network? Furthermore, what is a meaningful choice for k_{means}? It could be seen that the RMSE decreases with a higher k_{means}. However, a trade-off must be found between computational effort and resulting RMSE. These questions are investigated in this paragraph. Therefore, the RMSEs values are compared with the overall differences in the respective KPIs in order to be able to estimate the size of the error.

CQI In both scenarios, CQI has a fairly wide range of measured values, the minimum close to the worst possible value and the maximum close to the best possible one, resulting in a big difference for different cells under different SCV set combinations. Additionally, the values are distributed quite uniformly over the whole interval, i.e., there are basically no outliers. When looking at the overall results in Figure 8.16 and Figure A.17, it can be seen

that the results improve significantly from $k_{\text{means}} = 1$ to $k_{\text{means}} = 20$ and the difference between following k_{means}s gets constantly lower. In both cases, the RMSE value is a bit over 0.04 whereby the KNN and ANN model both provide the same performance. This corresponds to an error rate of $\approx 5\%$ which is in an acceptable range. At the same time, this means that the computational effort for the prediction of KPIs of untested SCV sets is reduced by $\approx 2/3$ compared to a cell-based calculation.

PiPo For PiPo, the situation is quite different. While the difference in the high traffic scenario is fairly low, i.e., 0.338, it reaches the maximum in the low traffic scenario, i.e., 1.0. However, only a handful of cells under a few SCV sets reach values ≥ 0.25. The rest can be seen as outliers and hence, for the decision for a k_{means}, a maximum difference of ≈ 0.25 is assumed. Regarding the high traffic scenario, the RMSE values significantly decrease until $k_{\text{means}} = 33$ while being quite stable for larger k_{means}s. In the low traffic scenario, the same applies for $k_{\text{means}} = 40$, the reason why these two k_{means}s should be chosen for the respective scenario. In both scenarios, the ANN model provides the best performance, an error rate of ≈ 0.007 in the high traffic scenario and ≈ 0.008 in the low traffic scenario. This corresponds to a ratio of around 2% and 3% respectively which is perfectly satisfying. Accordingly, the computational effort can be reduced by 1/2 in the high traffic and 1/3 in the low traffic scenario respectively compared to a cell-based calculation.

CL The KPI CL features the biggest range of the values it can take on. At the same time, no outliers can be clearly identified and hence, the differences as illustrated in Table 8.8 and Table 8.9 are assumed for further calculations. For the high traffic scenario, the RMSE values are ≤ 0.05 and hence, are in an acceptable area for $k_{\text{means}} \geq 25$. Consequently, this refers to an error rate of $\leq 5\%$. Assuming 5% as satisfactory, the same applies for the low traffic scenario for $k_{\text{means}} \geq 14$. The computational effort thereby can be reduced by $\approx 40\%$ and $\approx 75\%$ respectively. In both cases, the selected KNN and ANN model perform similarly.

The discussion in the previous paragraph only demonstrates how an appropriate k_{means} could be selected. However, in the end, the decision for a specific k_{means} is taken by the MNO and can also be changed during operation of the network at any time. Thereby, the selection of a certain k_{means} and the definition of what is seen as an *acceptable* error rate strongly depends on the MNO's preferences and the goals to be achieved in the network. For the presented Hamburg scenario – taking all KPIs into account – a k_{means} between 20 and 30 seems to be a good trade-off between providing satisfactory effect predictions and keeping the computational effort in an acceptable range. For the following evaluation of the different SON management approaches, k_{means} is set to 25 and the ANN model is selected for all KPIs in the CSM case.

8.3. Results of SON Management Approaches

The previous sections dealt with the generation and derivation of the SON management's input models and the selection of appropriate learning algorithms. What is still missing, is the proof that SON management actually helps to fulfill operator objectives and that in particular CSM achieves meaningful results. Therefore, above mentioned input models are used as input to SON management and a simulation is run for each of the presented SON management approaches, i.e., PBSM, ODSM, ASM and CSM. For all cases, the real network scenario, i.e., the Hamburg scenario, is used and, in order to guarantee comparability of the results, the same objective model (cf. Listing 8.4), manual context classes model (cf. Listing 8.3) and initial effect model (cf. Table A.1 - Table A.5) are applied. In addition to the approaches presented in this thesis, a default SON case serves as a baseline which reflects the current situation in mobile networks: The uniform, network-wide configuration of SON functions with default SCV sets. It is the minimum requirement that SON management is able to outperform the results of this simulation. The default SCV sets have been determined in Section 8.2.3.1 and are summarized in Table 8.12.

Table 8.12.: Default SCV sets for three deployed SON functions

ID	threshold	stepsize
CCO_0	0.6	0.1
MLB_0	0.3	-1
MRO_0	0.01	-10

All simulations have the same length, more precisely 600 GPs. They all start with the high traffic scenario and switch to the low traffic scenario after exactly 100 GPs. After additional 100 GPs it switches back to the high traffic scenario and so on, such that in the end, 300 GPs have been simulated in both the high and low traffic scenario.

Since it is one of the main objectives of this thesis to develop a SON management system that is not just theoretically working but also usable in practice, the focus in the evaluation is on the comparison of the fulfillment of operator objectives and the improvement of KPI values. During the discussions in the SEMAFOUR project it became apparent that operators are usually paid according to the fulfillment of their goals and hence, *usable* in this context means that the degree of fulfillment should increase from the baseline, i.e., the current situation, to CSM. Thereby, it is meaningful to evaluate the results for the particular manual context classes since the objective definitions are based on them. In the following, the results are presented for each of the five manual context classes, and thereby, two metrics are chosen for the evaluation:

Average Utility This metric indicates the average of all cells' utilities in the respective manual context class. That is, the weights for all KPIs that a cell

fulfills, are summed up and the arithmetic mean is built over all cells in the context class. Consequently, this value must be between 0 (none of the cells fulfills any KPI target) and 1 (all cells have achieved all KPI targets).

Percentage Target Fulfillment Since the average utility is based on a binary fulfillment of the KPI targets, an additional metric is taken into account, the percentage target fulfillment. Usually, not all cells fulfill all targets and hence, this metric gives an indication if the respective SON management approach was able to, at least, come closer to fulfilling the targets. Therefore, the KPI weights are multiplied with the degree of fulfillment (0 for the lowest possible KPI value and 1, if the KPI value is equal or better than the KPI target) and summed up for all KPIs. Afterwards, the arithmetic mean for all cells within the respective manual context class is built.

While the average utility represents the actual objective fulfillment, the percentage target fulfillment gives a more precise indication whether SON management has an effect on the KPIs or not. Note that the graphs of these two metrics may look fairly different: It may be the case that the percentage target fulfillment increases significantly from one GP to another while on the other hand, the average utility remains constant or even decreases. For instance, when having a very bad CQI value and a PiPo value that just fulfills the target in one GP, the utility in this round equals the weight of the PiPo target. In the next GP, imagine a CQI and a PiPo value, that both come very close to their KPI targets but do not fulfill them. Then, the utility is 0 while the percentage target fulfillment is close to 1.

While the baseline configures the network uniformly with default SCV sets over the whole simulation time, the PBSM case uses the approach as presented in Chapter 4 and the ODSM case the approach as presented in Chapter 5 over the whole simulation time. In contrast, ASM uses the ODSM approach during the first 200 GPs and switches to the ASM approach for the last 400 GPs. Finally, CSM uses ODSM for the first 200 GPs, ASM for GP 201 to 400 and the CSM approach for GP 401 to 600. These facts are summarized in the following table and apply for the evaluation of all context classes.

Table 8.13.: Applied approaches over the simulation time

Case	GP 1 - GP 200	GP 201 - GP 400	GP 401 - GP 600
Baseline	Default	Default	Default
PBSM	PBSM	PBSM	PBSM
ODSM	ODSM	ODSM	ODSM
ASM	ODSM	ASM	ASM
CSM	ODSM	ASM	CSM

When switching from one scenario to the other one, the number of users as well as any other changes of parameters in the scenario, happen instantly. Thereby, it can be easily investigated how many GPs the particular SON management approaches need to settle down in terms of the measured KPI values.

8.3.1. Results for City Center Macro Cells

The first context class under investigation is CLASS_1 which refers to macro cells in the Hamburg city center. In matters of this context class, none of the SON management approaches can achieve a higher average utility than around 0.35. This is due to the fact that none of the cells can fulfill the CQI target which has a weight of 0.6 in the high traffic scenario and 0.5 in the low traffic scenario respectively. The second observation is that all five cases more or less reach the same average utility within the first 200 GPs while the percentage target fulfillment offers that there is already a little difference. However, this difference can be justified by a random better selection of SCV sets since, e.g., ASM and CSM are both using the ODSM approach in this time frame and hence, the same combined effect model.

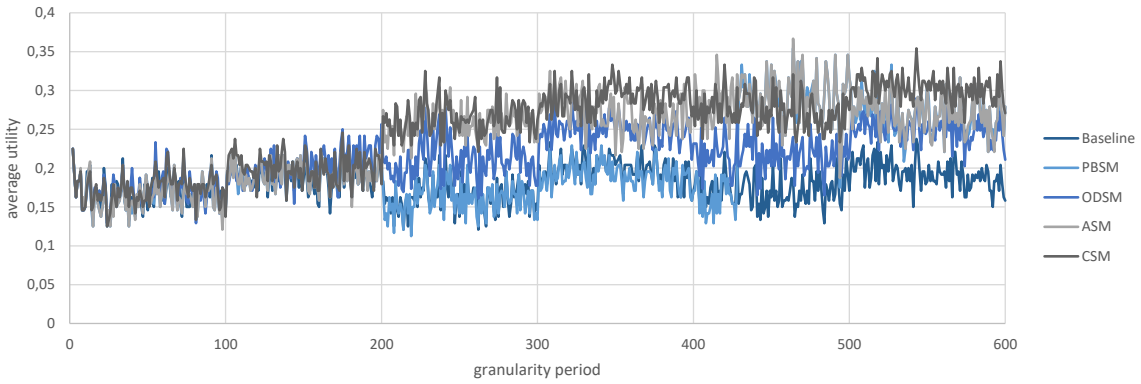


Figure 8.19.: Average utilities for manual context class CLASS_1

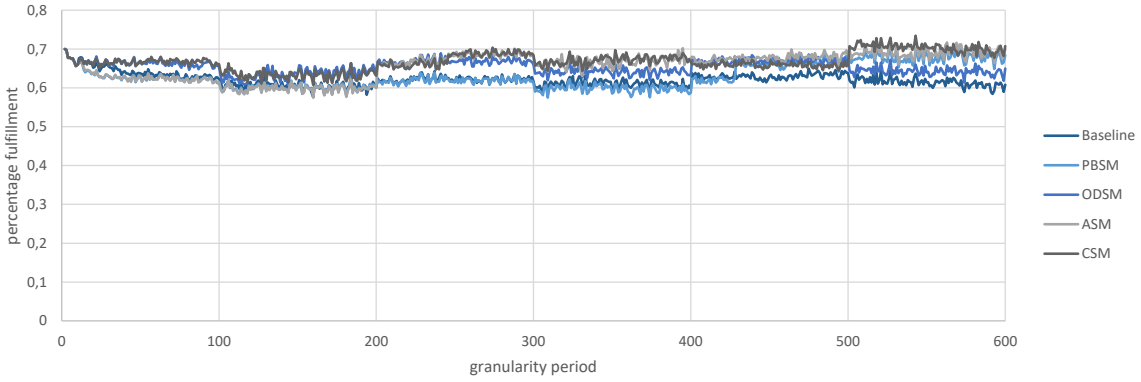


Figure 8.20.: Percentage target fulfillment for manual context class CLASS_1

The difference between the different stages of SON management becomes apparent in the further GPs. First of all, there is a significant difference in the average

objective fulfillment. While the baseline remains more or less constant over the whole simulation time, ASM and CSM easily outperform the baseline, but also ODSM and, most of the time, also PBSM. The outcome of these two cases is very similar between GP 201 and 500. For the last 100 GPs, i.e., in the low traffic scenario, CSM is able to find an up-to-now untested SCV set that performs even better than ASM such that both the average utility and the percentage target fulfillment increase.

8.3.2. Results for Industrial Area Macro Cells

The second context class, i.e., CLASS_2, refers to cells in the industrial Hamburg harbor area. In general, the same observations as for CLASS_1 can be made also for this context class: The baseline results remain more or less constant over the whole simulation time which is quite obvious due to the same SCV set deployed at any time. The same applies for the PBSM approach, meaning that it was not able to find optimal SCV sets that perform better than the default sets. This may have several reasons: First of all, PBSM does not consider combined SCV sets and second, it is only based on the quite unrealistic manufacturer effect model which may provide erroneous effect predictions.

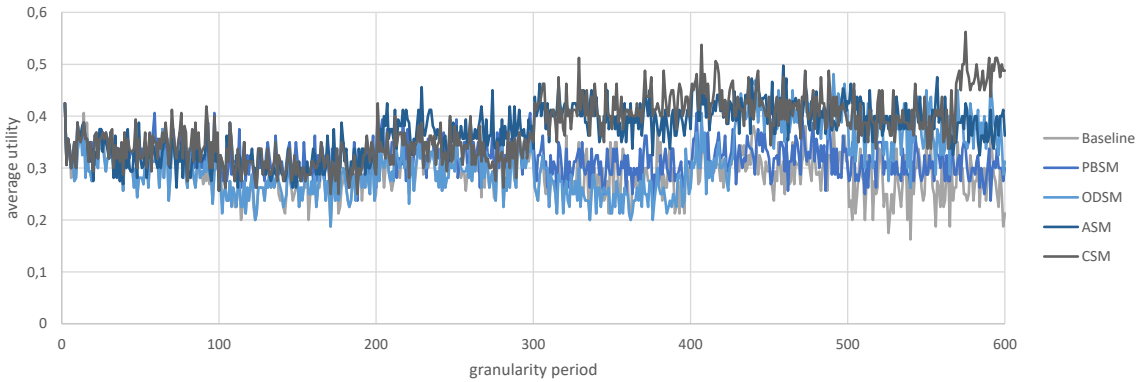


Figure 8.21.: Average utilities for manual context class CLASS_2

Starting from GP 301, ASM and CSM behave quite similar which can be justified by the fact that they work similar in case that the learned effects of the untested SCV sets do not lead to an optimal target fulfillment. However, in the last low traffic period, more precisely at GP 570, the average utility suddenly increases clearly. Here, CSM could identify an untested combined SCV set that indeed does better in fulfilling the operator objectives and also increases the percentage target fulfillment. It may be not obvious why this happens at such a late GP. In fact, CSM gathers data over the whole simulation and continuously adapts the real network effect model which serves as a basis for predicting effects in the learned effect model. These changes in the real network effect model also influence the

predicted fulfillment of operator objectives and hence, it may take some time until an untested SCV set combination can be identified as *optimal*.

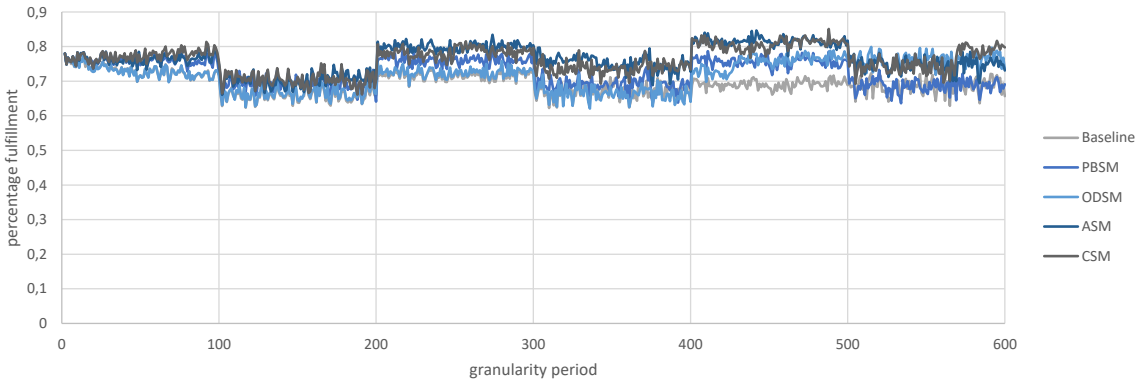


Figure 8.22.: Percentage target fulfillment for manual context class CLASS_2

An interesting fact that applies for all five cases under investigation is that the percentage target fulfillment is fairly high compared to the average utility. It has been already stated in Section 8.3 that this may occur when a lot of cells are close to fulfilling one or more of their targets which, in case of CLASS_2, is again the CQI target with a weight of 0.5 and 0.4 respectively. The high CQI weights also cause the sudden increase for the last 30 GPs in the CSM case: CSM was able to find an SCV set where at least a few of the 16 cells in CLASS_2 fulfill the CQI target.

8.3.3. Results for Highway Macro Cells

CLASS_3 covers macro cells which are located close to the highway. At first sight, it can be seen that both, the average utility as well as the percentage target fulfillment, are very high compared to CLASS_1 and CLASS_2. This is due to the fact that PiPo has a high importance in both scenarios and all of the four cells fulfill the PiPo target at any time.

The ODSM, ASM and CSM all calculate the same optimal combined SCV set within the first 30 GPs, however, they do this at different points in time. Here, one of the main advantages of the ODSM approach becomes apparent: Usually, more than one SCV set combinations provide the same best utility (according to the objective manager's calculations) such that the objective manager selects one of these SCV set combinations randomly, leading to a big diversity of tested combinations. Even though it may take some time until the optimal configuration has been found, this variety of SCV sets is useful at a later point in time for the ASM and CSM calculations since a lot of data is available in the real network effect model and hence, also for the generation of the learned effect model.

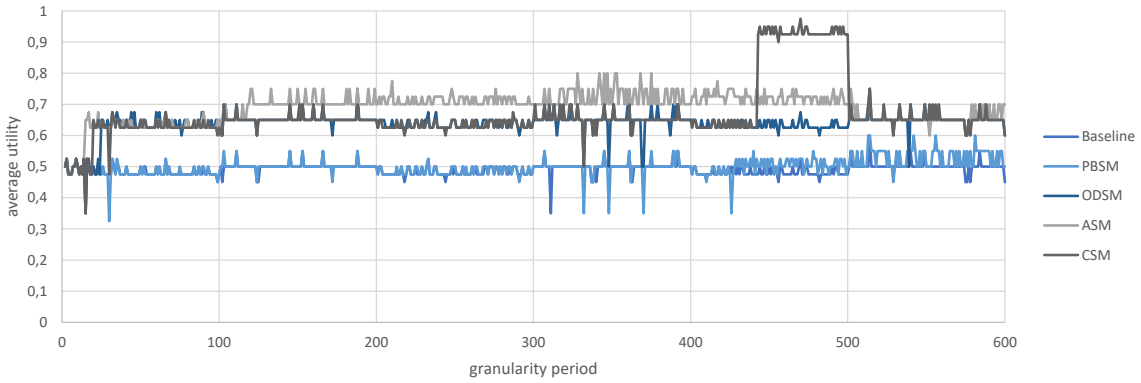


Figure 8.23.: Average utilities for manual context class CLASS_3

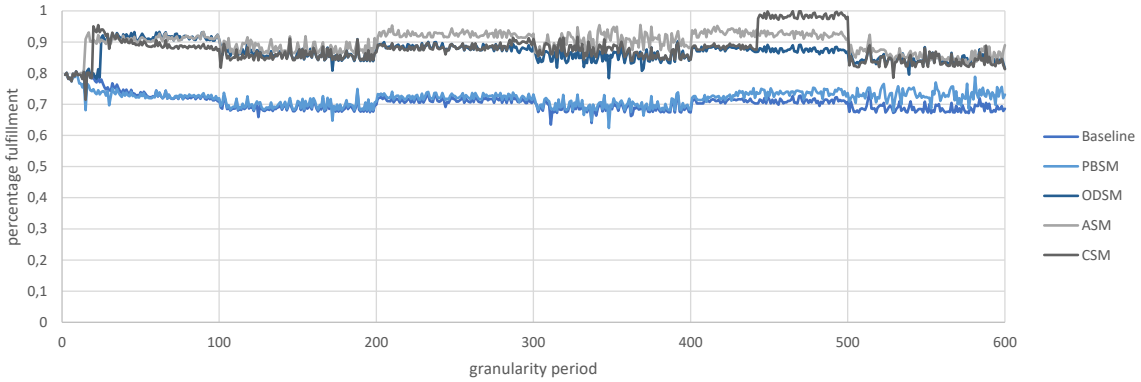


Figure 8.24.: Percentage target fulfillment for manual context class CLASS_3

In contrast to previously described context classes, ASM can clearly outperform CSM except for GP 440 to GP 500, even though both cases use the ASM approach for most of the time. While for the indicated time frame, CSM managed to find not only the best possible, but nearly the optimal SCV set combination (average utility and percentage target fulfillment close to 1), this may be curious for the other GPs. However, this can again be justified by the random selection of SCV sets within the first 200 GPs in case of equal utilities. This leads to a different set of available SCV sets in the two cases and consequently, different real network effect models.

8.3.4. Results for Suburban Macro Cells

The next cells under investigation are suburban cells, covered by CLASS_4. In contrast to CLASS_1 - CLASS_3, the differences between the five cases are not as huge. For the first 200 GPs, the performance can be denominated as equal. After GP 200, there is clearly a little plus for the ASM and CSM case while both of them

perform similarly. It seems that none of the untested SCV set combinations could provide a better performance than one of the already tested sets such that no such peak as for the previously investigated classes can be detected.

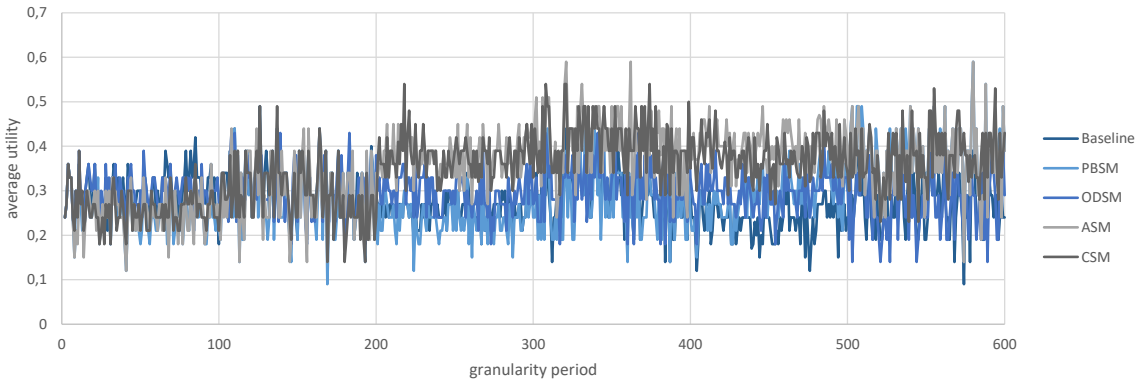


Figure 8.25.: Average utilities for manual context class CLASS_4

In summary, the baseline again remains fairly constant over the whole simulation time. Additionally, PBSM and ODSM are not able to satisfactorily improve the results of a uniformly deployed default SCV set. Only ASM and CSM achieve satisfactory results while both are still not able to fulfill CLASS_4's CQI target, resulting in an average utility which is under 0.5 most of the time due to the relatively high CQI weights.

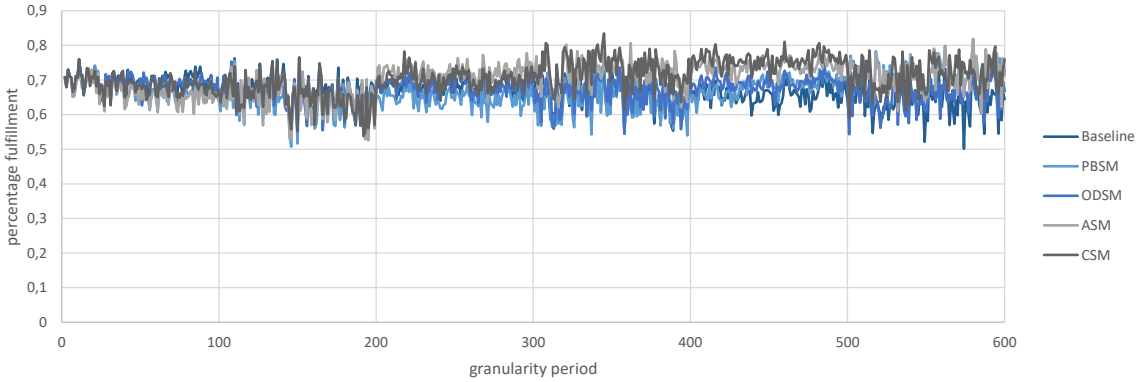


Figure 8.26.: Percentage target fulfillment for manual context class CLASS_4

8.3.5. Results for City Center Micro Cells

Finally, the results for the seven micro cells are analyzed. Note that these cells are deactivated in the low traffic scenario, leading to zero values for one half of

the simulated GPs. First of all, it can be seen, that the average utilities as well as the percentage target fulfillment values are extremely high for all cases and at the same time, the differences between them are fairly low. The most significant difference can be seen during the first 100 GPs: The various approaches need a different number of GPs until they converge at a certain average utility, around 0.95. The reason, why this value can not be exceeded, is, that there are always two cells in all cases which can not fulfill the CQI target. With a weight of 0.2, this target is not of great importance and hence, leads to only this small gap to the highest possible average utility. Since only at a few GPs, the PiPo and CL targets can not be fulfilled as well, the graph shows some troughs at later points in time.

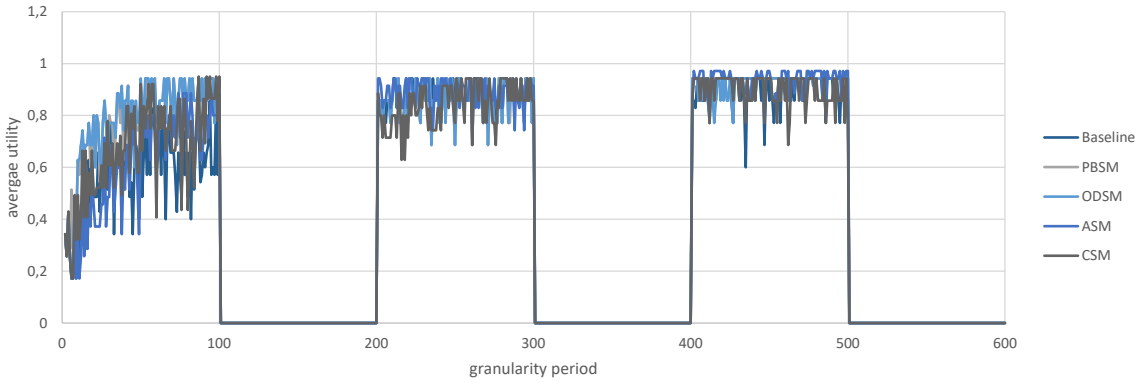


Figure 8.27.: Average utilities for manual context class CLASS_5

Furthermore, CLASS_5 is the only context class where CSM does not perform best. During the first 100 GPs, ODSM does a better job than any of the other approaches. In the further periods, ASM achieves the best results, followed by ODSM and PBSM.

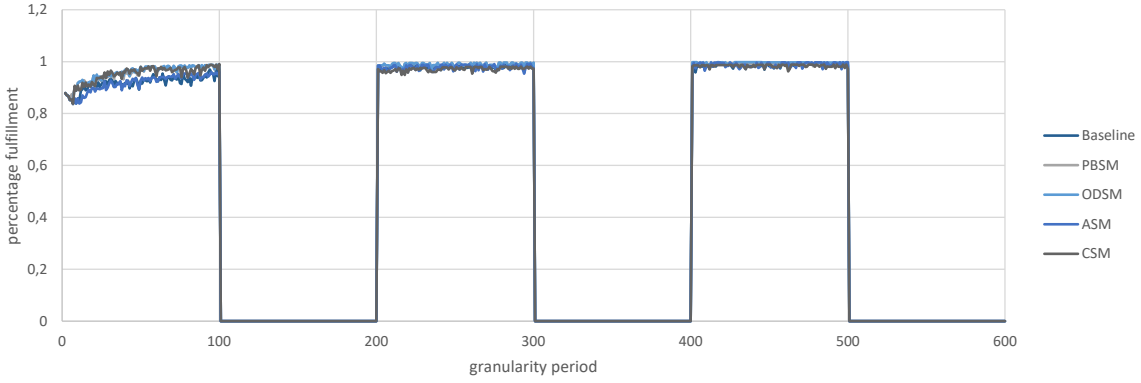


Figure 8.28.: Percentage target fulfillment for manual context class CLASS_5

8.3.6. Discussion

In this section, the value of SON management, considering the results presented in the previous sections, shall be discussed and summarized.

Constant Results of Baseline
In all cases, the results of applying default SCV sets remain constant over the whole simulation time. While this is more or less expected behavior, it is important to have this confirmed. Since at the same time, the baseline is far away from fulfilling KPI targets in a satisfactory way, there is still a lot of potential for improvement for the four presented SON management approaches. This also shows that it is not sufficient to run a uniform, network-wide configuration without considering the continuously changing environment. A SON management system is needed that permanently reacts to these changes and adapts the network accordingly.

Effect of Random SCV Set Selection in ODSM
During the first 200 GPs, three cases use the ODSM approach: ODSM, ASM and CSM. One may have expected that within this time frame, these approaches provide the same or at least similar results. However, except for CLASS_3, utilities as well as percentage target fulfillment are often fairly different. The reason for that has already been explained: There is a random component within ODSM that selects arbitrary SCV sets in case of equal (highest) utilities. After seeing the results, one may see this as a critical point. Nevertheless, this is an important factor especially in the ASM and CSM case. Without this random selection, i.e., when assuming a selection based on a distance calculation, the objective manager would probably choose the same SCV set combination every time, since no real network measurements are taken into account that could influence the reasoning process. As a consequence, there would be only a few SCV sets where measurements are available in the real network effect model, leading always to similar results in the ASM case and making the prediction of untested SCV sets in the CSM case nearly impossible due to the lack of data. Roughly spoken, it can be said that the promising results in terms of ASM and CSM would not be possible without this random selection.

Relationship between Percentage Target Fulfillment and Average Utility
There is a big difference between optimizing a network towards objective fulfillment or trying to improve the KPI values itself. This can be seen when comparing the curves of the average utility and percentage target fulfillment graphs. It becomes obvious at first sight that the curves often look fairly different. Also, the percentage target fulfillment can be close to 100% while at the same time, having a really bad utility. SON management, as it is described in this thesis, aims at optimizing operator objectives since this is what operators are paid for. Accordingly, this confirms the importance of Objective 4 of this thesis: The SON management needs to be designed in a way that it is trustworthy for an MNO. Most of the

work which can be found related to SON management, aims at optimizing the KPI values without considering operator objectives, which is clearly against the actual goals of MNOs.

Influence of Machine Learning
For three out of the five manual context classes, CSM could identify an untested SCV set that significantly performs better than those tested before. Furthermore, CSM never deployed an untested SCV set combination that decreased one of the two metrics significantly. This in turn means that the effect prediction works quite well since CSM was either using the real network effect model or only untested combined SCV sets that provide at least an equally positive effect as each of the tested SCV set combinations.

Difference between Cases over Simulation Time
Except for CLASS_5, the gap between the worst and best performing case became larger with increasing simulation time. During the first 200 GPs, all cases often yielded the same results. After that, the baseline remained constant while PBSM and ODSM usually needed more time to settle down. ASM and CSM often received a significant boost after using the real network effect model for the first time. Since the adaptation of the real network effect model is a continuous process, this lasted until the final GP for some of the context classes, e.g., CLASS_1 and CLASS_2 in the high traffic scenario. The peaks in CSM often intensify this observation and further increase the gap. Due to this finding, it would be interesting to see how the behavior of the different SON management would change with an increasing simulation time. Will the results converge, if they do at all? Will the learned effect model be consulted more often? However, a more powerful computer is needed for longer simulations, since the SEASON II simulator requires a lot of resources and hence, this is subject to future work.

The Benefit of SON Management
The most important question is probably: After seeing the results of the simulations, how well is SON management performing in fulfilling the operator objectives? It can be clearly said that SON management has a considerable effect on the fulfillment of operator objectives. Moreover, the effect gets more significant with later stages of development. That is, using an initial effect model instead of default SCV sets brings the first little boost. Upgrading this to a combined effect model and the usage of a real network effect model results in a further improvement, and using machine learning even outperforms these results. When looking at the concrete values, i.e., the average utilities of the particular context classes, there is still a lot of potential for further improvement while an optimal target fulfillment, i.e., all cells have a utility of 1, can probably never be achieved. Furthermore, note that the KPI targets in these simulations are based on experience and could look rather different in practice.

9 Conclusion and Outlook

In the previous chapters several approaches have been presented that deal with the development of a SON management system. These approaches have been evaluated against each other and compared to the current situation in mobile networks in order to show their advantages and the benefit of SON management. In this chapter, this thesis and the results are shortly summarized and the achievements are related to the objectives defined in Chapter 1. Finally, a few ideas are presented how SON management can be used in the future and how to further extend the most sophisticated CSM approach.

9.1. Summary

In this thesis, four approaches on SON management, namely PBSM, ODSM, ASM and CSM have been introduced. These approaches have been consecutively developed, whereby each management approach aims at improving the previous one and at overcoming its predecessors' shortcomings. Thereby, each of these approaches can solve one or more of this thesis' objectives, defined in Section 1.2:

Objective 1 *Automate the process of finding optimal SCV sets and close the manual gap between operator objectives and SCV sets.*

Objective 2 *Automatically generate realistic and complete input models which are continuously updated according to the current network state such that they optimally support the SON management system in finding the best possible network configuration.*

Objective 3 *Reliably estimate the performance of untested SCV sets dependent on automatically derived context information which has been reduced to a manageable level.*

Objective 4 *Develop a fully automated SON management system which allows MNOs to comprehend, restrict and influence automated actions at any time.*

In general, all four SON management approaches aim at the same goal: Relieve the human MNO from manual tasks, thereby reducing OPEX and facilitating operators to remain profitable. However, the different solutions each represent one stage on the way to develop a sophisticated SON management system, each of

them having their own advantages and drawbacks. Thus, in the following, the SON management approaches are shortly summarized and it is shown how far the particular approaches contribute to solving these objectives.

9.1.1. Policy-based SON Management

In Chapter 4, an approach has been presented that allows to overcome the manual gap between network operator-defined technical objectives for the operation and management of a SON-enabled mobile radio network, and the configuration of SON functions that aim at optimizing NCPs in order to fulfill these technical objectives. The manual gap comprises, first of all, the missing availability of an automated transformation between technical objectives and SCV sets (automation gap), furthermore, the lacking capabilities to dynamically adapt the SCPs of a SON function to a changing operational and network context (dynamics gap) and finally, the lack of operator and SON function manufacturer knowledge in a form that can be processed automatically (knowledge gap).

Several models have been introduced that allow the standardized description of the information required to operate a SON-enabled mobile radio network. The objective model, provided by the network operator, describes the KPI targets, priorities associated with the KPI targets, and the conditions under which these KPI targets and priorities shall apply. The effect model, provided by the SON function manufacturer, describes which KPI targets the SON function can pursue and the according SCV sets to configure the SON function. The context model provides a description of the properties of operational and network status context information. PBSM is the first SON management approach that uses these types of automatically processable models and thereby closes the knowledge gap.

An objective manager has been introduced that creates an SCV set policy using the information provided in the three models. This SCV set policy maps specific context regions to an SCV set per SON function configuring the SON function to pursue the technical objectives. In this way, the objective manager closes the automation gap by performing an automated reasoning process. Furthermore, a policy system has been proposed which evaluates the SCV set policy according to a specific context and configures the SON accordingly. This allows to dynamically react on changing network and operational conditions, thus closing the dynamics gap.

The presented approach represents an important step towards automated network operation, by shifting the responsibility of the human network operator from the repetitive configuration and operation of individual SON functions towards the definition of technical objectives according to which the mobile network shall operate. The approach thereby represents a means for an objective-driven control of a SON-enabled mobile radio network. By overcoming the manual gap, PBSM provides a first solution for Objective 1 and at the same time, an

approach that does not violate Objective 4. The usage of a design-time approach which generates an SCV set policy first before deploying new configurations in the network, enables the MNO to always interrupt an automated reconfiguration if necessary.

9.1.2. Objective-driven SON Management

In Chapter 5, an ODSM approach has been presented as an extension of the initially introduced PBSM approach which dynamically configures a SON-enabled mobile network according to context-specific operator objectives and a combined effect model. This is achieved by an objective manager component which combines several specific manufacturer-provided effect models into a combined effect model, thereby identifying SCV set conflicts, and evaluating this combination against a context-dependent objective model. In contrast to the former approach, the presented concept is put on a mathematical foundation for both the description of the promised system performance for a SCV set in the effect model as well as the definition of the desired system performance by the operator. This enables more expressive effect models and objectives which allows the definition of thresholds for KPIs in addition to minimization or maximization targets. Furthermore, it is possible to determine the best SCV set over a set of weighted objectives instead of ranked objectives which allows to find better trade-offs between their satisfaction.

Furthermore, the manufacturer-provided effect models and as a consequence, the combined effect model in this approach on SON management are context-dependent. This extension accounts for the need to express that an SCV set for a SON function can produce diverse system behavior in different operational contexts.

Since ODSM can be seen as an extension to PBSM, this management approach also solves Objective 1 while using a different type of reasoning process in the objective manager due to different and more complex input models. Having a more expressive objective model, but especially a more precise effect model, marks an important step towards solving Objective 2, the generation of realistic input models. Considering combinations of SCV sets is important since in reality, usually more than one SON function is deployed in the network. However, the combined effect model in this approach is still based on simulations done by the SON function manufacturer such that Objective 2 can not be seen as solved by adopting ODSM. Similar to PBSM, ODSM is also designed in a way that it provides possibilities for an MNO to intervene and influence automated actions at any time, thereby gaining trust in the system and partially solving Objective 4. These possibilities include the generation of an SCV set policy by which the MNO always has an overview of SON management results and the different types of combining manufacturer effect models depending on his or her level of trust.

9.1.3. Adaptive SON Management

In Chapter 6, an ASM approach has been presented that complements static SON manufacturer-provided effect models with real network KPI values that are based on measurements from the dynamic network environment. By acquiring and analyzing measured network data, and mapping it with the currently active SCV sets, a real network effect model is created. Using this real network effect model, the SCV set policy and thereby the SON functions' SCV sets, are updated such that the SON functions contribute better towards achieving KPI targets. Thereby, SON management is no longer dependent on externally provided and simulation based effect models, but is turned into a closed control loop which permanently improves itself.

In the ASM approach, another major shortcoming of the PBSM and ODSM approaches has been overcome: The reduction of the huge context state space to a manageable number of context classes. Cells where a similar behavior is assumed, are combined to a context class such that in the end, an MNO only has to define objectives for the small number of classes instead of a huge number of context states. In the end, this also affects the SCV set policy in terms of clearness and manageability.

In Chapter 8, it has been shown that ASM results in the biggest increase of all SON management approaches when comparing them to the respective previous stage of development. This is mainly due to the fact that ASM takes real measured KPI values into account and performs a much more complex yet more realistic reasoning process. However, one of the disadvantages of ASM is the fact that it still can only react on changes but not act in a predictive manner. That is, it can only use SCV sets where the effect is already known and contained in the real network effect model.

It is obvious that also ASM solves Objective 1 by automatically configuring the network according to operator defined objectives. Furthermore, ASM has a big stake in solving Objective 2, since it is the first approach to introduce a real network effect model and a manual context classes model. The real network effect model is a product of permanently gathering network data and thereby, perfectly reflects the behavior of the system where SON management is actually applied. The manual context classes model is created by inspection of the real network and hence, only contains network attribute combinations in the shape of context classes that really exist in the network. With the more complex ASM approach, further possibilities of influencing the functioning of SON management come along. The MNO can decide about the trustworthiness of the real network effect model and the point in time when it shall be applied.

9.1.4. Cognitive SON Management

In Chapter 7, a CSM approach has been presented that extends the existing ASM approach with machine learning capabilities. More precisely, supervised learning techniques are used to predict the effects of up-to-now untested SCV sets, resulting in a complete effect model, and unsupervised learning techniques are used to automatically cluster the set of cells into classes of cells with similar behavior according to the achieved KPI values. Additionally, the usage of different types of cell classifications requires the objective manager to calculate optimal SCV sets for each cell individually, resulting in a more precise SCV set policy.

In Chapter 8, four different algorithms in the field of supervised learning have been extensively evaluated with various different regression models in order to find the best performing model for predicting effects of untested SCV sets. These models were trained on data from all possible number of cell clusters such that in the end, the error rates in terms of RMSE could be compared for all models and every number of clusters. ANN thereby delivered the best results with a promising error rate for a higher number of clusters. The best performing ANN model has been implemented in the overall CSM approach and the CSM results were compared to previous stages of development. In most evaluated cases, CSM was able to identify an untested SCV set combination that could significantly increase the cells' utilities and the average KPI values. At the same time, the values never decreased compared to the ASM approach which proves that the used ANN model worked well in terms of KPI effect predictions.

CSM can be seen as an ASM approach equipped with machine learning capabilities and hence, it also solves Objective 1. While it is nearly impossible to make perfectly accurate predictions, the evidence was produced in the evaluation part, that the learned effect model delivers appropriate results. Thus, the learned effect model also fulfills Objective 2, since it sufficiently reflects the real measured effects. The same applies for Objective 3, for the same reason. It has been shown that machine learning can reliably estimate the effect of untested SCV sets and, as a consequence, can help to improve the fulfillment of operator objectives. Finally, the usage of newly generated, learned models is also in accordance with Objective 4. The operator can decide about the point in time at which a learned effect model shall be initially used. Furthermore, after each GP, the RMSE is reported to the MNO such that a current image of the underlying learning process is always reflected.

9.2. Future Work

Even though CSM is the most sophisticated approach in this thesis, there are still some points that can be improved and where CSM can be extended. Some of these points that go beyond the scope of this thesis are presented in this section.

It starts with suggestions how the results of the learning process can be further improved. Afterwards, possible extensions that could be combined with the here presented SON management approaches, are described in more detail. Finally, an idea is presented how SON management could be integrated with network slicing.

9.2.1. Machine Learning

Hämäläinen et al. already mentioned the concept of Cognitive Radio Network (CRN) in [HSS11] as a necessary future extension of SON:

> "[Future network scenarios] are highly dynamic and induce frequent changes of the operational context. Therefore, the SON concept is regarded as not fully meeting the challenges imposed by these new scenarios. In consequence, SON has to evolve [...]. [These extensions] lead to the concept of CRN."

A CRN is considered to be capable of reasoning and remembering, capabilities already implemented to a high degree by the aforementioned CSM framework. However, in order to achieve a CRN system as described by [HSS11], the presented CSM system needs to be extended. Even though the achieved results in terms of RMSE are promising, there is still room for further improvements: A logical next step would be the test of other learning algorithms and evaluating their results against those illustrated in Chapter 8 of this thesis. Additionally, it might be worth looking into more domains to base the process of learning the effects of untested SCV sets on. Values such as antenna height, distance and amount of neighboring cells and others might enable a more accurate effect prediction. Furthermore, the regression models should be tested using data from different (more heterogeneous) scenarios, data generated by different simulation engines or even data collected from real networks.

In the course of mobile networks getting more and more complex with an increasing amount of base stations due to the increasing amount of mobile device users, the *big data* issue is of growing importance. While the focus of this thesis is on showing that SON management is theoretically as well as practically working at all, the runtime of presented approaches and of the learning process in particular has not been investigated. However, in a real network with thousands of cells and the multiple of data that has been produced in this thesis' simulations, this point can not be neglected, especially when also taking additional parameters into account.

9.2.2. Extensions to SON Management

Each of the SON management approaches that have been presented in this thesis can operate as a standalone component what has been proven in the evaluation part. However, there are some meaningful extensions, whose implementation and combination with SON management could be worth being evaluated. First of all, [Fre16] presents a more complex objective model: Instead of a binary KPI target fulfillment, the range of a KPI target is separated into an optimal, an acceptable and an unacceptable range, allowing for a more complex evaluation of the objectives. This approach can be easily combined with the ODSM, ASM or CSM approach since the differences in the objective models only influence the utility calculations when selecting an optimal SCV set combination.

In Chapter 5, a method has been presented to perform a design-time SON coordination when combining the manufacturer-provided effect models of different SON functions. Thereby, the respective KPI effects are compared with each other and SCV sets whose combination leads to a contrary effect on one of the KPIs, are not considered for further calculations. In Section 5.4, it has already been said, that this pre-coordination is not what is actually understood by SON coordination in the literature. SON coordination usually works as an additional component during runtime, observing which SCPs are touched by the various SON functions and intervening in case a conflict occurs. While the pre-coordination in ODSM, ASM and CSM is still useful for detecting possible conflicts beforehand, it can be simply extended with a runtime SON coordination component: Before deploying a new SCV set combination on a cell, SON coordination can be interconnected and consulted, showing whether a projected SON function reconfiguration causes a conflict or not.

9.2.3. SON Management for Network Slicing

Network slicing is currently one of the hot topics in the field of mobile networks and hence, ideas are presented how SON management and network slicing can be combined. Furthermore, problems are depicted occurring when combining these two domains. First of all, a definition of network slicing shall be given according to Fendt et al. [Fen+18a]:

> "Network slicing is one of the key features of 5G mobile networks to cope with the diverging network requirements introduced by new use cases, like the IoT, autonomous driving and the Industry of the Future. Network slices are isolated, virtualized, end-to-end networks optimized for specific use cases. But still they share a common physical network infrastructure."

That is, several use cases claim resources of the shared physical network. For each of these use cases, a virtual network is created with its own goals and objectives.

Here, SON management comes into play: It may seem the easiest way to deploy a SON management system for each of the network slices individually. However, note that the same physical network is at the basis of all the different network slices and hence, the objectives of the particular use cases are conflicting with the utmost probability, requiring for a component that manages and coordinates the sets of network slice-specific objectives and combines them into a universal set for the underlying physical network. Several approaches are thinkable for this combination: For instance, objectives could be joined for each KPI to the one which has the hardest target to achieve. In doing so, SON management is prevented from stopping to optimize a certain KPI before having fulfilled the hardest one. However, this approach does not work for the objective weights. An approach to solve this problem could be to build the arithmetic mean of the weights, having the disadvantage that an increasing number of slices would probably cause an equal distribution of the weights, making them needless in the end such that a more sophisticates solution must be found here.

Part IV.

Annex

Bibliography

[3GP12] 3GPP. *Telecommunication management; Self-Organizing Networks (SON) Policy Network Resource Model (NRM) Integration Reference Point (IRP); Requirements (Release 11). Technical specification TS 32.521 V11.1.0*. 3rd Generation Partnership Project. 2012.

[3GP13] 3GPP. *Telecommunication management; Self-Organizing Networks (SON) Policy Network Resource Model (NRM) Integration Reference Point (IRP); Information Service (IS) (Release 11). Technical specification TS 32.522 V11.7.0*. 3rd Generation Partnership Project. 2013.

[3GP18a] 3GPP. *Technical Specification Group Services and System Aspects; Network architecture (Release 15). Technical specification TS 36.213 V15.3.0*. 3rd Generation Partnership Project. 2018.

[3GP18b] 3GPP. *Telecommunication management; Self-Organizing Networks (SON); Concepts and requirements (Release 15). Technical specification TS 32.500 V15.0.0*. 3rd Generation Partnership Project. 2018.

[3GP18c] 3GPP. *Telecommunication management; Self-Organizing Networks (SON); Self-healing concepts and requirements (Release 15). Technical specification TS 32.541 V15.0.0*. 3rd Generation Partnership Project. 2018.

[Aga+09] Nedaa Agami et al. "A neural network based dynamic forecasting model for Trend Impact Analysis". In: *Technological Forecasting and Social Change Volume 76* (Sept. 2009), pp. 952–962.

[Ame12] 4G Americas. "Developing and Integrating a High Performance HET-NET". In: *4G Americas, Whitepaper* (Oct. 2012).

[Ami+15] Mehdi Amirijoo et al. *Demonstration Scenarios (updated version)*. Deliverable D3.4. SEMAFOUR Project, May 2015.

[Ari+10] G. Aristomenopoulos et al. "Autonomic mobility and resource management over an integrated wireless environment — A GANA oriented architecture". In: *2010 IEEE Globecom Workshops* (Dec. 2010), pp. 545–550.

[AV07] David Arthur and Sergei Vassilvitskii. "K-Means++: The Advantages of Careful Seeding". In: *2007 18th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)* (Jan. 2007), pp. 1027–1035.

[AY14] Khalid A. A. Abakar and Chongwen Yu. "Performance of SVM based on PUK kernel in comparison to SVM based on RBF kernel in prediction of yarn tenacity". In: *Indian Journal of Fibre and Textile Research 39* (Mar. 2014), pp. 55–59.

[Ban+04] Arosha K. Bandara et al. "A Goal-based Approach to Policy Refinement". In: *Fifth IEEE International Workshop on Policies for Distributed Systems and Networks (POLICY 2004)* (June 2004), pp. 229–239.

[Ban+11] Tobias Bandh et al. "Policy-based coordination and management of SON functions". In: *12th IFIP/IEEE International Symposium on Integrated Network Management (IM 2011) and Workshops* (May 2011), pp. 827–840.

[Ban13] Tobias Bandh. *Coordination of autonomic function execution in Self-Organizing Networks*. PhD Thesis. Technische Universität München, Mar. 2013.

[Bar+13] E. Barrett et al. *Specification of cognitive network management framework and functions*. Deliverable D3.1. COMMUNE Project, Jan. 2013.

[BD09] Ronen I. Brafman and Carmel Domshlak. "Preference Handling – An Introductory Tutorial". In: *AI Magazine, vol. 30, no. 1* (2009), pp. 58–86.

[Ben09] Yoshua Bengio. "Learning Deep Architectures for AI". In: *Foundations and Trends in Machine Learning* (2009), pp. 1–127.

[BHH05] George E.P. Box, J. Stuart Hunter, and William G. Hunter. *Statistics for Experimenters. Design, Innovation, and Discovery, 2nd Edition*. Hoboken: NJ: John Wiley & Sons, Inc., 2005.

[Bis06] Christopher M. Bishop. *Pattern Recognition and Machine Learning. with Applications in R*. Ed. by Michael Jordan, Jon Kleinberg, and Bernhard Schölkopf. New York: NY: Springer New York, 2006.

[BSR11] Tobias Bandh, Henning Sanneck, and Raphael Romeikat. "An Experimental System for SON Function Coordination". In: *2011 IEEE 73rd Vehicular Technology Conference (VTC Spring)* (May 2011), pp. 1–2.

[Buc+18] Christian Buchner et al. *SEASON II USER-MANUAL. SON2018a*. Ed. by Sina Khatibi. NOMOR Research GmbH. 2018.

[CAA13] Richard Combes, Zwi Altman, and Eitan Altman. "Coordination of autonomic functionalities in communications networks". In: *2013 11th International Symposium and Workshops on Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks (WiOpt)* (May 2013), pp. 364–371.

[Cam+15] Luis Campoy et al. *Integrated SON Management Implementation Recommendations*. Deliverable D5.4. SEMAFOUR Project, Aug. 2015.

[CWP17] Cisco and/or its affiliates. "Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2016-2021". In: *Cisco White Paper* (Feb. 2017).

[DJD17a] Tony Daher, Sana Ben Jemaa, and Laurent Decreusefond. "Cognitive management of self - Organized radio networks based on multi armed bandit". In: *2017 IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)* (Oct. 2017), pp. 1–5.

[DJD17b] Tony Daher, Sana Ben Jemaa, and Laurent Decreusefond. "Q-Learning for Policy Based SON Management in wireless Access Networks". In: *2017 IFIP/IEEE Symposium on Integrated Network and Service Management (IM)* (May 2017), pp. 1091–1096.

[DJD18a] Tony Daher, Sana Ben Jemaa, and Laurent Decreusefond. "Linear UCB for Online SON Management". In: *2018 IEEE 87th Vehicular Technology Conference (VTC Spring)* (June 2018), pp. 1–5.

[DJD18b] Tony Daher, Sana Ben Jemaa, and Laurent Decreusefond. "Softwarized and distributed learning for SON management systems". In: *NOMS 2018 - 2018 IEEE/IFIP Network Operations and Management Symposium* (Apr. 2018), pp. 1–7.

[DK13] Khoa Truong Dinh and Slawomir Kuklinski. "Joint implementation of several LTE-SON functions". In: *2013 IEEE Globecom Workshops (GC Wkshps)* (Dec. 2013), pp. 953–957.

[DRO18] Red Hat Inc. *Drools*. visited on 05/11/2018. 2018. URL: `https://www.drools.org/`.

[Eck07] Taufiq Rochaeli ; Claudia Eckert. "Expertise Knowledge-Based Policy Refinement Process". In: *Eighth IEEE International Workshop on Policies for Distributed Systems and Networks (POLICY'07)* (June 2007), pp. 61–65.

[Fen+18a] Andrea Fendt et al. "A Formal Optimization Model for 5G Mobile Network Slice Resource Allocation". In: *The 9th IEEE Annual Information Technology, Electronics & Mobile Communication Conference (IEMCON)* (Nov. 2018), pp. 101–106.

[Fen+18b] Andrea Fendt et al. "A Network Slice Resource Allocation and Optimization Model for End-to-End Mobile Networks". In: *IEEE 1st 5G World Forum (5GWF'18)* (July 2018).

[Fen+18c] Andrea Fendt et al. "A Network Slice Resource Allocation Process in 5G Mobile Networks". In: *2018 12th International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS)* (July 2018), pp. 695–704.

[FLS14a] Christoph Frenzel, Simon Lohmüller, and Lars Christoph Schmelz. "Dynamic, context-specific SON management driven by operator objectives". In: *2014 IEEE Network Operations and Management Symposium (NOMS)* (May 2014), pp. 1–8.

[FLS14b] Christoph Frenzel, Simon Lohmüller, and Lars Christoph Schmelz. "SON Management based on Weighted Objectives and Combined SON Function Models". In: *2014 11th International Symposium on Wireless Communications Systems (ISWCS)* (Aug. 2014), pp. 149–153.

[Fre+15] Christoph Frenzel et al. "Objective-driven coordination in self-organizing networks". In: *2015 IEEE 26th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)* (Sept. 2015), pp. 1453–1458.

[Fre+16] Christoph Frenzel et al. "Demonstrator for utility-based SON management". In: *2016 IEEE 27th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)* (Sept. 2016), pp. 1–2.

[Fre16] Christoph Frenzel. *Objective-Driven Operations of Self-Organizing Networks*. PhD Thesis. Universität Augsburg, Mar. 2016.

[FSB13] Christoph Frenzel, Henning Sanneck, and Bernhard Bauer. "A Fuzzy, Utility-Based Approach for Proactive Policy-Based Management". In: *International Workshop on Rules and Rule Markup Languages for the Semantic Web (RuleML 2013)* (July 2013), pp. 84–98.

[Gal+12] A. Galani et al. "A policy based framework for governing Future networks". In: *2012 IEEE Globecom Workshops* (Dec. 2012), pp. 802–806.

[Gaz+18] Juraj Gazda et al. "Unsupervised Learning Algorithm for Intelligent Coverage Planning and Performance Optimization of Multitier Heterogeneous Network". In: *IEEE Access, vol. 6* (June 2018), pp. 39807–39819.

[Göt+15] Dario Götz et al. *Integrated SON Management - Policy Transformation and Operational SON Coordination (final results)*. Deliverable D5.3. SEMAFOUR Project, Feb. 2015.

[Gue+06] Antonia Guerrero et al. "Expertise Knowledge-Based Policy Refinement Process". In: *17th IFIP/IEEE International Workshop on Distributed Systems: Operations and Management, DSOM 2006* (Oct. 2006), pp. 61–65.

[Hah+14] Sören Hahn et al. "SON Management simulator implementation and findings". In: *2014 IEEE Network Operations and Management Symposium (NOMS)* (May 2014), pp. 1–15.

[Hah+15a] Sören Hahn et al. "Classification of Cells Based on Mobile Network Context Information for the Management of SON Systems". In: *2015 IEEE 81st Vehicular Technology Conference (VTC Spring)* (May 2015), pp. 1–5.

[Hah+15b] Sören Hahn et al. *Final report on a unified self-management system for heterogeneous radio access networks*. Deliverable D6.6. SEMAFOUR Project, Aug. 2015.

[Han+11] Xiaogang Han et al. "An Optimized K-Nearest Neighbor Algorithm for Large Scale Hierarchical Text Classification". In: *Joint ECML/P-KDD PASCAL Workshop on Large-Scale Hierarchical Classification* (2011), pp. 2–12.

[Hea08] Jeff Heaton. *Introduction to Neural Networks for Java*. Heaton Research, 2008.

[HK14] Sören Hahn and Thomas Kürner. "Managing and altering mobile radio networks by using SON function performance models". In: *2014 11th International Symposium on Wireless Communications Systems (ISWCS)* (Aug. 2014), pp. 214–218.

[HSK18] Sören Hahn, Michael Schweins, and Thomas Kürner. "Impact of SON function combinations on the KPI behaviour in realistic mobile network scenarios". In: *2018 IEEE Wireless Communications and Networking Conference Workshops (WCNCW)* (Apr. 2018), pp. 1–6.

[HSS11] Seppo Hämäläinen, Henning Sanneck, and Cinzia Sartori. *LTE Self-Organising Networks (SON). Network Management Automation for Operational Efficiency*. Ed. by Seppo Hämäläinen, Henning Sanneck, and Cinzia Sartori. Chichester: UK: John Wiley & Sons, Ltd., 2011.

[Iac+14a] Ovidiu Iacoboaiea et al. "Coordinating SON instances: Reinforcement learning with distributed value function". In: *2014 IEEE 25th Annual International Symposium on Personal, Indoor, and Mobile Radio Communication (PIMRC)* (Sept. 2014), pp. 1642–1646.

[Iac+14b] Ovidiu Iacoboaiea et al. "Low complexity SON coordination using reinforcement learning". In: *2014 IEEE Global Communications Conference* (Dec. 2014), pp. 4406–4411.

[Iac+14c] Ovidiu Iacoboaiea et al. "SON Coordination for parameter conflict resolution: A reinforcement learning framework". In: *2014 IEEE Wireless Communications and Networking Conference Workshops (WCNCW)* (Apr. 2014), pp. 196–201.

[Iac+16] Ovidiu-Constantin Iacoboaiea et al. "SON Coordination in Heterogeneous Networks: A Reinforcement Learning Framework". In: *IEEE Transactions on Wireless Communications, vol. 15* (Sept. 2016), pp. 5835–5847.

[IM12a] Muhammad Naseer ul Islam and Andreas Mitschele-Thiel. "Cooperative Fuzzy Q-Learning for self-organized coverage and capacity optimization". In: *2012 IEEE 23rd International Symposium on Personal, Indoor and Mobile Radio Communications - (PIMRC)* (Sept. 2012), pp. 1406–1411.

[IM12b] Muhammad Naseer ul Islam and Andreas Mitschele-Thiel. "Reinforcement learning strategies for self-organized coverage and capacity optimization". In: *2012 IEEE Wireless Communications and Networking Conference (WCNC)* (Apr. 2012), pp. 2818–2823.

[Jam+13] Gareth James et al. *An Introduction to Statistical Learning. with Applications in R*. Ed. by G. Casella, S. Fienberg, and I. Olkin. New York: NY: Springer New York, 2013.

[Jem+13] Sana Ben Jemaa et al. *Integrated SON Management Requirements and Basic Concepts*. Deliverable D5.1. SEMAFOUR Project, Dec. 2013.

[JSON] Douglas Crockford. *Introducing JSON*. visited on 03/01/2019. URL: https://www.json.org/.

[KN10] A. Kousaridas and G. Nguengang. *Final Report on Self-Organisation and its Implications in Wireless Access Networks*. Deliverable D2.3. SelfNET Project, Apr. 2010.

[KNN10] Saravanan Thirumuruganathan. *A Detailed Introduction to K-Nearest Neighbor (KNN) Algorithm*. visited on 31/10/2018. 2010. URL: https://saravananthirumuruganathan.wordpress.com/2010/05/17/a-detailed-introduction-to-k-nearest-neighbor-knn-algorithm/.

[KR08] Leonard Kaufman and Peter J. Rousseeuw. *Finding Groups in Data. An Introduction to Cluster Analysis*. Hoboken: NJ: John Wiley & Sons, Inc., 2008.

[Kür+10] Thomas Kürner et al. *Final Report on Self-Organisation and its Implications in Wireless Access Networks*. Deliverable D5.9. SOCRATES Project, Jan. 2010.

[Lee15] Jeff Leek. *The Elements of Data Analytic Style*. Lean Publishing, 2015.

[LH00] Harri Lappalainen and Antti Honkela. "Bayesian Nonlinear Independent Component Analysis by Multi-Layer Perceptrons". In: *Girolami M. (eds) Advances in Independent Component Analysis. Perspectives in Neural Computing* (2000), pp. 93–121.

[Loh+15] Simon Lohmüller et al. "Policy-Based SON Management Demonstrator". In: *2015 IEEE 81st Vehicular Technology Conference (VTC Spring)* (May 2015), pp. 1–2.

[Loh+18] Simon Lohmüller et al. "SON Function Performance Prediction in a Cognitive SON Management System". In: *2018 IEEE Wireless Communications and Networking Conference Workshops (WCNCW)* (Apr. 2018), pp. 13–18.

[LSH16] Simon Lohmüller, Lars Christoph Schmelz, and Sören Hahn. "Adaptive SON management using KPI measurements". In: *NOMS 2016 - 2016 IEEE/IFIP Network Operations and Management Symposium* (Apr. 2016), pp. 625–631.

[Man+16] Christian Mannweiler et al. "Cross-domain 5G Network Management for Seamless Industrial Communications". In: *NOMS 2016 - 2016 IEEE/IFIP Network Operations and Management Symposium* (Apr. 2016), pp. 868–872.

[Mar+10] Nicola Marchetti et al. "Self-organizing networks: State-of-the-art, challenges and perspectives". In: *8th International Conference on Communications, COMM 2010* (June 2010), pp. 503–508.

[MD93] Christopher Z. Mooney and Robert D. Duval. *Bootstrapping. A Nonparametric Approach to Statistical Inference*. Ed. by Susan McElroy. CA: Newbury Park: Sage Publications, Inc., 1993.

[Mei+10] Juha Meinilä et al. *D5.3: WINNER+ Final Channel Models*. Deliverable D5.3. CELTIC Project, June 2010.

[Mit97] Tom M. Mitchell. *Machine Learning*. Ed. by C.L. Liu and Allen B. Tucker. NY: New York: McGraw-Hill Companies, Inc., 1997.

[MM13a] Stephen S. Mwanje and Andreas Mitschele-Thiel. "A Q-Learning strategy for LTE mobility Load Balancing". In: *2013 IEEE 24th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)* (Sept. 2013), pp. 2154–2158.

[MM13b] Stephen S. Mwanje and Andreas Mitschele-Thiel. "Minimizing Handover Performance Degradation Due to LTE Self Organized Mobility Load Balancing". In: *2013 IEEE 77th Vehicular Technology Conference (VTC Spring)* (June 2013), pp. 1–5.

[MS93] Jonathan D. Moffett and Morris S. Sloman. "Policy Hierarchies for Distributed Systems Management". In: *IEEE Journal on Selected Areas in Communications (JSAC) Volume 11* (Dec. 1993), pp. 1404–1414.

[MSM16] Stephen S. Mwanje, Lars Christoph Schmelz, and Andreas Mitschele-Thiel. "Cognitive Cellular Networks: A Q-Learning Framework for Self-Organizing Networks". In: *IEEE Transactions on Network and Service Management, vol. 13* (Mar. 2016), pp. 85–98.

[MSM18] Stephen S. Mwanje, Henning Sanneck, and Andreas Mitschele-Thiel. "Synchronized Cooperative Learning for Coordinating Cognitive Network Management Functions". In: *IEEE Transactions on Cognitive Communications and Networking, vol. 4* (Feb. 2018), pp. 244–256.

[Mur12] Kevin P. Murphy. *Machine Learning. A Probabilistic Perspective*. Cambridge, Massachusetts: The MIT Press, 2012.

[Nak+06] Teruo Nakajima et al. "Next-Generation Network Management System". In: *NEC Technical Journal - Next-Generation Network Services* (2006).

[NGM07] NGMN Alliance. "NGMN Use Cases related to Self Organising Network, Overall Description". In: *NGMN White Paper* (May 2007).

[Nie15] Michael A. Nielsen. *Neural Networks and Deep Learning*. Online Book: Determination Press, 2015.

[ØG12] Olav Østerbø and Ole Grøndalen. "Benefits of Self-Organizing Networks (SON) for Mobile Operators". In: *Journal of Computer Networks and Communications Volume 2012* (Sept. 2012).

[PC17] George Philipp and Jaime G. Carbonell. "Nonparametric Neural Networks". In: *5th International Conference on Learning Representations (ICLR 2017)* (Apr. 2017).

[PON13] Kevin Twidle. *Ponder2*. visited on 05/11/2018. 2013. URL: http://ponder2.net/.

[Pre+07] William H. Press et al. *Numerical Recipes. The Art of Scientific Computing*. MA: Cambridge: Cambridge University Press, 2007.

[Qin+13] Wencong Qin et al. "A Q-learning approach for mobility robustness optimization in Lte-Son". In: *2013 15th IEEE International Conference on Communication Technology* (Nov. 2013), pp. 818–822.

[RAD18] Ian Poole. *Radio-Electronics.com*. visited on 13/11/2018. 2018. URL: https://www.radio-electronics.com/info/cellulartelecomms/self-organising-networks-son/basics-tutorial.php.

[RBS11] Raphael Romeikat, Bernhard Bauer, and Henning Sanneck. "Automated Refinement of Policies for Network Management". In: *17th Asia-Pacific Conference on Communications (APCC 2011)* (Oct. 2011), pp. 439–444.

[RHK16] Dennis M. Rose, Sören Hahn, and Thomas Kürner. "Evolution from network planning to SON management using the simulator for mobile networks (SiMoNe)". In: *2016 IEEE 27th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)* (Sept. 2016), pp. 1–2.

[Roj96] Raúl Rojas. *Neural Networks. A Systematic Introduction*. Berlin: Springer-Verlag, 1996.

[Rom12] Raphael Romeikat. *Domain-Specific Development of Event Condition Action Policies*. PhD Thesis. Universität Augsburg, Oct. 2012.

[RW06] Carl E. Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. MA: Cambridge: MIT Press, 2006.

[SBG15] Meryem Simsek, Mehdi Bennis, and Ismail Güvenc. "Context-aware mobility management in HetNets: A reinforcement learning approach". In: *2015 IEEE Wireless Communications and Networking Conference (WCNC)* (Mar. 2015), pp. 1536–1541.

[Sch+08] Lars Christoph Schmelz et al. *Framework for the development of self-organisation methods*. Deliverable D2.4. SOCRATES Project, July 2008.

[Sch+14a] Lars Christoph Schmelz et al. "Demonstrator for objective driven SON operation". In: *2014 11th International Symposium on Wireless Communications Systems (ISWCS)* (Aug. 2014), pp. 506–507.

[Sch+14b] Lars Christoph Schmelz et al. *Integrated SON Management - Policy Transformation and Operational SON Coordination (first results)*. Deliverable D5.2. SEMAFOUR Project, June 2014.

[Sch+14c] Lars Christoph Schmelz et al. "SON Management Demonstrator". In: *2014 IEEE Network Operations and Management Symposium (NOMS)* (May 2014), pp. 1–2.

[Sch+16] Lars Christoph Schmelz et al. "Demonstrator for adaptive SON management". In: *NOMS 2016 - 2016 IEEE/IFIP Network Operations and Management Symposium* (Apr. 2016), pp. 991–992.

[SCL07] Scikit-Learn. *Scikit-Learn - K-Means*. visited on 29/10/2018. 2007. URL: http://scikit-learn.org/stable/modules/clustering.html#k-means.

[Scu+08] Neil Scully et al. *Use Cases for Self-Organising Networks*. Deliverable D2.1. SOCRATES Project, Mar. 2008.

[SD13] K. Gnana Sheela and S.N. Deepa. "Review on Methods to Fix Number of Hidden Neurons in Neural Networks". In: *Mathematical Problems in Engineering, vol. 2013* (May 2013).

[SEM12] SEMAFOUR. *SEMAFOUR website*. visited on 26/10/2018. 2012. URL: http://www.fp7-semafour.eu/.

[SFL14] Lars Christoph Schmelz, Christoph Frenzel, and Simon Lohmüller. "Network Entity and method for Controlling a SON-Function". WIPO Pub. No. WO 2014/191469 A1. Dec. 2014.

[SK05] N. Samaan and A. Karmouch. "An automated policy-based management framework for differentiated communication systems". In: *IEEE Journal on Selected Areas in Communications, Volume 23* (Dec. 2005), pp. 2236–2247.

[SON09] Nokia Siemens Networks. *Self-Organizing Network (SON) Introducing the Nokia Siemens Networks SON Suite – an efficient , future-proof platform for SON Executive summary*. visited on 03/01/2019. 2009. URL: http://cwi.unik.no/images/c/c3/SON%7B%5C_%7Dwhite%7B%5C_%7Dpaper%7B%5C_%7DNSN.pdf.

[Str+06] J. Strassner et al. "Autonomic Networking: Prototype Implementation of the Policy Continuum". In: *2006 1st IEEE International Workshop on Broadband Convergence Networks* (Apr. 2006), pp. 1–10.

[Str03] John Strassner. *Policy-based Network Management. Solutions for the Next Generation*. CA: San Francisco: Morgan Kaufman Publishers, 2003.

[TOH14] Sven Tomforde, Alexander Ostrovsky, and Jörg Hähner. "Load-aware reconfiguration of LTE-antennas dynamic cell-phone network adaptation using organic network control". In: *2014 11th International Conference on Informatics in Control, Automation and Robotics (ICINCO)* (Sept. 2014), pp. 236–243.

[Tsa+13a] Kostas Tsagkaris et al. "SON Coordination in a Unified Management Framework". In: *2013 IEEE 77th Vehicular Technology Conference (VTC Spring)* (June 2013), pp. 1–5.

[Tsa+13b] Kostas Tsagkaris et al. *Unified Management Framework (UMF) Specifications Release 3*. Deliverable D2.3. UniverSelf Project, Nov. 2013.

[UFLDL] Stanford University UFLDL Tutorial. *Multi-Layer Neural Network*. visited on 01/11/2018. URL: http://ufldl.stanford.edu/tutorial/supervised/MultiLayerNeuralNetworks/.

[USS07] Yathiraj B. Udupi, Akhil Sahai, and Sharad Singhal. "A classification-based approach to policy refinement". In: *10th IFIP/IEEE International Symposium on Integrated Network Management (IM'07)* (May 2007), pp. 785–788.

[WA13] Andrew Gordon Wilson and Ryan Prescott Adams. "Gaussian Process Kernels for Pattern Discovery and Extrapolation". In: *30th International Conference on Machine Learning (ICML 2013)* (June 2013), pp. 1067–1075.

[WB09] Zhou Wang and Alan C. Bovik. "Mean squared error: Love it or leave it? A new look at Signal Fidelity Measures". In: *IEEE Signal Processing Magazine Volume 29* (2009), pp. 98–117.

[WEK19] The University of Waikato. *Weka 3: Data Mining Software in Java*. visited on 07/01/2019. URL: https://www.cs.waikato.ac.nz/ml/weka/.

[Wes+01] A. Westerlinen et al. "Terminology for Policy-Based Management Status". In: *Internet Engineering Task Force, Request for Comments 3198* (Nov. 2001).

[WM05] Cort J. Wilmott and Kenji Matsuura. "Advantages of the Mean Absolute Error (MAE) over the Root Mean Square Error (RMSE) in Assessing Average Model Performance". In: *Climate Research 30* (Dec. 2005), pp. 79–82.

[Yan+18] Yang Yang et al. "DECCO: Deep-Learning Enabled Coverage and Capacity Optimization for Massive MIMO Systems". In: *IEEE Access, vol. 6* (Apr. 2018), pp. 23361–23371.

List of Acronyms

3GPP	3rd Generation Partnership Project
ANN	Artificial Neural Network
ANR	Automatic Neighbor Relationship Setup
ARFF	Attribute-Relation File Format
ASM	Adaptive SON Management
CAPEX	Capital Expenditures
CCO	Coverage and Capacity Optimization
CIO	Cell Individual Offset
CL	Physical Cell Load
CM	Configuration Management
COD	Cell Outage Detection
COC	Cell Outage Compensation
COMMUNE	Cognitive Network Management under Uncertainty
CQI	Channel Quality Indicator
CRN	Cognitive Radio Network
CSM	Cognitive SON Management
DCR	Dropped-Call Rate
DSL	Domain-Specific Language
EC	Energy Consumption
ECA	Event Condition Action
EDGE	Enhanced Data Rates for GSM Evolution
ES	Energy Saving
EU	European Union
FM	Failure Management
GANA	Generic Autonomic Network Architecture
GP	Granularity Period
GPR	Gaussian Process Regression
GUI	Graphical User Interface

HO	hand-over
HOSR	Hand-over Success Rate
IMPEX	Implementational Expenditures
IoT	Internet of Things
JSON	JavaScript Object Notation
KNN	k-nearest Neighbors Regression
KPI	Key Performance Indicator
LTE	Long Term Evolution
LR	Linear Regression
LOO	Leave One Out
MAE	Mean Absolute Error
MIMO	multiple-input multiple-output
MLB	Mobility Load Balancing
MLP	Multilayer Perceptron
MNN	Multilayer Neural Network
MNO	Mobile Network Operator
MRO	Mobility Robustness Optimization
MSE	Mean Squared Error
NCP	Network Configuration Parameter
NE	Network Element
NEM	Network Empowerment Mechanism
NGMN	Next Generation Mobile Networks
NM	Network Management
OAM	Operation, Administration and Maintenance
ODSM	Objective-driven SON Management
OPEX	Operational Expenditures
PBM	Policy-based Management
PBSM	Policy-based SON Management
PCI	Physical Cell ID
PDP	Policy Decision Point

PEP	Policy Enforcement Point
PiPo	Hand-over Ping-Pong Rate
PM	Performance Management
PUK	Pearson VII Universal Kernel
QoE	Quality of Experience
QoS	Quality of Service
RACH	Random Access Channel
RAN	Radio Access Network
RAT	Radio Access Technology
RBF	Radial Basis Function
RLF	Radio Link Failure
RMSE	Root Mean Squared Error
RSS	Residual Sum of Squares
SCP	SON Function Configuration Parameter
SCV	SON Function Configuration Parameter Value
SEASON	System Experience of Advanced SON
SEMAFOUR	Self-Management for Unified Heterogeneous Radio Access Networks
SINR	Signal to Interference plus Noise Ratio
SLA	Service Level Agreement
SOCRATES	Self-Optimisation and Self-Configuration in Wireless Networks
SON	Self-Organizing Network
TTT	Time to Trigger
UE	User Equipment
UMTS	Universal Mobile Telecommunication System

List of Figures

List of Tables

List of Listings

A Evaluation

A.1. Manufacturer Effect Model

Table A.1.: Manufacturer effect model for CLASS_1

Traffic	CCO Set	MRO Set	MLB Set	CQI Effect	PiPo Effect	CL Effect
high	-1	0	-1	0.27	0.07	0.86
high	-1	-1	2	0.36	0.11	0.81
high	1	-1	-1	0.35	0.03	0.89
high	-1	-1	1	0.35	0.12	0.81
high	0	-1	-1	0.35	0.03	0.89
high	-1	2	-1	0.36	0.11	0.81
high	2	-1	-1	0.28	0.03	0.85
high	-1	-1	0	0.36	0.11	0.81
high	-1	1	-1	0.36	0.11	0.81
low	-1	0	-1	0.36	0.06	0.64
low	-1	-1	2	0.51	0.09	0.6
low	1	-1	-1	0.52	0.04	0.63
low	-1	-1	1	0.5	0.1	0.58
low	0	-1	-1	0.51	0.04	0.63
low	-1	2	-1	0.51	0.09	0.61
low	2	-1	-1	0.39	0.03	0.59
low	-1	-1	0	0.51	0.09	0.56
low	-1	1	-1	0.51	0.09	0.6

Table A.2.: Manufacturer effect model for CLASS_2

Traffic	CCO Set	MRO Set	MLB Set	CQI Effect	PiPo Effect	CL Effect
high	-1	0	-1	0.26	0.07	0.69
high	-1	-1	2	0.35	0.05	0.61
high	1	-1	-1	0.36	0.08	0.69
high	-1	-1	1	0.35	0.05	0.6
high	0	-1	-1	0.36	0.08	0.69
high	-1	2	-1	0.36	0.05	0.6
high	2	-1	-1	0.27	0.05	0.65
high	-1	-1	0	0.36	0.05	0.55
high	-1	1	-1	0.35	0.05	0.61
low	-1	0	-1	0.35	0.07	0.42
low	-1	-1	2	0.49	0.04	0.34
low	1	-1	-1	0.51	0.08	0.43
low	-1	-1	1	0.48	0.05	0.34
low	0	-1	-1	0.51	0.08	0.43
low	-1	2	-1	0.49	0.05	0.34
low	2	-1	-1	0.37	0.04	0.38
low	-1	-1	0	0.49	0.05	0.31
low	-1	1	-1	0.49	0.04	0.34

Table A.3.: Manufacturer effect model for CLASS_3

Traffic	CCO Set	MRO Set	MLB Set	CQI Effect	PiPo Effect	CL Effect
high	-1	0	-1	0.21	0.03	0.97
high	-1	-1	2	0.26	0.03	0.97
high	1	-1	-1	0.21	0.02	1.0
high	-1	-1	1	0.27	0.02	0.95
high	0	-1	-1	0.21	0.02	1.0
high	-1	2	-1	0.26	0.02	0.99
high	2	-1	-1	0.25	0.0	0.88
high	-1	-1	0	0.26	0.03	0.97
high	-1	1	-1	0.26	0.03	0.97
low	-1	0	-1	0.24	0.02	0.84
low	-1	-1	2	0.3	0.03	0.8
low	1	-1	-1	0.25	0.02	0.99
low	-1	-1	1	0.31	0.03	0.83
low	0	-1	-1	0.25	0.02	0.99
low	-1	2	-1	0.31	0.02	0.81
low	2	-1	-1	0.29	0.0	0.8
low	-1	-1	0	0.31	0.03	0.83
low	-1	1	-1	0.3	0.03	0.8

Table A.4.: Manufacturer effect model for CLASS_4

Traffic	CCO Set	MRO Set	MLB Set	CQI Effect	PiPo Effect	CL Effect
high	-1	0	-1	0.25	0.0	0.29
high	-1	-1	2	0.35	0.0	0.2
high	1	-1	-1	0.38	0.0	0.31
high	-1	-1	1	0.37	0.0	0.27
high	0	-1	-1	0.38	0.0	0.31
high	-1	2	-1	0.36	0.0	0.22
high	2	-1	-1	0.27	0.0	0.29
high	-1	-1	0	0.36	0.0	0.24
high	-1	1	-1	0.35	0.0	0.2
low	-1	0	-1	0.33	0.0	0.16
low	-1	-1	2	0.48	0.0	0.11
low	1	-1	-1	0.53	0.0	0.16
low	-1	-1	1	0.5	0.0	0.15
low	0	-1	-1	0.53	0.0	0.16
low	-1	2	-1	0.5	0.0	0.13
low	2	-1	-1	0.37	0.0	0.16
low	-1	-1	0	0.5	0.0	0.13
low	-1	1	-1	0.48	0.0	0.11

Table A.5.: Manufacturer effect model for CLASS_5

Traffic	CCO Set	MRO Set	MLB Set	CQI Effect	PiPo Effect	CL Effect
high	-1	0	-1	0.58	0.0	0.72
high	-1	-1	2	0.62	0.0	0.87
high	1	-1	-1	0.21	0.0	0.58
high	-1	-1	1	0.62	0.0	0.87
high	0	-1	-1	0.21	0.0	0.58
high	-1	2	-1	0.62	0.0	0.83
high	2	-1	-1	0.95	0.0	0.44
high	-1	-1	0	0.62	0.0	0.86
high	-1	1	-1	0.62	0.0	0.87

A.2. Learned Effect Model Results

A.2.1. Linear Regression

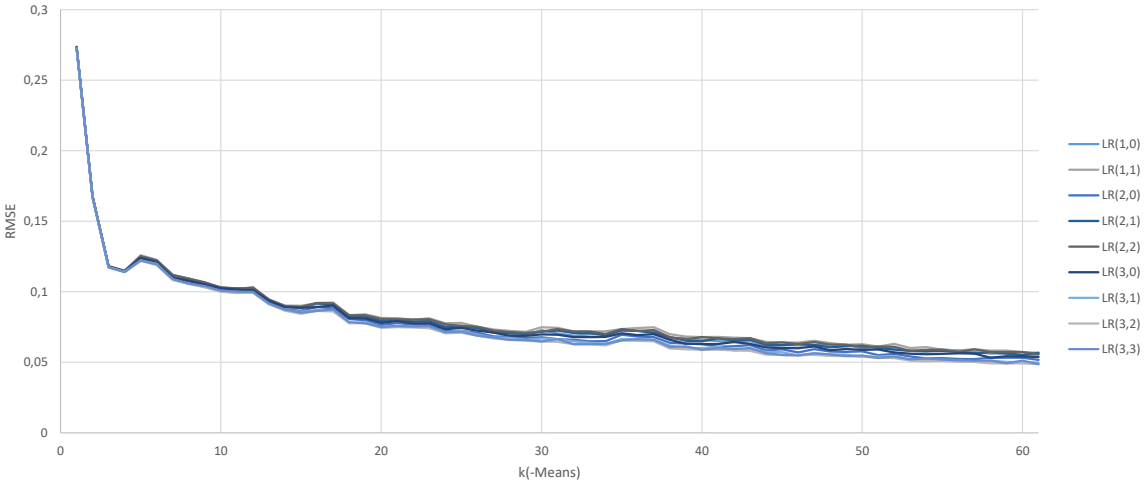


Figure A.1.: LR: Results in terms of RMSE for KPI CL in a high traffic scenario

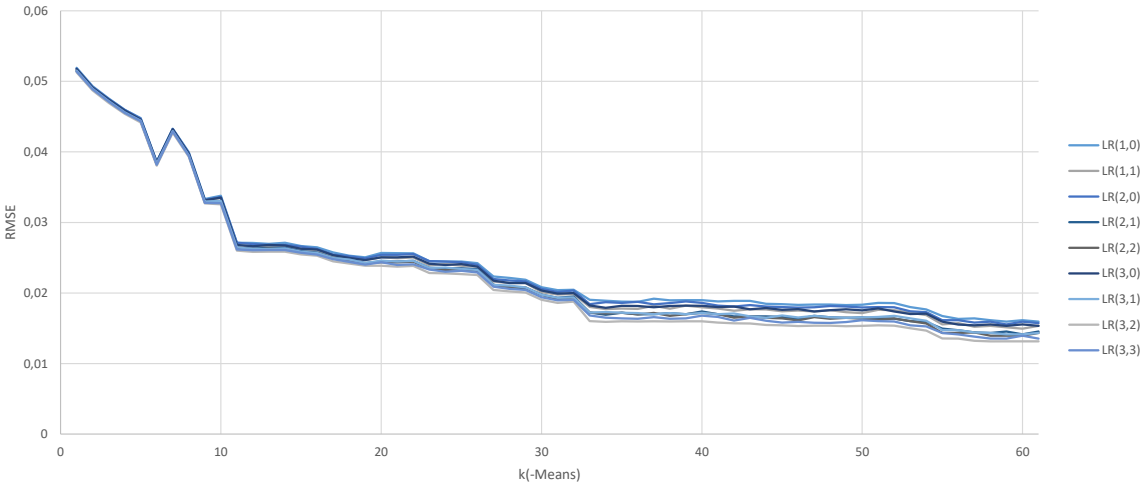


Figure A.2.: LR: Results in terms of RMSE for KPI PiPo in a high traffic scenario

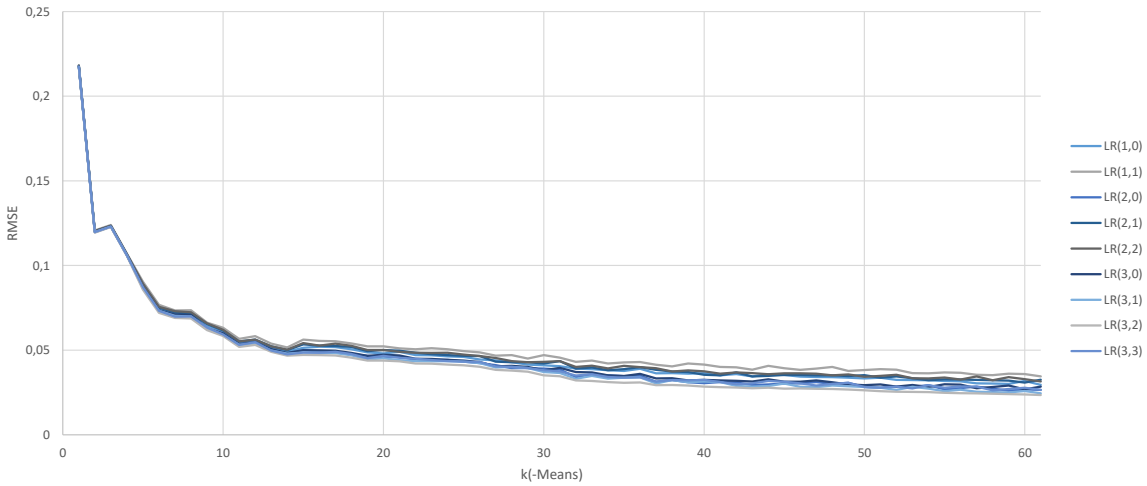


Figure A.3.: LR: Results in terms of RMSE for KPI CL in a low traffic scenario

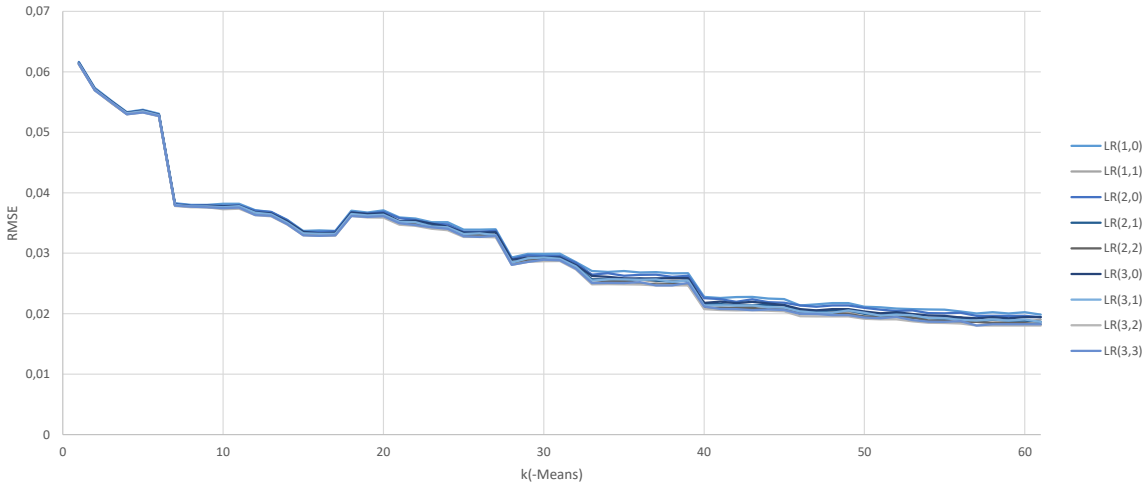


Figure A.4.: LR: Results in terms of RMSE for KPI PiPo in a low traffic scenario

A.2.2. Gaussian Process Regression

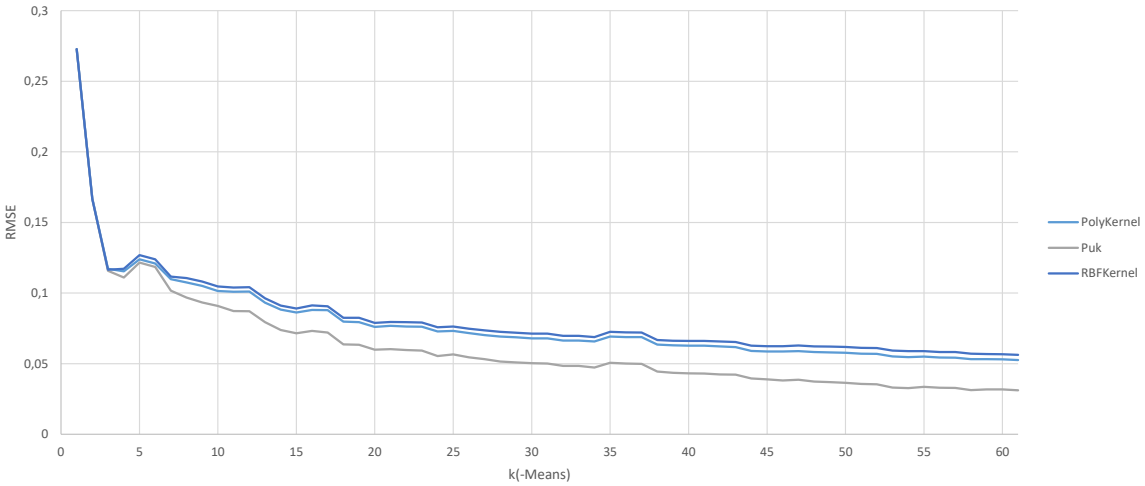


Figure A.5.: GPR: Results in terms of RMSE for KPI CL in a high traffic scenario

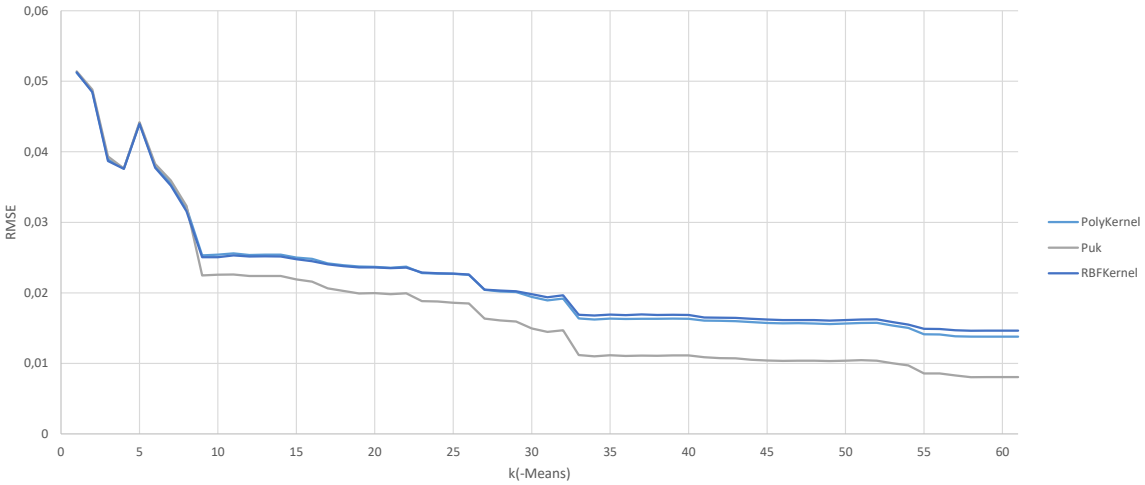


Figure A.6.: GPR: Results in terms of RMSE for KPI PiPo in a high traffic scenario

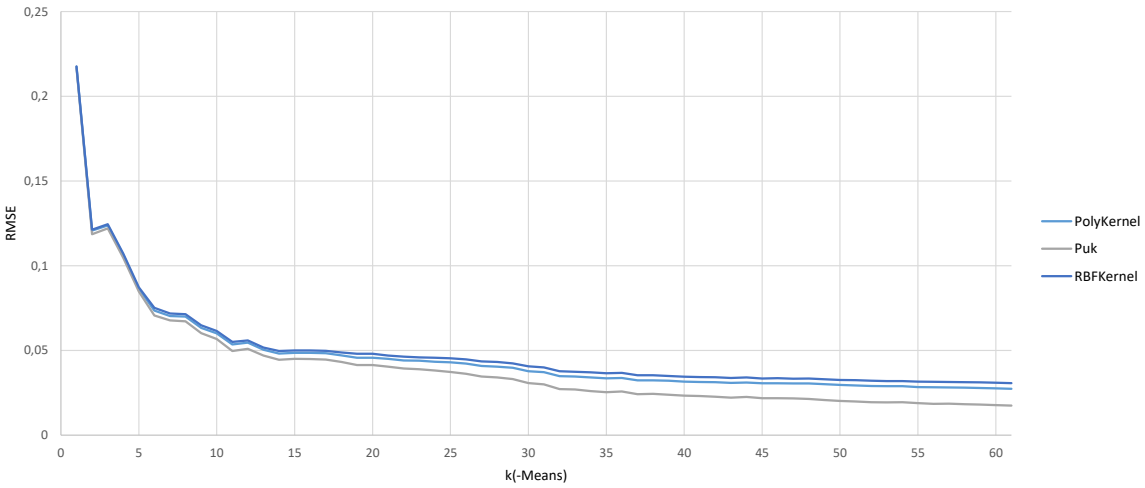


Figure A.7.: GPR: Results in terms of RMSE for KPI CL in a low traffic scenario

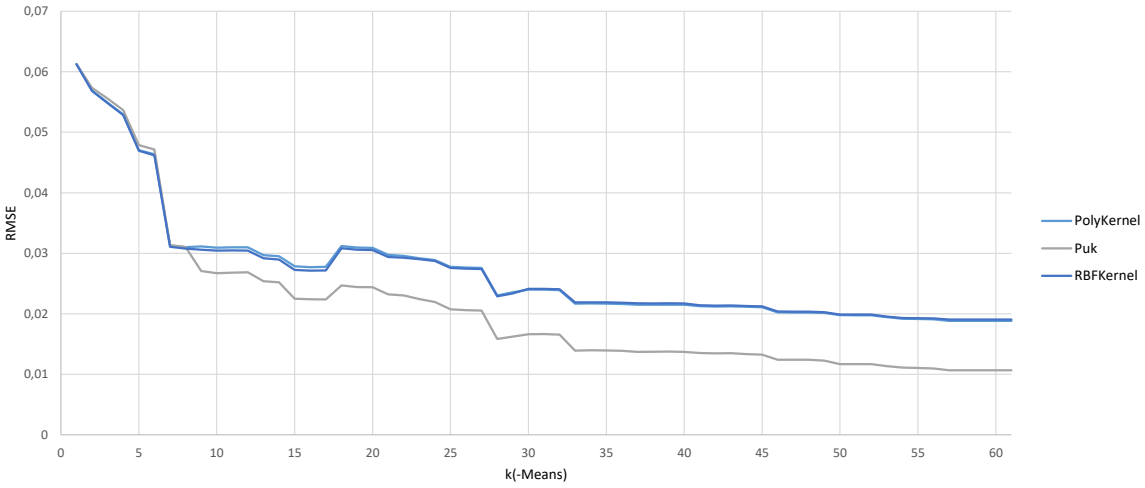


Figure A.8.: GPR: Results in terms of RMSE for KPI PiPo in a low traffic scenario

A.2.3. k-Nearest Neighbors Regression

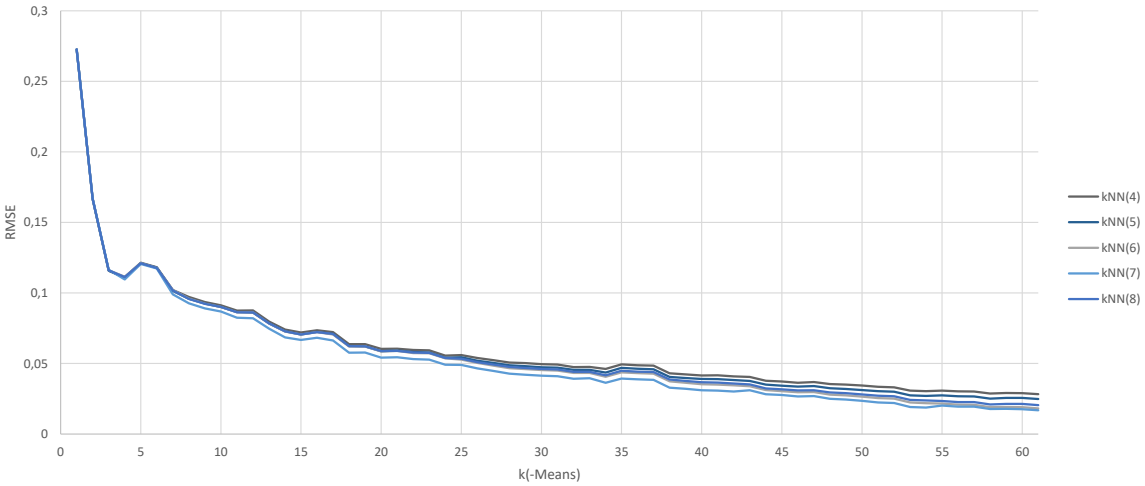


Figure A.9.: KNN: Results in terms of RMSE for KPI CL in a high traffic scenario

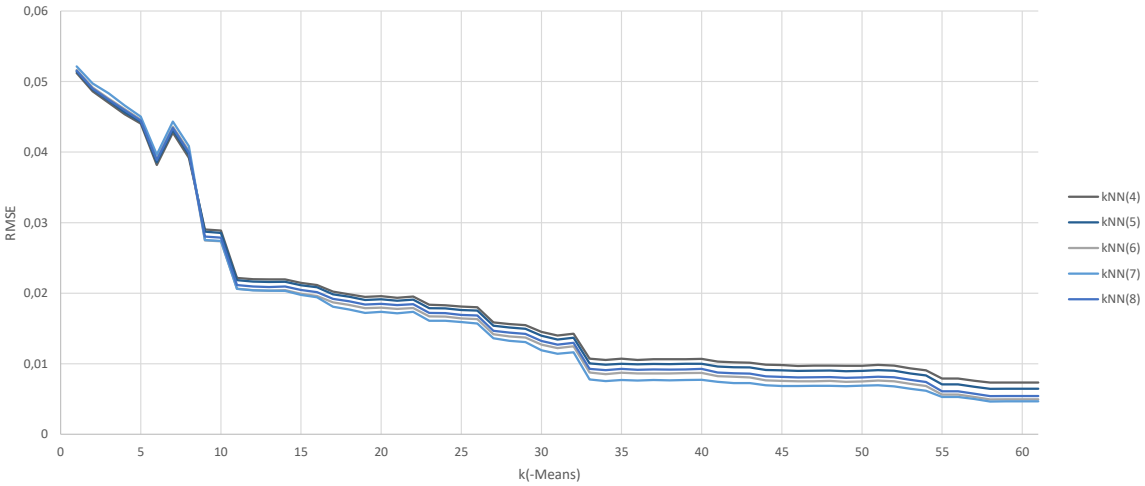


Figure A.10.: KNN: Results in terms of RMSE for KPI PiPo in a high traffic scenario

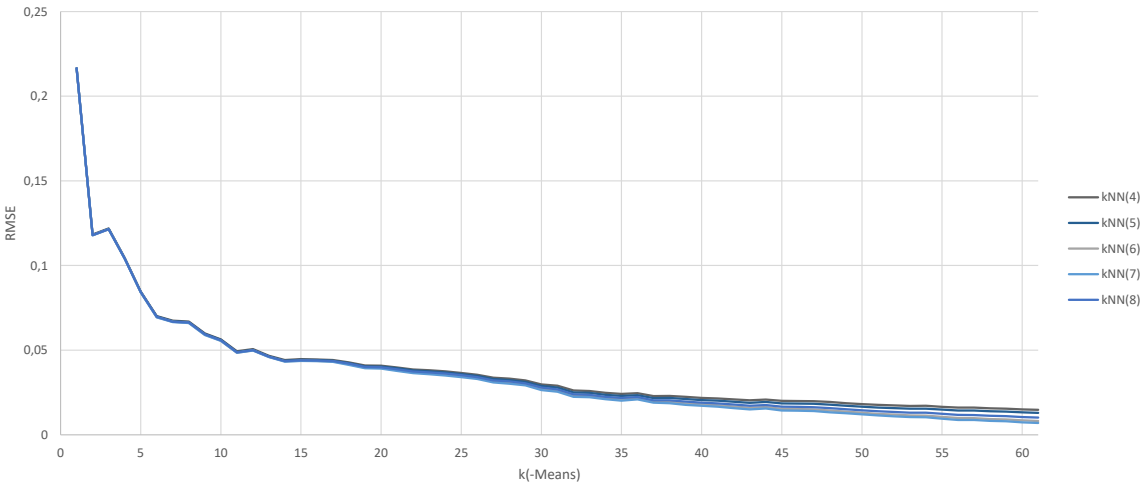


Figure A.11.: KNN: Results in terms of RMSE for KPI CL in a low traffic scenario

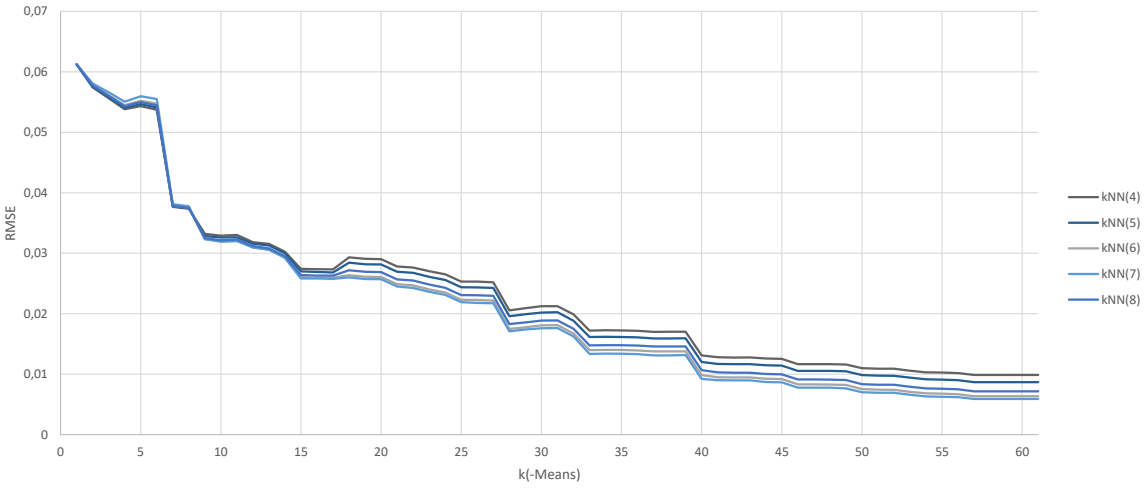


Figure A.12.: KNN: Results in terms of RMSE for KPI PiPo in a low traffic scenario

A.2.4. Artificial Neural Network

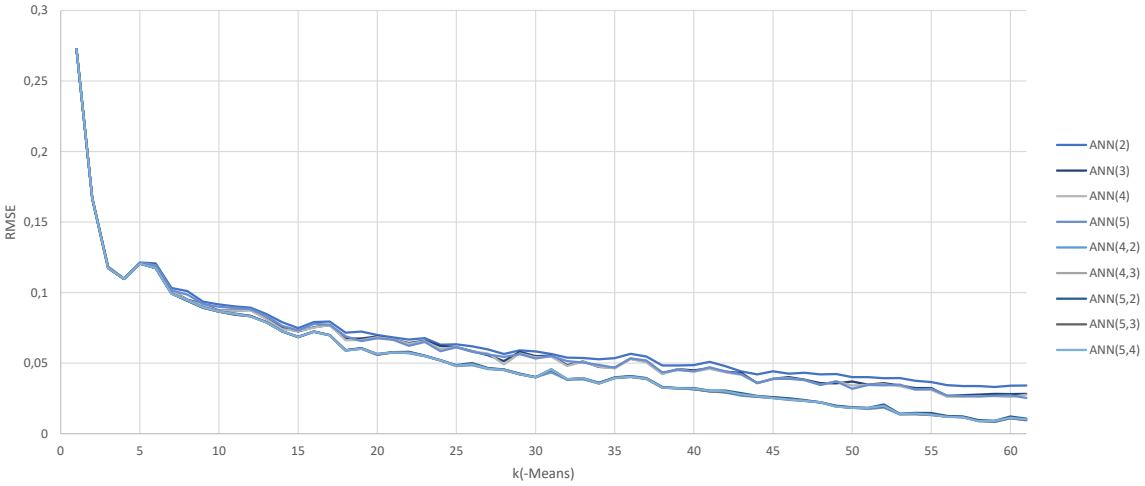


Figure A.13.: ANN: Results in terms of RMSE for KPI CL in a high traffic scenario

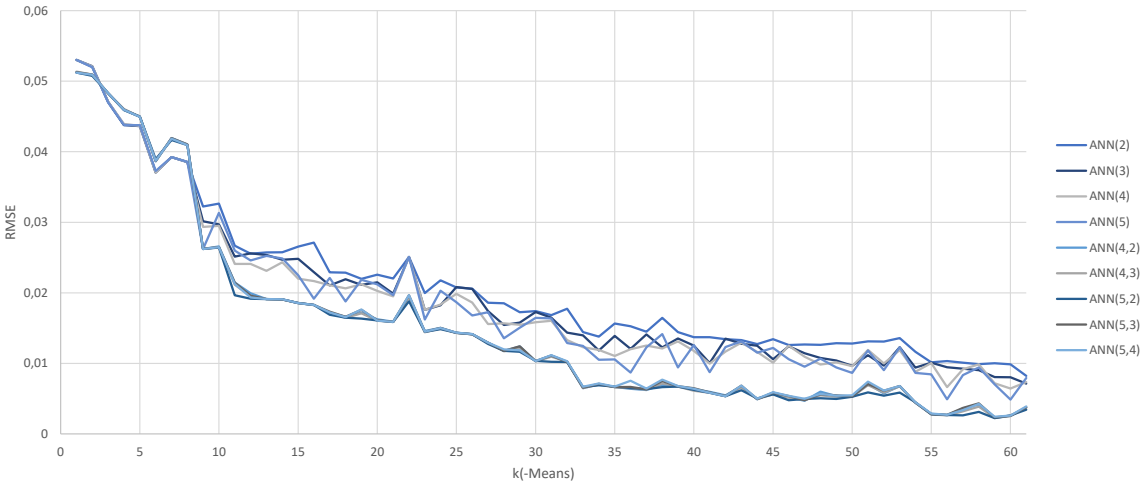


Figure A.14.: ANN: Results in terms of RMSE for KPI PiPo in a high traffic scenario

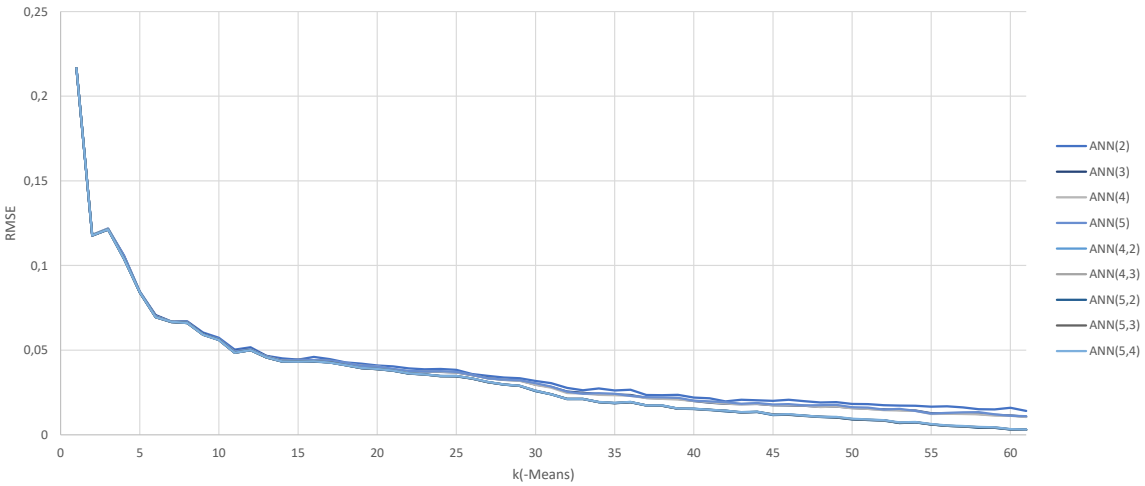


Figure A.15.: ANN: Results in terms of RMSE for KPI CL in a low traffic scenario

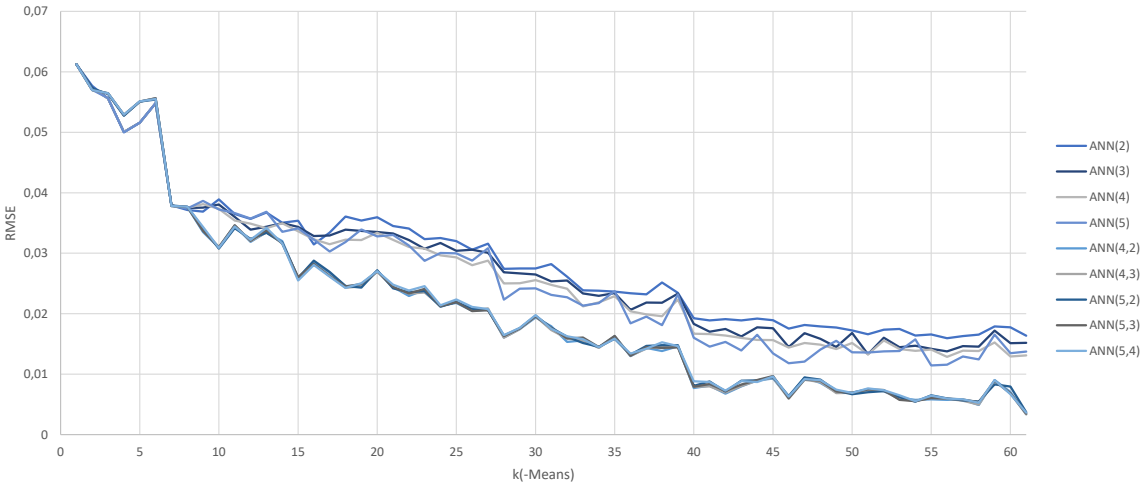


Figure A.16.: ANN: Results in terms of RMSE for KPI PiPo in a low traffic scenario

A.2.5. Comparison of Different Algorithms

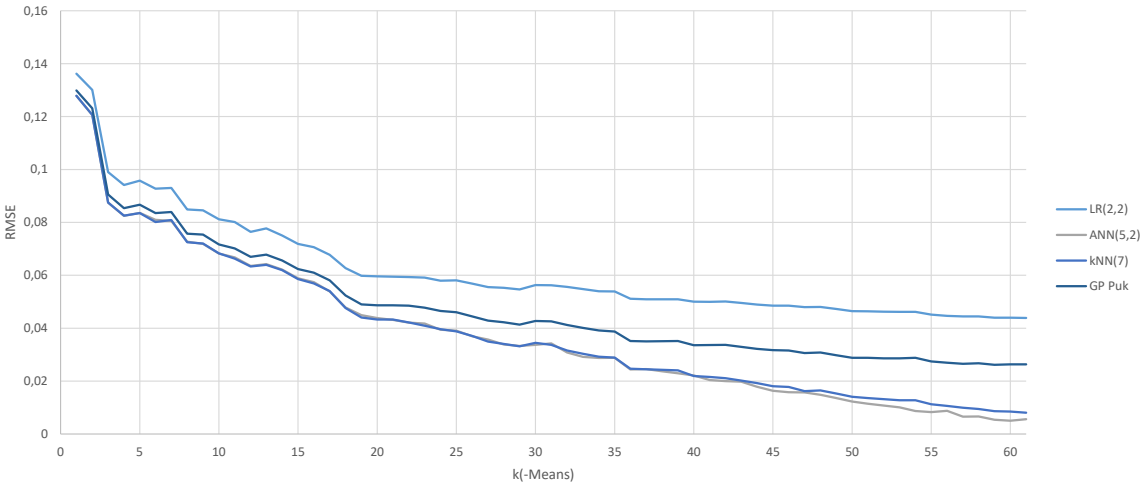


Figure A.17.: Overall results in terms of RMSE for KPI CQI in a low traffic scenario

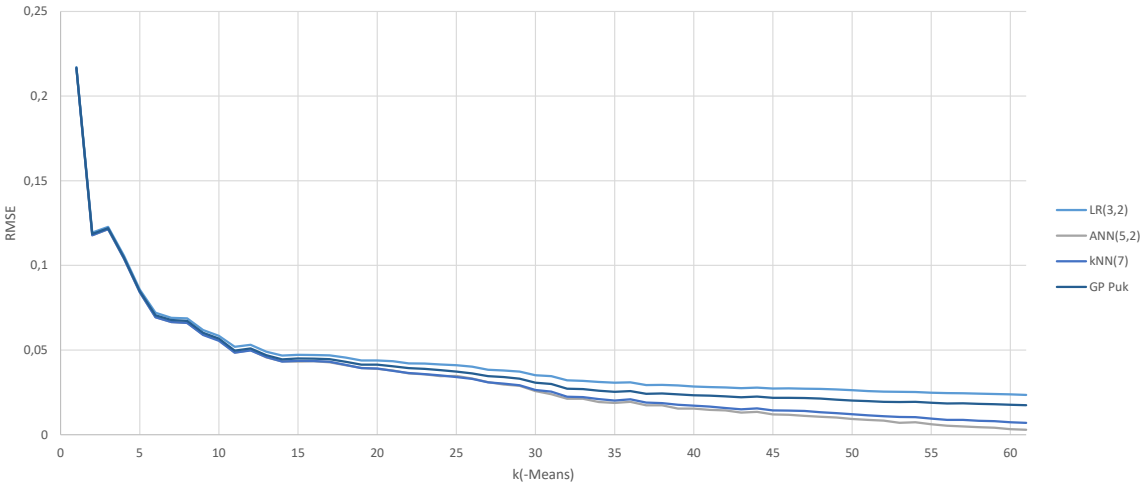


Figure A.18.: Overall results in terms of RMSE for KPI CL in a low traffic scenario

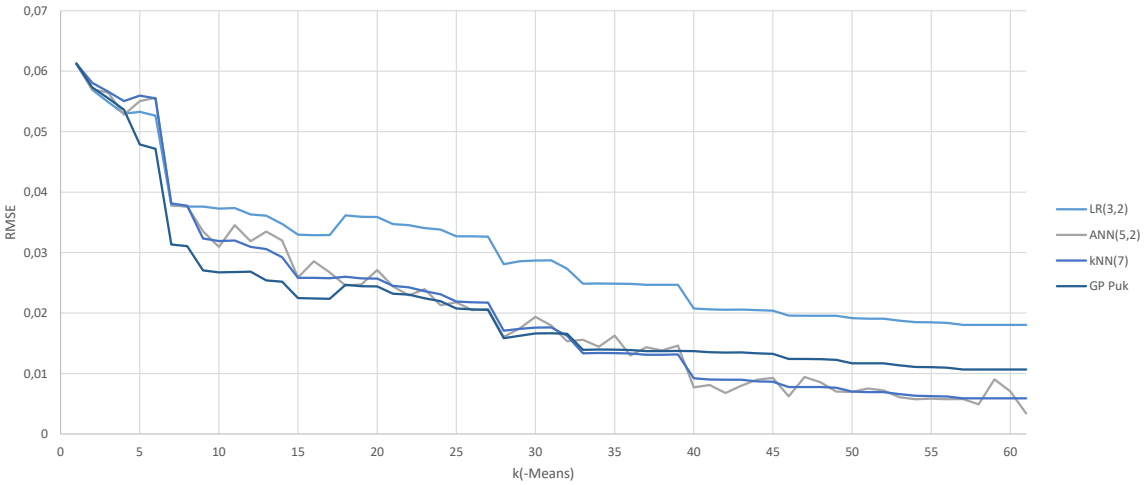


Figure A.19.: Overall results in terms of RMSE for KPI PiPo in a low traffic scenario