UNIVERSITÄT AUGSBURG
Universitätsbibliothek

OPEN ACCESS
OPUS AUGSBURG

An approach to robust resource allocation in large-scale systems of systems

Oliver Kosak, Gerrit Anders, Florian Siefert, Wolfgang Reif

Angaben zur Veröffentlichung / Publication details:

Kosak, Oliver, Gerrit Anders, Florian Siefert, and Wolfgang Reif. 2015. "An approach to robust resource allocation in large-scale systems of systems." In *2015 IEEE 9th International Conference on Self-Adaptive and Self-Organizing Systems, 21-25 September 2015, Cambridge, MA, USA*, edited by Jan-Philipp Steghöfer, Emma Hart, and Gregory T. Sullivan, 1–10. Piscataway, NJ: IEEE. https://doi.org/10.1109/saso.2015.8.

Nutzungsbedingungen / Terms of use:

licgercopyright

Dieses Dokument wird unter folgenden Bedingungen zur Verfügung gestellt: / This document is made available under these conditions:
Deutsches Urheberrecht
Weitere Informationen finden Sie unter: / For more information see:
https://www.uni-augsburg.de/de/organisation/bibliothek/publizieren-zitieren-archivieren/publiz/

2015 IEEE 9th International Conference on Self-Adaptive and Self-Organizing Systems

An Approach to Robust Resource Allocation in Large-Scale Systems of Systems

Oliver Kosak, Gerrit Anders, Florian Siefert, and Wolfgang Reif
Institute for Software & Systems Engineering, Augsburg University, Germany
E-Mail: {oliver.kosak, anders, siefert, reif}@informatik.uni-augsburg.de

***Abstract*—Resource allocation, in terms of balancing supply and demand, is a common problem in many mission-critical systems, such as smart grids. If the participants' contribution is subject to inertia, they have to solve the problem for a specific time span in advance, that is, create schedules on the basis of demand predictions. To improve the system's stability, the participants not only have to anticipate the predictions' inherent uncertainties, but also provide an appropriate amount of degrees of freedom, i.e., reserves, that enable reactive contribution adjustments. Reserves are of particular interest because the problem's complexity and uncertainty call for rather coarse-grained schedules and an anticipation can turn out to be wrong.**

In this paper, we present an approach to robust resource allocation in self-organizing hierarchical systems that is flanked by an auction-based algorithm. To deal with uncertain demand, agents learn characteristic prediction errors and create schedules for different possible developments of the demand. This allows the agents to choose the most suitable schedule at runtime. Further, they schedule feedback-driven reserves to be able to cope with unforeseen situations. Based on the example of creating power plant schedules in decentralized autonomous power management systems, we show that our auction-based algorithm clearly outperforms a regio-central approach.

***Index Terms*—Resource Allocation; Uncertainty; Electronic Markets; Online Stochastic Optimization; Smart Grids**

I. Resource Allocation under Uncertainty

In a resource allocation problem (RAP), the goal is to stipulate the contributions of the system components in a way that their sum satisfies a given demand. Solving such a RAP is the main issue in several technical systems. In gas pipeline and water supply systems, for instance, the challenge is to maintain the system's pressure at a certain level. Regarding district heating systems, the network temperature has to be kept between specific bounds. Similarly, the main task in power management systems (PMSs) is to maintain the balance between power production and consumption at all times [1].

In this paper, we address a one-good RAP without externalities [2] that has to be solved by a large set of agents $\mathcal{A} = \{a_1, \ldots, a_m\}$. The primary goal is to find an allocation so that, in each time step t, the sum of the individual agents' contributions matches a demand – that is imposed by the environment – as accurately as possible. Further, the costs of satisfying the demand should be kept low. Because we assume that the demand exceeds the limited resources of a single agent, the agents have to solve the RAP in cooperation.

In various systems that control physical devices, the agents' possible behavior is subject to inertia. In such cases, the agents' contribution might not change quickly enough to reactively adapt to the demand in all situations. Consequently, their contribution has to be specified *proactively* in the form of *schedules* for all N time steps contained in a *scheduling window* $\mathcal{W}$ that defines a fixed time span. In PMSs, this problem is also known as *economic load dispatch* [3] or *unit commitment* [4]. It can be solved, e.g., by centralized [5], [6], market-based [7], or decentralized [8] algorithms.

In particular in large-scale systems, the scheduling problem introduces two central and interconnected challenges:

1) **Scalability:** Solving the scheduling problem is NP-hard [9]; both with regard to the number $|\mathcal{A}|$ of agents involved and time steps N schedules are created for in advance, resulting in a likely complexity of $O\left(2^{|\mathcal{A}| \cdot N}\right)$.
2) **Uncertainty:** The RAP has to be solved in spite of uncertainties. These are introduced by the environment in the form of inaccurate demand predictions as well as by the system participants themselves. In this paper, we focus on uncertain demand, i.e., *unintentional* uncertainties.

Regarding scalability, hierarchical problem decomposition has been proposed as a generic approach to deal with large-scale systems prohibiting a centralized solution [10], [11]. As we showed in [12], agents can establish and maintain appropriate hierarchical system structures using self-organization. While this addresses the exponent $|\mathcal{A}|$, a too fine-grained *schedule resolution* $\Delta\pi$ (we define $\Delta\pi$ as a multiple of the difference Δt between two successive time steps t and $t+1$), which unnecessarily increases the number of time steps N in $\mathcal{W} = \{t_{\text{now}} + i \cdot \Delta\pi \mid i \in \{1, \ldots, N\}\}$, should also be avoided to further deal with the problem's complexity. At the same time, a too fine-grained schedule resolution is not useful because, given that demand predictions tend to become more accurate as a future point in time approaches, uncertainties require that schedules are periodically revised at runtime. A recalculation is needed at least after $N \cdot \Delta\pi$ time steps.

Especially in mission-critical systems, such as supply systems, the ability to deal with uncertainties becomes a major concern because their failure can have massive consequences for people, industries, and public services. In such cases, the system's stability and availability is of utmost importance, meaning that the demand has to be satisfied despite fluctuations and unexpected events. In PMSs, for example, our RAP's demand corresponds to the so-called *residual load* which is defined as the difference between the overall non-dispatchable consumption and the output of all non-dispatchable power

plants. The dispatchable prosumers have to fulfill the residual load in spite of fluctuations originating from changing weather conditions and stochastic consumer behavior (we use the term "prosumer" to refer to producers and consumers). As for unexpected events, dispatchable prosumers might not be able to comply with their schedules in the light of technical difficulties.

For this reason, uncertainties not only have to be anticipated but also quantified in order to obtain meaningful schedules that allow the system to operate more robustly and efficiently. Since uncertainties are subject to change over time, they must be assessed at runtime. *Trust-based scenarios* (TBSs) [13] have been proposed as an approach to probabilistic model creation. As other approaches based on the social concept of *trust*, TBSs assume that prior observations are indicative of future behavior. On the basis of past experiences, TBSs calculate the empirical discrete probability distribution reflecting an agent's or the environment's behavior. Each TBS thus represents an expected behavior over a series of future time steps and has a certain probability of occurrence. Since such an expectation can turn out to be wrong, we presented the idea of improving robustness by creating schedules for multiple TBSs in [14]. This approach allows to **(1)** anticipate *aleatoric*, i.e., intrinsic random and irreducible, uncertainties and **(2)** reduce *epistemic*, i.e., systematic, uncertainties by putting more effort into the optimization of likely scenarios. That way, the agents self-improve their *flexibility* in dealing with expected uncertainties as they can choose the most appropriate schedule at runtime.

Using expectations, uncertainties manifest themselves in the form of deviations between the anticipated demand (i.e., the "presumed *realization*" of the demand) and its (actual) realization. To maintain the balance between supply and demand, we have to allow the agents to temporarily disregard their scheduled contributions in order to reactively compensate for these deviations, especially for the following three reasons: **(R1)** Due to complexity and uncertainties, we calculate schedules for a coarse-grained time pattern of multiples of $\Delta\pi$, whereas the demand has to be satisfied in a fine-grained time pattern of $\Delta t \leq \Delta\pi$. In the synchronous grid of Continental Europe, for instance, schedules are typically created with a resolution of $\Delta\pi = 15$ minutes, whereas imbalances have to be detected and compensated for within seconds to ensure the grid's stable operation [1]. **(R2)** Some agents might contribute according to the wrong scenario. Such a mistake is caused by a false expectation how the demand will develop, e.g., from t_{now} until $t_{\text{now}} + \Delta\pi$. **(R3)** The system must be able to deal with unforeseen developments of the demand, i.e., even those that are not captured in a TBS. In [15], we showed that TBS-based schedules can proactively guide reactive decisions, i.e., self-adaptation, of inert agents, thereby improving the system's efficiency and stability. Increasing the agents' available degrees of freedom by scheduling *reserves* strengthens their ability to cope with unforeseen situations, i.e., aleatoric uncertainties.

All these aspects, anticipating uncertainties, scheduling for several possible scenarios, and incorporating reserves aim at increasing the system's robustness with respect to epistemic and aleatoric uncertainties. Because of the problem's dynamic and uncertain nature, optimal solutions are not relevant and not even desired, though. The effort would be out of proportion to the benefit. This justifies the application of heuristics.

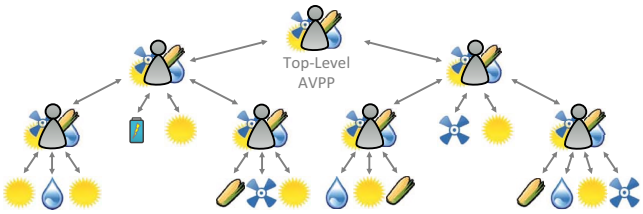

Fig. 1. Hierarchical system structure of a future autonomous and decentralized power management system: Prosumers are structured into systems of systems represented by AVPPs that act as intermediaries to decrease the complexity of control and scheduling. AVPPs can be part of other AVPPs.

With regard to these challenges, the contributions of this paper are threefold: **(C1)** We outline a robust formalization of the scheduling problem on the basis of TBSs, *online stochastic optimization* [16], and feedback-driven reserves (see Sect. III). **(C2)** For its scalable solution in large-scale systems of systems, we present a regionalized auction-based mechanism based on [7] (see Sect. IV). **(C3)** In our evaluation (see Sect. V), we show that it outperforms a regio-central approach that is based on a state-of-the-art optimizer. Throughout the paper, the problem of creating schedules in a decentralized autonomous PMS serves as illustrative example (see Sect. II). Related work is discussed in Sect. VI, before we conclude the paper and give an outlook on future work in Sect. VII.

II. Case Study

The wide-spread installation of weather-dependent power plants as well as the advent of new consumer types like electric vehicles put a lot of strain on power grids. Additionally, small dispatchable power plants (e.g., biogas plants) owned by individuals or cooperatives feed in power without external control. To save expenses, gain more flexibility, and deal with uncertainties, future *autonomous* PMSs have to take advantage of the full potential of dispatchable prosumers by incorporating them into the scheduling scheme. Further, uncertainties have to be (1) anticipated when creating schedules and (2) compensated for locally to prevent their propagation through the system.

In [17], we presented the concept of *Autonomous Virtual Power Plants* (AVPPs) as an approach to meet the challenges of future PMSs. AVPPs represent self-organizing groups of two or more power plants of various types. Each AVPP has to satisfy a fraction of the overall demand by periodically calculating schedules for its dispatchable power plants (see Sect. III). The overall demand is the sum of the demand in each AVPP's local environment consisting of non-dispatchable prosumers. Depending on its composition, an AVPP's local demand is either positive or negative. To avoid affecting other parts of the system, each AVPP's dispatchable power plants have to reactively compensate for deviations resulting from fluctuations of the uncertain local demand.

To be able to hold the balance between energy supply and demand despite changing AVPP-internal or environmental conditions, AVPPs autonomously reorganize their structure at

runtime. In particular, if an AVPP cannot repeatedly satisfy its assigned fraction of the overall demand or compensate for its local uncertainties, it triggers a reorganization. The goal is to form a structure of similar AVPPs that are likely to feature a heterogeneous composition. This promotes robustness in two ways: (1) By distributing unreliable power plants among AVPPs, the chance of fluctuations is reduced and the system's robustness increases. (2) By balancing the AVPPs' degrees of freedom (e.g., by mixing different generator types), their ability to locally deal with uncertainties, i.e., fluctuations, is promoted.

To cope with the vast number of dispatchable power plants, we proposed a self-organizing hierarchical structure in which AVPPs act as *intermediaries* (see Fig. 1). By decomposing the system, the number of subordinate[1] dispatchable power plants (including AVPPs) each AVPP controls is reduced, which results in shorter scheduling times for the overall system.

While we focus on power generation in this paper, both dispatchable and non-dispatchable power consumers can easily be integrated into AVPPs as well as the presented algorithms.

III. Robust Hierarchical Resource Allocation

In this section, we formalize the problem of robust schedule creation from a "regio-central" perspective. We start with the top-down creation of TBS-based schedules in Sect. III-A. In Sect. III-B, we extend this problem by scheduling reserves.

A. Scheduling on the Basis of Trust-based Scenario Trees

With regard to the hierarchical system structure, the agents autonomously create schedules in a regionalized and top-down manner. In this setting, inner nodes act as *intermediaries* $\lambda \in \mathcal{I}$ (with $\mathcal{I} \subset \mathcal{A}$). We refer to the root of the hierarchy as the top-level intermediary $\Lambda \in \mathcal{I}$. Each intermediary λ represents the group of its subordinate agents $\mathcal{A}_\lambda \subset \mathcal{A}$ (e.g., if $\mathcal{I} = \{\Lambda\}$, $\mathcal{A}_\Lambda = \mathcal{A} \setminus \{\Lambda\}$), i.e., a subsystem. As the hierarchy constitutes a tree, the sets of subordinate agents are pairwise disjoint.

When creating schedules, the top-level intermediary Λ first determines the overall expected demand D_Λ^T for the scheduling window $\mathcal{W}$ by aggregating the local demand the intermediaries expect in their local environment. Afterwards, Λ distributes D_Λ^T to its subordinate agents $a \in \mathcal{A}_\Lambda$ in the form of schedules S_a^T (Λ's own schedule S_Λ^T is equivalent to D_Λ^T). Λ's subordinate intermediaries $\lambda \in \mathcal{I}_\Lambda$ (with $\mathcal{I}_\Lambda = \mathcal{A}_\Lambda \cap \mathcal{I}$) are, in turn, responsible for redistributing the demand D_λ^T stipulated in their schedule S_λ^T to their own subordinate agents $\mathcal{A}_\lambda$ etc. Ultimately, the overall expected demand is distributed in the form of schedules to all agents $a \in \mathcal{A}$ in a top-down manner.

As stated before, the (local) demand is based on a stochastic process driven by the environment (consider, e.g., meteorological influences). To improve the satisfaction of the actual demand, i.e., its realization, intermediaries have to quantify and anticipate possible uncertainties. To this end, we propose that intermediaries use the concept of TBSs [13]: Each intermediary measures the accuracy of its local demand predictions for a scheduling window $\mathcal{W}$ as a sequence of deviations from the actual local demand in the N time steps in $\mathcal{W}$. By classifying the deviations (by means of bins, i.e., intervals, of a predefined size), each intermediary obtains an individual empirical probability mass function representing the uncertainties in its local demand predictions.[2] The presumed probability that a specific sequence of deviations occurs depends on how often it was observed relative to the occurrence of other sequences. Each observed sequence represents a TBS. Together with an arbitrary root, the set of TBSs can be combined to a so-called *trust-based scenario tree* (TBST), whose transitions are annotated with conditional probabilities. Branches result from common prefixes of the sequences of deviations representing TBSs. Every time schedules are calculated, intermediaries update their discrete probability distributions on the basis of their latest experiences.

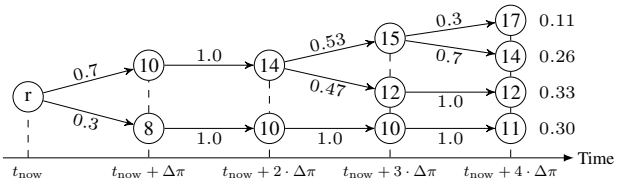

Fig. 2. An illustration of a TBST representing the demand D_λ^T an intermediary λ has to distribute among its subordinate agents $\mathcal{A}_\lambda$ in the course of the schedule creation: Each path from the root r to a leaf is a TBS that embodies a possible development of the demand λ is accountable for. Except for r, every node thus represents an expected demand (all values in MW) in a future time step in the scheduling window $\mathcal{W} = \{t_{\text{now}} + \Delta\pi, \ldots, t_{\text{now}} + 4 \cdot \Delta\pi\}$. The values at the TBST's edges indicate the conditional probabilities that the demand changes from one value to another. We define the probability $p(n)$ that a node n occurs as follows: $p(r) = 1$, $n \neq r \rightarrow p(n) = p(n \mid f(n)) \cdot p(f(n))$, where the function f returns the parent of a node n. Hence, a TBS's probability of occurrence is equal to that of the corresponding leaf. With regard to the illustration, assume that λ's assigned demand for $t_{\text{now}} + \Delta\pi$ is 9MW with an additional local demand of 1MW with probability 0.7 and -1MW with probability 0.3, resulting in a demand to distribute of 10MW and 8MW.

To achieve a stable operation of the overall system, i.e., to satisfy the overall demand as accurately as possible, each intermediary λ takes responsibility for scheduling its subordinate agents $\mathcal{A}_\lambda$ in a way that allows them to reactively and locally compensate for uncertainties in λ's local demand. For this purpose, λ uses the above-mentioned discrete probability distribution in conjunction with its assigned demand, i.e., its schedule S_λ^T, and its local demand prediction to determine multiple scenarios for the future demand it is accountable for (for details, we refer the interested reader to [14]). In combination, these demand scenarios yield the TBST D_λ^T, that is, the demand that λ has to distribute to its subordinate agents $\mathcal{A}_\lambda$. An example of such a tree is depicted in Fig. 2.

When calculating schedules, λ stipulates the contribution $S_a^T[n]$ of each subordinate agent $a \in \mathcal{A}_\lambda$ for each node $n \in D_\lambda^T \setminus \{r\}$ (the root r of S_a^T reflects a's current state and is therefore not considered). The schedule S_a^T of an agent a is thus a tree that has the same shape as D_λ^T since it captures the uncertainties of λ's local demand. This allows λ to encode

[1] "Subordinate" plants are those an AVPP is *directly* responsible for, i.e., those on its next lower level in the hierarchy.

[2] Note that intermediaries intentionally do not anticipate uncertainties resulting from the local demand of subordinate intermediaries. This is not necessary because an intermediary that cannot compensate for its local uncertainties within its subsystem is indicative of an improper system structure. Instead, a reorganization has to be triggered that re-establishes a suitable distribution of uncertainties and degrees of freedom among the subsystems.

strategies for possibly necessary contribution adjustments.

The *online multi-stage stochastic optimization problem* (cf. [16]) that is solved by each intermediary $\lambda \in \mathcal{I}$ to create schedules can be formalized in the form of a *constraint satisfaction optimization problem* (CSOP) [18] (note that we do *not* assume the domains to be finite) as follows:

$$\min_{S_a^T[n]} \quad \alpha_\Delta \cdot \mathbb{E}(\Delta) + \alpha_\Gamma \cdot \mathbb{E}(\Gamma) \tag{1}$$

$$\text{s. t.} \quad \forall a \in \mathcal{A}_\lambda, \forall n \in D_\lambda^T \setminus \{r\} : S_a^{\min} \leq S_a^T[n] \leq S_a^{\max},$$
$$\overrightarrow{S}_a^{\min}\left(S_a^T[f(n)]\right) \leq S_a^T[n] \leq \overrightarrow{S}_a^{\max}\left(S_a^T[f(n)]\right),$$

$$\text{with} \quad \mathbb{E}(\Delta) = \sum_{n \in D_\lambda^T \setminus \{r\}} p(n) \cdot |\underbrace{S_{\mathcal{A}_\lambda}^T[n]}_{\sum_{a \in \mathcal{A}_\lambda} S_a^T[n]} - D_\lambda^T[n]|,$$

$$\mathbb{E}(\Gamma) = \sum_{a \in \mathcal{A}_\lambda,\, n \in D_\lambda^T \setminus \{r\}} p(n) \cdot \kappa_a(S_a^T[n])$$

In this optimization problem, the scheduled contributions $S_a^T[n]$ for subordinate agents $a \in \mathcal{A}_\lambda$ and nodes $n \in D_\lambda^T \setminus \{r\}$ are the decision variables. An agent's present contribution is included in $S_a^T[r]$. Since scheduling is performed for the time span defined by the scheduling window $\mathcal{W}$, each node $n \in D_\lambda^T$ can be assigned to a specific time step in $\mathcal{W} \cup \{t_{\text{now}}\}$. As stated in Sect. I, the intermediaries' objective is to find an allocation that meets the demand as closely and cost-effectively as possible. Since each node $n \in D_\lambda^T$ has a probability of occurrence $p(n)$, an intermediary can achieve the primary objective by minimizing the expected total demand violation $\mathbb{E}(\Delta)$, which is defined as the expected value of the absolute deviation between the total scheduled contribution $S_{\mathcal{A}_\lambda}^T[n]$ and the demand $D_\lambda^T[n]$ over all $n \in D_\lambda^T \setminus \{r\}$ (since we sum up *absolute* violations, negative and positive deviations cannot cancel each other out).[3] Similarly, the secondary objective is achieved by minimizing the expected total costs $\mathbb{E}(\Gamma)$ that are based on agent-specific cost functions κ_a. We use the parameters α_Δ and α_Γ to establish the desired prioritization of the two objectives. We assume that the contribution of each agent $a \in \mathcal{A}$ is subject to a minimal $S_a^{\min}$ and maximal $S_a^{\max}$ contribution. The property of inertia is regarded by the functions $\overrightarrow{S}_a^{\min}$ and $\overrightarrow{S}_a^{\max}$ that further restrict a's contribution $S_a^T[n]$ for a node n depending on its contribution $S_a^T[f(n)]$ in n's parent $f(n)$.

To attain a scalable regio-central solution, each intermediary $\lambda' \in \mathcal{I} \setminus \{\Lambda\}$ uses *model abstraction* [19] to deduce a compact representation of the behavior of the represented collective, i.e., its subordinate agents $\mathcal{A}_{\lambda'}$, at runtime and transfers this model to its superior λ. As this reduces the number of decision variables and constraints λ has to consider when solving the CSOP, we obtain a complexity of $O\left(2^{|\mathcal{A}_\lambda| \cdot N}\right)$.

By taking account of D_λ^T's structure and scheduling contributions for nodes (note that multiple TBSs might share a path from the root to a node within the TBST) depending on the scheduled contributions in the corresponding parent and child nodes, our TBST-based schedules always represent feasible solutions. This means that we ensure that the transitions from one node to another comply with the regarded agent's control model. For instance, interpreting the TBST in Fig. 2 as a schedule S_a^T, the transition from r to 10MW as well as from r to 8MW is guaranteed to be feasible for a. That way, agents can defer their decision which contribution to make until the time step a branch in their schedule S_a^T has to be taken. By heading for the most promising direction, the system self-improves its ability to deal with uncertain demand. As we optimize expectations $\mathbb{E}$ on the basis of empirical discrete probability distributions, we obtain *robust solutions* [20], meaning that the system can operate efficiently and effectively in various likely situations.

[3] Technically, we calculate the expected value of the demand violation over all time steps given the probabilities of TBSs. It can be shown that our formulation based on nodes is equivalent to that, although $p(n)$ is not a probability distribution over all nodes.

B. Scheduling Reserves in Hierarchical Systems

As motivated in Sect. I (see **(R1)**, **(R2)**, **(R3)**), it is crucial to enable the agents to deviate from their scheduled contributions in order to balance supply and demand. In [15], we discussed how the subordinate agents $a \in \mathcal{A}_\lambda$ of an intermediary λ can use their schedules S_a^T as a blueprint for how to reactively adjust their contribution to compensate for differences between λ's actual and anticipated local demand. Although TBST-based schedules always contain feasible transitions, due to inertia, *spontaneous* switches (i.e., those after a branch had to be taken) from one targeted contribution to another (w.r.t. our example, from the scheduled 8MW to a new target for $t_{\text{now}} + \Delta\pi$ of, e.g., 9MW) are only possible if the agents feature an adequate amount of reserves in the form of additional degrees of freedom.

With regard to the scheduling problem, incorporating reserves, i.e., flexibility, is an additional objective that must not hinder an intermediary λ from satisfying its demand D_λ^T. Because the provision of reserves comes at a price, we propose that each intermediary λ uses its knowledge about prior reactive contribution adjustments as feedback to estimate its locally *required* negative $R_\lambda^{req,-}[n]$ and positive $R_\lambda^{req,+}[n]$ reserves that should be available between two nodes $f(n), n \in D_\lambda^T$. When creating schedules, λ now not only distributes the demand D_λ^T but also reserves to its subordinate agents $a \in \mathcal{A}_\lambda$. In the following, we describe the scheduling of reserves from the perspective of an arbitrary intermediary λ, taken from somewhere in the hierarchy. We only refer to positive reserves in our explanations. The provision of negative reserves is analogously defined. Note that the expectations $\mathbb{E}(R^{vio})$ and $\mathbb{E}(R^{vio\downarrow})$ defined below actually refer to the expected *sum* of positive and negative reserve violations.

The reserves $R_\lambda^{dis,+}[n]$ that λ has to *distribute* to $\mathcal{A}_\lambda$ are defined by the sum of the locally *required* reserves $R_\lambda^{req,+}[n]$ and λ's *assigned* reserves $R_\lambda^{ass,+}[n]$. The latter are prescribed in λ's schedule S_λ^T. When scheduling reserves, λ tries to minimize the expected violation $\mathbb{E}(R^{vio})$ of $R_\lambda^{dis,+}[n]$ (see Equations 3 and 5). λ's ability to schedule reserves in $n \in D_\lambda^T \setminus \{r\}$ hinges on the *additional* reserves $R_a^{add,+}[n]$ λ's subordinate agents $a \in \mathcal{A}_\lambda$ can promise to provide for λ in n (see Eq. 6). $R_a^{add,+}[n]$ is subject to a's *available* reserves $R_a^{avl,+}[n]$, which mainly depend on a's scheduled contribution in $f(n)$ and n, as well as the reserves $R_a^{req\downarrow,+}[n]$ required by a's subsystem

(with $R_a^{req\downarrow,+}[n] = R_a^{req,+}[n] + \sum_{\lambda' \in \mathcal{I}_a} R_{\lambda'}^{req\downarrow,+}[n]$ for intermediaries (i.e., if $a \in \mathcal{I}$), and $R_a^{req\downarrow,+}[n] = 0$ otherwise). To ensure that each subsystem represented by a subordinate intermediary $\lambda' \in \mathcal{I}_\lambda$ has the chance to schedule sufficient reserves itself, λ additionally tries to minimize the expected value $\mathbb{E}(R^{vio\downarrow})$ of the violations $R_{\lambda'}^{vio\downarrow,+}[n]$ of its subordinate subsystems' required reserves $R_{\lambda'}^{req\downarrow,+}[n]$ (see Equations 4 and 7). To preserve the subsystems' autonomy and for a local compensation for deviations, we regard the minimization of $\mathbb{E}(R^{vio\downarrow})$ as more important than the minimization of $\mathbb{E}(R^{vio})$. Again, we use the parameters $\alpha_{R^{vio\downarrow}}$ and $\alpha_{R^{vio}}$ to obtain this prioritization. While the satisfaction of the demand remains paramount, we degrade the minimization of the costs to reflect our risk-averse attitude. Hence, we extend the *objective function* of the scheduling problem formalized in Eq. 1 as follows:

$$\min_{\substack{S_a^T[n],\\ R_a^{ass,+}[n]}} \alpha_\Delta \cdot \mathbb{E}(\Delta) + \alpha_\Gamma \cdot \mathbb{E}(\Gamma) + \alpha_{R^{vio}} \cdot \mathbb{E}(R^{vio}) + \alpha_{R^{vio\downarrow}} \cdot \mathbb{E}(R^{vio\downarrow}) \tag{2}$$

$$\text{with } \mathbb{E}(R^{vio}) = \sum_{n \in D_\lambda^T \setminus \{r\}} p(n) \cdot \Big(R_\lambda^{vio,+}[n] + R_\lambda^{vio,-}[n] \Big), \tag{3}$$

$$\mathbb{E}(R^{vio\downarrow}) = \sum_{\lambda' \in \mathcal{I}_\lambda,\, n \in D_\lambda^T \setminus \{r\}} p(n) \cdot (R_{\lambda'}^{vio\downarrow,+}[n] + R_{\lambda'}^{vio\downarrow,-}[n]) \tag{4}$$

$$\text{and } R_\lambda^{vio,+}[n] = \max\Big\{0, R_\lambda^{dis,+}[n] - \sum_{a \in \mathcal{A}_\lambda} R_a^{add,+}[n]\Big\}, \tag{5}$$

$$R_a^{add,+}[n] = \max\big\{0, R_a^{avl,+}[n] - R_a^{req\downarrow,+}[n]\big\} \tag{6}$$

$$R_{\lambda'}^{vio\downarrow,+}[n] = \Big|\min\Big\{0, R_{\lambda'}^{avl,+}[n] - R_{\lambda'}^{req\downarrow,+}[n]\Big\}\Big| \tag{7}$$

(analogously for $R_\lambda^{vio,-}[n], R_a^{add,-}[n], R_{\lambda'}^{vio\downarrow,-}[n]$)

In Fig. 3, we illustrate the reserves $R_a^{avl,+}[n]$ an agent $a \in \mathcal{A}$ can provide between two nodes $f(n), n \in D_\lambda^T$. Our calculations assume that reserves can be mobilized from one time step to another, i.e., from t to $t + \Delta t$. Finally, an intermediary λ prescribes the assigned reserves $R_a^{ass,+}[n]$ for each agent $a \in \mathcal{A}_\lambda$ proportionally to $R_a^{add,+}[n]$ and inversely proportional to $R_{total}^{add,+}[n] := \sum_{a' \in \mathcal{A}_\lambda} R_{a'}^{add,+}[n]$ (with $R_\Lambda^{ass,+}[n] := 0$):

$$R_a^{ass,+}[n] = \begin{cases} 0 \text{ if } R_{total}^{add,+}[n] = 0 \\ R_a^{add,+}[n] \cdot \min\Big\{1, \frac{R_\lambda^{dis,+}[n]}{R_{total}^{add,+}[n]}\Big\} \text{ otherwise} \end{cases} \tag{8}$$

The evaluation of alternative strategies for assigning reserves that could take additional optimization criteria into account, such as the agents' costs, are subject to future work.

Because the optimized provision of reserves exacerbates the search for high-quality solutions (see Sect. V), we present an auction-based heuristic for the robust solution of the RAP in hierarchical systems that does not require intermediaries to minimize $\mathbb{E}(R^{vio\downarrow})$ in the following section.

IV. Robust and Cooperative Resource Allocation

In this section, we propose TruCAOS$^{\text{R}}$, an auction-based algorithm that solves the scheduling problem in a cooperative and regionalized manner. As the provision and consumption of resources, such as electricity, are already subject to rewards and costs in real systems, a market-based approach is a natural choice. TruCAOS$^{\text{R}}$ is based on TruCAOS^{+} [7]. While TruCAOS^{+} primarily focuses on how to deal with an uncertain provision of scheduled resources, i.e., contributions, we concentrate on *uncertain demand* in this paper. Hence, TruCAOS$^{\text{R}}$ substantially differs from TruCAOS^{+} in its ability to obtain *robust solutions* by **(1)** creating schedules on the basis of TBSTs for the demand and **(2)** optimizing the provision of required reserves in the form of additional degrees of freedom. Sect. IV-A gives an overview of TruCAOS$^{\text{R}}$'s basic procedure, before Sect. IV-B highlights how it obtains robust solutions.

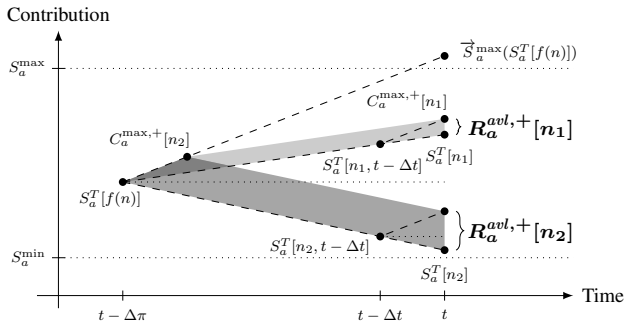

Fig. 3. Two illustrations of the *positive reserves* $R_a^{avl,+}[n]$ an agent $a \in \mathcal{A}$ can provide between two nodes $f(n)$ and $n \in \{n_1, n_2\}$ (indicated by the gray areas): The amount of $R_a^{avl,+}[n]$ is subject to a's scheduled contributions $S_a^T[f(n)]$ and $S_a^T[n]$ as well as its control model. The latter captures a's inertia represented by the function $\overrightarrow{S}_a^{\max}$ as well as its minimal $S_a^{\min}$ and maximal $S_a^{\max}$ contributions, among others. With regard to its scheduled contributions $S_a^T[f(n)]$ and $S_a^T[n]$ (we assume a linear change of a's contribution between $f(n)$ and n), $R_a^{avl,+}[n]$ denotes, e.g., the *minimum* (in case of a *pessimistic* approach) of the maximal additional contributions that a can mobilize within the single time steps between $f(n)$ and n. In this context, $C_a^{\max,+}[n]$ is the maximal contribution a can provide between $f(n)$ and n if it increases its contribution as much as possible from one time step to another. Clearly, $C_a^{\max,+}[n]$ is bounded by $S_a^{\max}$. If $S_a^T[f(n)] > S_a^T[n]$ (case $n = n_2$ in our illustration), a can provide more reserves than in case of $S_a^T[f(n)] < S_a^T[n]$ (case $n = n_1$ in our illustration) because we also include reserves that result from not decreasing a contribution from one time step to another.

A. Basic Procedure

Similarly to the general explanation in Sect. III, TruCAOS$^{\text{R}}$ determines schedules in a top-down manner: The schedule creation is triggered by the top-level intermediary Λ and subordinate intermediaries $\lambda \in \mathcal{I} \setminus \{\Lambda\}$ are responsible for recursively distributing their fraction D_λ^T of the overall demand. In TruCAOS$^{\text{R}}$, this is achieved by enabling all agents $a \in \mathcal{A}_\lambda$ to sell or buy resources to or from their superordinate intermediary λ according to the demand D_λ^T that λ has to distribute. This is done in an iterative and incremental process that, in its basic form, is reminiscent of an iteratively performed first-price sealed-bid auction (see, e.g., [21]). As before, D_λ^T captures the uncertainties of λ's local demand and is derived from λ's schedule S_λ^T, that is, the result of taking part in the auction of its superordinate intermediary.

In all bidding iterations, an intermediary λ has the goal to increase the expected satisfaction of D_λ^T by announcing an auction in which the agents $a \in \mathcal{A}_\lambda$ can bid for a part of the *remaining demand* D_λ^{rem} that λ has to distribute (note that D_λ^{rem} might also be negative in some nodes). The corresponding *call for proposals* (CFP) contains a fraction $g \in [0, 1]$ of

D_λ^{rem} and is sent to all $a \in \mathcal{A}_\lambda$. If $g < 1$, the demand is distributed among the best bidders in the course of multiple bidding iterations, resulting in a fairer allocation of resources.

Each agent $a \in \mathcal{A}_\lambda$ that receives the CFP and wants to sell or buy resources to or from λ responds with a *proposal* S_a^{pro}, i.e., a *proposed schedule*, that includes a promised contribution as well as the costs, i.e., a remuneration, for providing the contribution. Note that a proposer could also offer money to its intermediary, e.g., if it wants to lower its contribution or buy resources. Having gathered the proposals of all bidders in a set $\mathcal{P}$, λ completes this bidding iteration by identifying and accepting one or more suitable winner proposals $\mathcal{P}_w \subseteq \mathcal{P}$ as explained in Sect. IV-B. All other proposals are rejected. In the possibly following bidding iteration, all proposers – including the "winners" – are allowed to send new proposals.

The *most recently* accepted proposal of an agent a defines its schedule S_a^T, which is a contract between a and its intermediary for all time steps in the regarded scheduling window $\mathcal{W}$: a has to comply with its schedule in exchange for payment. For this reason, whenever a wants to change its scheduled contribution, it has to make a proposal S_a^{pro} that, if accepted, replaces the previously accepted schedule. In TruCAOS$^{\mathrm{R}}$, a schedule S_a^T does not become invalid if a new schedule creation is started. Instead, it is refined in the course of successive schedule creations. As a result, intermediaries, acting as auctioneers, as well as proposers can rely on their negotiated contracts. When initializing D_λ^{rem}, λ therefore subtracts the existing schedules of all subordinate agents $a \in \mathcal{A}_\lambda$ from the demand D_λ^T that has to be distributed. However, due to dynamic uncertainties in λ's local demand, the shape of D_λ^T can change from one schedule creation to another and thus differ from the shape of the existing schedules of λ's subordinate agents. For such situations, we use a greedy algorithm to find an adequate mapping of the scenarios of existing schedules to those in D_λ^T. Such a mapping is determined by each intermediary before the first bidding iteration. For nodes that were not regarded in the previous schedule creation (i.e., those outside the previous scheduling window), auctioneers and proposers assume a neutral contribution of zero. Because the resulting schedule might contradict an agent's control model, there is a dedicated phase in which the agents can correct their schedules. As described in [7], minimally inverse corrections are incentivized. This ensures that the agents do not misuse this feature for their own sake, which might impair the quality of the solution.

Each agent $a \in \mathcal{A}$ creates its proposals S_a^{pro} by solving a CSOP that is very similar to the one specified in Equations 1 and 2: Proposals depend on a's constraint model, its current state, and the demand specified in the CFP. However, since each agent now only has to determine the scheduled contribution for itself, TruCAOS$^{\mathrm{R}}$ mitigates the scalability issues discussed in Sections I and III-A. Instead of $O\left(2^{|\mathcal{A}_\lambda| \cdot N}\right)$, the complexity of solving the RAP for a subsystem $\mathcal{A}_\lambda$ is only $O\left(j_{\mathrm{total}} \cdot \left(|\mathcal{A}_\lambda| \cdot 2^N + 2^{|\mathcal{A}_\lambda|}\right)\right)$, where $|\mathcal{A}_\lambda| = |\mathcal{P}|$, j_{total} is the total number of iterations needed to solve the RAP, and $O\left(2^{|\mathcal{A}_\lambda|}\right)$ the complexity of selecting the winner proposals $\mathcal{P}_w$ from $\mathcal{P}$. Note that we still benefit from a hierarchical approach due to the complexity of proposal selection. Because intermediaries submit bids before their subordinates, they have to be aware of the capabilities of their collective to avoid infeasible schedules that would have to be corrected. As explained in Sect. III-A, model abstraction ensures that the above-mentioned complexity holds for the entire hierarchy. Because intermediaries do not impose schedules on subordinates in TruCAOS$^{\mathrm{R}}$, they can even use these abstract models to create their own proposals in $O\left(2^N\right)$.

The iterative process of distributing the remaining demand D_λ^{rem} terminates either if it is sufficiently satisfied, i.e., if its absolute values are below a predefined threshold $D_{\max}^{rem}$, or if λ did not receive any proposals (i.e., $\mathcal{P} = \emptyset$), or if there is no proposal that increases the satisfaction of D_λ^T, or if a maximum number of bidding iterations $j_{\max}$ is exceeded. The information about termination originates from the top-level intermediary and propagates downwards in the hierarchy.

B. Obtaining Robust Solutions

To identify one or more suitable winner proposals $\mathcal{P}_w \subseteq \mathcal{P}$, an intermediary λ could simply sort the proposals in $\mathcal{P}$ in descending order of their *price-performance ratio* – which we define as the ratio between the expected total contribution and the expected remuneration – and declare that the proposal with the best ratio that improves the remaining demand's satisfaction wins the auction. While this *merit order* approach leads to low-priced allocations, it is computationally inefficient because it clearly needs a large number of bidding iterations to satisfy the demand and does not allow to take advantage of synergy effects that can result from accepting a combination of proposals. The latter is especially beneficial in the context of multi-objective optimization. We therefore decided for a compromise between the number of bidding iterations and resulting monetary costs: To ensure that acceptable proposals and thus the overall result feature a sufficient quality, intermediaries filter, i.e., reject, proposals with a price-performance ratio worse than a historical average, calculated as a moving average over the last k_{ppr} schedule creations. While this procedure does not yield minimal costs, it ensures that the average costs do not increase. If no such proposal exists, only the proposals with the best price-performance ratio pass the filtering stage. As unsuitable proposals are rejected, the complexity of the combinatorial optimization problem that is solved to determine $\mathcal{P}_w$ is reduced.

An intermediary λ chooses $\mathcal{P}_w$ in such a way that there is no other combination that yields a greater expected gain in satisfaction of the remaining demand. In this context, λ's secondary and tertiary goals are to minimize the expected violation $\mathbb{E}(R^{vio})$ of the reserves $R_\lambda^{dis,+}[n] = R_\lambda^{req,+}[n] + R_\lambda^{ass,+}[n]$ it has to distribute in all $n \in D_\lambda^T$ (the same holds for negative reserves $R_\lambda^{dis-}[n]$) and to minimize the expected costs, respectively. Note that these objectives as well as their lexicographical order correspond to those defined in Eq. 2. The only but important difference is that λ does not have to care about the satisfaction of the required reserves on subordinate system levels by minimizing $\mathbb{E}(R^{vio\downarrow})$ because each subordinate $a \in \mathcal{A}_\lambda$ determines suitable contributions

and additional reserves $R_a^{add,+}[n]$ it can provide for λ itself. As λ is merely informed about $R_a^{add,+}[n]$ in the agents' proposals, computational costs are reduced. Analogously to Eq. 8, TruCAOS$^{\mathrm{R}}$ assigns the reserves $R_a^{ass,+}[n]$ proportionally to $R_a^{add,+}[n]$ and inversely proportional to $R_{total}^{add,+}[n]$.

While, in general, each agent $a \in \mathcal{A}$ could incorporate individual and variable objectives when generating proposals, we assume that a generates its proposals as follows in order to remain competitive: According to the intermediaries' primary objective, a creates proposals that maximize the expected satisfaction of the demand announced in the CFP. Maximizing the expected additional reserves $R_a^{add,+}[n]$ that can be provided to its intermediary λ is secondary. In case a is an intermediary, suggesting $R_a^{add,+}[n]$ makes sure that a has sufficient flexibility to satisfy the reserves $R_a^{req\downarrow,+}[n]$ required by its subsystem $\mathcal{A}_a$ when it is its turn to calculate schedules. This procedure leads to a top-down propagation of assigned reserves. As in Eq. 1, we assume that agents stipulate their remunerations according to their cost functions κ_a.

V. Evaluation

The goal of our evaluation is to investigate if TruCAOS$^{\mathrm{R}}$ is more suitable for obtaining robust solutions than the regio-central approach explained in Sect. III (hereinafter called *RegioC*). For this purpose, we evaluate TruCAOS$^{\mathrm{R}}$ as well as RegioC in three different modes in the context of our case study: In the first mode, called "E-nu", demand predictions are not subject to uncertainties. In the second and third mode, called "E-nr" and "E-r", uncertainties are present but reserves are only scheduled in E-r. All modes were performed in combination with four different hierarchies of AVPPs, amounting to 12 evaluation scenarios. For each evaluation scenario, we performed 100 runs.

A. Test Bed

We base our evaluation on a hierarchical structure of AVPPs, consisting of 173 dispatchable and 350 non-dispatchable power plants of different types (hydro, biofuel, gas power plants as well as solar plants and wind generators). While each power plant is modeled as an individual agent, we use a single agent to represent all non-dispatchable consumers that is assigned to Λ's local environment. The demand, i.e., residual load, originates from weather-dependent power plants and the consumers. For physical power plant data (such as production boundaries), load curves, and weather data influencing the output of weather-dependent power plants, we use real world data.[4] Each dispatchable power plant's inertia is defined within typical boundaries. The production costs, i.e., the average costs of providing a contribution, range from $6.50\frac{\text{EUR cent}}{\text{kWh}}$ to $17.50\frac{\text{EUR cent}}{\text{kWh}}$. The evaluation is implemented in a sequential, round-based execution model in which each round corresponds to a specific time step $t \in \mathcal{T}$. Power plants have to satisfy a prescribed residual load over a period of half a day, corresponding to $|\mathcal{T}| = 240$ discrete time steps, each representing $\Delta t = 3$min. Every 15min, AVPPs update their TBSs and create schedules for the next hour with a resolution of $\Delta\pi = 15$min (i.e., $N = 4$) on the basis of residual load predictions. Uncertainty in these predictions is generated by means of Markov models that randomly modify the actual behavior of non-dispatchable power plants and the consumers. To reflect time-dependent behavior, the prediction error of a specific time step depends on prediction errors of previous time steps. The accuracy of predictions of non-dispatchable power plants within the same AVPP is coupled as is the case with weather predictions for different locations. Given that schedules are recalculated every 15min, required reserves only have to be scheduled for the next 15min, i.e., for all TBST nodes $n \in D_\lambda^T$ that map to a time step $t \in \mathcal{W}$ with $t - t_{\text{now}} = 15$min. Each AVPP $\lambda \in \mathcal{I}$ uses the maximum of its negative and positive reactive contribution adjustments within the last 30min as required negative $R_\lambda^{req,-}[n]$ and positive $R_\lambda^{req,+}[n]$ reserves.

The optimization problems that have to be solved to generate proposals and determine winner proposals in TruCAOS$^{\mathrm{R}}$ as well as those of RegioC are formulated as mixed integer linear programs and solved by IBM ILOG CPLEX[5]. Regarding Eq. 2, the parameters α_Δ, α_Γ, $\alpha_{R^{vio}\downarrow}$, and $\alpha_{R^{vio}}$ are fixed in a way that establishes the desired prioritization among the different objectives. For TruCAOS$^{\mathrm{R}}$, we initialize the parameters introduced in Sect. IV as follows: $k_{\text{ppr}} = 5$, $D_{\max}^{rem} = 5$kW, $g = 0.2$ if $D_\lambda^{rem} \cdot g > 1000$kW, else $g = 1$. To keep the runtime of RegioC within reasonable bounds, AVPPs stop the schedule creation after 15s if a valid solution is present. If not, they give CPLEX another 10min. If this threshold is exceeded, no solution is found and the run is aborted.

We examine the following three questions of interest: **(Q1)** *Does TruCAOS^{R} outperform RegioC in terms of runtime?* **(Q2)** *Does TruCAOS^{R} obtain better results w.r.t. expected demand satisfaction, provision of required reserves, and costs?* **(Q3)** *How does the hierarchical system structure affect quality and runtime?* To investigate the impact of hierarchical system structures on TruCAOS$^{\mathrm{R}}$'s and RegioC's performance, we performed our evaluations on four different structures called "superflat", "flat", "deep", and "deeper" of height 1, 2, 3, and 5, and with an average number of 173, 14.23, 6.44, and 2.42 dispatchable power plants per AVPP, respectively. All these structures were generated offline on the basis of the same set of power plants. In our experiments, we disabled the system's self-organization to be able to analyze the influence of these predefined structures on the algorithms' performance.

B. Results

Table I shows the results for modes E-nr and E-r in combination with the four different system structures. First of all, we observe that, with regard to a specific combination of system structure and mode, TruCAOS$^{\mathrm{R}}$ and RegioC had to deal with quite the same degree of uncertainties in E-nr and E-r, mirrored in the number of TBSs in each schedule

[4] See EnergyMap (2012, http://www.energymap.info/), LEW (2012, http://www.lew-verteilnetz.de/), and LfL (2010, http://www.lfl.bayern.de/).

[5] IBM ILOG CPLEX Optimizer, Version 12.4, 2011: http://www-01.ibm.com/software/commerce/optimization/cplex-optimizer/

	RegioC								TruCAOS^{R}							
	superflat		flat		deep		deeper		superflat		flat		deep		deeper	
	E-nr	E-r	E-nr	E-r	E-nr	E-r	E-nr	E-r	E-nr	E-r	E-nr	E-r	E-nr	E-r	E-nr	E-r
#Runs Exceeding the Max. Scheduling Time	39	68	0	20	0	7	0	5	—	—	—	—	—	—	—	—
#TBSs per Schedule	1.62 (0.96)	1.65 (0.97)	1.98 (1.30)	1.97 (1.29)	1.95 (1.32)	1.97 (1.34)	1.83 (1.32)	1.85 (1.34)	1.75 (1.15)	1.76 (1.18)	1.95 (1.29)	1.96 (1.31)	1.92 (1.30)	1.96 (1.36)	1.83 (1.32)	1.82 (1.31)
Max. Seq. Scheduling Time [s]	46.37 (77.51)	126.38 (129.21)	0.43 (0.34)	111.80 (213.37)	0.22 (0.10)	27.94 (59.84)	0.20 (0.07)	16.68 (20.15)	1.10 (0.79)	1.45 (1.09)	1.23 (0.85)	1.64 (0.91)	1.87 (0.55)	2.66 (0.78)	2.98 (0.65)	4.61 (1.18)
Scheduling Time per AVPP [s]	46.37 (77.51)	126.38 (129.21)	0.14 (0.17)	10.20 (64.49)	0.06 (0.04)	1.51 (11.37)	0.03 (0.02)	0.25 (2.11)	1.10 (0.79)	1.45 (1.09)	0.34 (0.31)	0.42 (0.38)	0.28 (0.20)	0.36 (0.30)	0.15 (0.16)	0.20 (0.25)
Exp. Violation of the Ass. Residual Load per Schedule Creation [kW]	10.92 (127.28)	60.04 (1464.10)	457.95 (1066.28)	1131.58 (11309.59)	8212.31 (5100.85)	13250.51 (13672.30)	39572.63 (13010.64)	52904.41 (20112.41)	0.54 (0.98)	0.45 (0.93)	156.77 (477.06)	164.28 (516.05)	1123.09 (1637.33)	1089.03 (1704.78)	13756.14 (9831.97)	12526.38 (7893.77)
Exp. Costs of Schedule [1000 EUR]	150.67 (38.45)	149.69 (38.09)	163.44 (33.83)	161.67 (34.04)	165.50 (32.45)	166.92 (31.95)	177.16 (28.80)	175.29 (29.36)	153.85 (38.69)	153.92 (38.54)	167.47 (32.31)	167.06 (32.39)	162.66 (35.98)	163.38 (36.03)	177.41 (26.33)	177.73 (26.09)
Avg. Exp. Violation of Req. Reserves over Schedule Creations in Run [kW]	41679.20 (2101.53)	9.93 (24.63)	64017.60 (3298.07)	15805.11 (2417.76)	66985.54 (3371.81)	27095.42 (2667.58)	73376.43 (3929.13)	42192.57 (3734.92)	41556.09 (2595.41)	1.50 (14.96)	74053.92 (4961.36)	8678.90 (1637.53)	70029.36 (2957.42)	7886.57 (1352.67)	87332.14 (9571.86)	20106.47 (2686.00)

TABLE I

EVALUATION RESULTS OBTAINED BY TRUCAOS^{R} AND REGIOC FOR MODES E-NR AND E-R AS WELL AS DIFFERENT HIERARCHIES. "#RUNS EXCEEDING THE MAX. SCHEDULING TIME" DENOTES THE NUMBER OF REGIOC RUNS THAT WERE ABORTED DUE TO AN EXCEEDED MAXIMAL SCHEDULING TIME OF 10MIN. ALL OTHER RESULTS ARE AVERAGED OVER THE 100 RUNS PER EVALUATION SCENARIO. VALUES IN PARENTHESES DENOTE STANDARD DEVIATIONS.

creation. Variations between different system structures result from different local environments. Compared to E-nu – the mode without uncertainties –, E-r's average maximal sequential runtime (i.e., the maximum of the sums of the scheduling times in each branch originating from the root of the AVPP structure) over all structures increased from 0.87s ($\sigma = 0.52$s) to 2.59s ($\sigma = 1.26$s) in case of TruCAOS^{R} and from 1.51s ($\sigma = 2.39$s) to 70.70s ($\sigma = 48.83$s) in case of RegioC. This demonstrates that robustness comes at the price of longer runtimes.

With regard to RegioC and E-r, we notice that RegioC's average max. seq. runtime significantly declines with the height of the hierarchy; from 126.38s to 16.68s. This comes along with a decrease in the average scheduling time per AVPP from 126.38s to 0.25s as well as a remarkably drop in the number of runs without a solution, i.e., those that had to be aborted by RegioC due to an exceeded maximal scheduling time of 10min. While this highlights the benefit of problem decomposition, it also shows that the system must be able to find suitable structures at runtime: The expected violation of the assigned residual load increases with the height of the hierarchy from 60.04kW to 52.90MW (the mean residual load was about 1.5GW). The same applies to the costs of a schedule which rise by about 25k EUR from "superflat" to "deeper" (we imposed a penalty of $17.50\frac{\text{EUR cent}}{\text{kWh}}$ for violated assigned residual loads), and the violation of required reserves. All this can be attributed to a higher degree of the system's fragmentation and errors introduced by model abstraction [19]. While "deeper" has the least number of aborted runs, "deep" yields an acceptable tradeoff between quality, runtime, and aborted runs for E-r. The same observations can be made for E-nr, where "flat" yields an acceptable tradeoff. Due to the optimization of the provision of reserves in E-r, RegioC's violation of reserves is considerably smaller in E-r than in E-nr. However, compared to E-nr, E-r's number of aborted runs is significantly higher. Moreover, E-nr obtains a shorter average max. seq. runtime, which is between 46.37s and 0.20s, and a lower average expected violation of the assigned residual load, ranging between 10.92kW and 39.57MW.

With regard to TruCAOS^{R}, we also notice that problem decomposition leads to a decline in the average scheduling time per AVPP; from 1.10s to 0.15s in case of E-nr and from 1.45s to 0.20s in case of E-r. In contrast to RegioC, TruCAOS^{R}'s max. seq. scheduling time increases with the height of the hierarchy, though. We explain this behavior by the overhead introduced by additional hierarchy levels: First, due to the complexity of the AVPPs' control models, AVPPs need more time for generating proposals than physical power plants. Second, the computationally expensive selection of winner proposals, which scales exponentially with the size of AVPPs (see Sect. IV-A), has to be accomplished top-down and thus sequentially. So there is definitely a tradeoff between the size of AVPPs and the height of the hierarchy. In spite of this behavior, TruCAOS^{R} does not only achieve much lower max. seq. runtimes for E-r (1.45s for "superflat") than RegioC (27.94s for "deep"), but it also obtains better solutions in terms of the violation of the assigned residual load. While fragmentation also leads to a growth of the expected violation of the assigned residual load with the hierarchy's height, it only increases from 0.54kW to 13.76MW in case of E-nr and from 0.45kW to 12.53MW in case of E-r. In particular, the values for "superflat" and "flat" are remarkably low. Compared to RegioC, TruCAOS^{R}'s accuracy in case of E-nr comes at the expense of higher max. seq. runtimes though: 1.10s for "superflat" and TruCAOS^{R} compared to 0.43s in case of "flat" and RegioC. Further, TruCAOS^{R} is able to satisfy the required reserves in E-r's "superflat" runs very accurately; also note that TruCAOS^{R} completed all runs as opposed to RegioC. Compared to RegioC, the average violation of reserves of 7.89MW in case of "deep" is also very low. Given that RegioC's average expected costs range between approximately 150k EUR and 177k EUR, TruCAOS^{R} achieves very promising results of about 154k EUR for "superflat". The benefit of TruCAOS^{R}'s price-performance filtering discussed in Sect. IV-B becomes evident when disabling its functionality: Then, the costs rise to 180k EUR in case of "superflat" and E-r.

To sum up, TruCAOS^{R} significantly outperforms RegioC in terms of solution quality and runtime when creating schedules in uncertain environments. While we observed that hierarchical organizations support the timely calculation of robust solutions, suitable structures have to be formed at runtime to achieve a suitable tradeoff between quality and runtime.

VI. RELATED WORK

In the body of literature, there are several approaches that solve problems similar to the RAP presented in this paper.

However, to the best of our knowledge, our approach is unique in the way we combine online stochastic optimization, self-organized hierarchical problem decomposition, and a market-based mechanism to achieve a robust and scalable solution.

In the field of operations research and in the context of RAPs, online stochastic optimization [16] is often used to cope with uncertainties of different sources. Most of such scenario-based solutions (cf. [22], [23]) formalize the problem as a two-stage decision problem due to the computational complexity of solving multi-stage decision problems with a centralized optimizer [24]. Complexity is reduced in two ways: (1) Two-stage approaches either assume that there is no uncertainty in the the first stage (cf. [23]), that is, the next time step schedules are created for, or capture uncertainties in a single expected value which is often only weakly informative (cf. [22]). (2) Although the second stage considers uncertainties in the form of scenarios for the remaining time steps in the scheduling window, so-called non-anticipativity constraints that would treat the scenarios as a tree are not regarded. Apart from that, [22] further does not take the scenarios' probability into account. In contrast to these centralized solutions, our approach uses self-organization and a hierarchical market-based mechanism to deal with the complexity of multi-stage decision problems. As we have shown in [14], [15], scenario-tree-based schedules are an enabler for efficient self-adaption in order to achieve a stable operation of the overall system. While [23] also schedules reserves to increase the system's robustness, they are considered as hard constraints instead of an objective.

[25] presents a centralized scheduling heuristic for solving a combinatorial RAP under uncertain energy demand and renewable supply. While [25] improves robustness by maximizing the expected outcome, scenarios are not learned at runtime but sampled from a fixed probability distribution. Similar to TruCAOS$^{\mathrm{R}}$'s price-performance filtering, a greedy algorithm creates schedules according to the price-performance ratio of proposed contributions. In addition to the *unintentional* uncertainties we focus on in this paper, [25] addresses *intentional* uncertainties on the basis of the principle of incentive-compatible mechanism design [26]. In the context of supply and demand management in smart grids, various market-based approaches use mechanism design to prevent uncertainties arising from strategic misbehavior and gambling through pricing mechanisms (e.g., [27], [28]). To reactively deal with unintentional prediction errors, [28] proposes an additional online balancing mechanism that compensates for deviations between demand and contributions. The mechanism presented in [27] mitigates unintentional uncertainties by forming coalitions of agents such that prediction errors cancel each other out. Our self-organized formation of AVPPs mentioned in Sect. II has a similar goal, but additionally balances degrees of freedom to promote the ability of reactive contribution adjustments. In [29], Dash et al. introduce a payment function that reduces the risk that bidders overestimate their contribution if it could unintentionally turn out to be lower than expected. Underestimations that also lead to imbalances are not penalized.

Other market-based approaches that solve problems similar to our RAP are presented in [30], [31], [32]. DEZENT [30] is a bottom-up market-based mechanism in which agents balance energy supply and demand by negotiating and concluding contracts within fixed price frames that are tightened from one iteration to another. A further approach based on a hierarchical system structure is PowerMatcher [31] in which the root balances supply and demand by determining an equilibrium price, based on aggregated demand, supply, and price predictions, to establish a market equilibrium. The auctioneer is thus a central component of the system. Unlike TruCAOS$^{\mathrm{R}}$, DEZENT and PowerMatcher only create schedules for the next instead of multiple future time steps. Stigspace [32] is a coordination mechanism that uses a blackboard, called stigspace, as the medium of communication between distributed energy resources in order to create schedules in an iterative process. Initially, the stigspace is used to announce the demand that has to be fulfilled by the distributed energy resources. These in turn revise their schedule in order to minimize the remaining demand. The new schedule is then posted to the stigspace where the remaining demand is updated accordingly. This process is repeated until the demand is sufficiently satisfied. Because every schedule posted to the stigspace is "accepted", only the RAP's primary objective – to allocate resources according to a given demand – might be achieved. In contrast to TruCAOS$^{\mathrm{R}}$, the above-mentioned approaches do not deal with uncertainties.

In [8], Hinrichs et al. present a decentralized heuristic, called COHDA$_2$, for solving a combinatorial optimization problem. The algorithm is based on a neighborhood structure in which agents iteratively adjust their configuration, i.e., state, in order to achieve a global as well as individual objectives. The former can be thought of as the objectives of our RAP. In COHDA$_2$, every time an agent changes its configuration, it informs its neighbors about its new configuration as well as the currently known configuration of its neighbors and the neighbors' neighbors etc. it received in a prior time step. Although each agent only has current information about its direct neighborhood, it synthesizes a complete representation of the configuration of all agents in the system over time. To decide on suitable own configurations, the agents take this locally perceived global configuration into account. While COHDA$_2$ solves a combinatorial optimization problem in a truly decentralized manner, we use a market-based approach on the basis of a self-organizing hierarchical system structure. Further, COHDA$_2$ does not consider uncertainties.

VII. Conclusion

In this paper, we presented an approach for robust resource allocation in large-scale systems in the light of uncertain demand and agents exhibiting inert behavior. This requires the RAP to be solved proactively by creating coarse-grained schedules on the basis of demand predictions. To allow the agents to anticipate uncertainties and, at the same time, be able to reactively self-adapt their contributions at runtime, we proposed to combine techniques from online stochastic optimization with the provision of feedback-driven reserves. Scenario-based schedules based on empirical discrete probability distributions of prediction errors allow the agents to cope

with different possible developments of the demand. To deal with systems of large size, we formalized the problem for self-organizing hierarchical system structures. Further, we outlined a market-based solution, called TruCAOS^{R}, that solves the RAP in a robust manner. Based on a case study from decentralized autonomous power management, we empirically demonstrated that TruCAOS^{R} clearly outperforms a regio-central approach that utilizes a state-of-the-art optimizer.

In future work, we will investigate the interaction of schedule creation and the self-organizing hierarchy in more detail. As for TruCAOS^{R}, we want to enable heterogeneous agents, e.g., peaking and base load power plants, to follow individual preferences and learn strategies for making proposals that correspond to their type, thereby increasing the overall system's performance. Moreover, we will examine TruCAOS^{R}'s ability to simultaneously deal with uncertain demand and contributions.

ACKNOWLEDGMENT

This research is partly sponsored by the research unit *OC-Trust* (FOR 1085) of the German Research Foundation.

REFERENCES

[1] UCTE, "UCTE Operation Handbook – Policy 1: Load-Frequency Control and Performance," Union for the Co-ordination of Transmission of Electricity, Tech. Rep. UCTE OH P1, 2009.

[2] T. Van Zandt, "Hierarchical Computation of the Resource Allocation Problem," *European Economic Review*, vol. 39, no. 3-4, pp. 700–708, April 1995.

[3] L. Rani, M. Mam, and S. Kumar, "Economic Load Dispatch In Thermal Power Plant Taking Real Time Efficiency As An Additional Constraints," *International Journal of Engineering Research & Technology (IJERT)*, vol. 2, no. 7, 2013.

[4] N. Padhy, "Unit Commitment – A Bibliographical Survey," *IEEE Transactions on Power Systems*, vol. 19, no. 2, pp. 1196–1205, 2004.

[5] J. S. Heo, K. Y. Lee, and R. Garduno-Ramirez, "Multiobjective Control of Power Plants Using Particle Swarm Optimization Techniques," *IEEE Transactions on Energy Conversion*, vol. 21, no. 2, pp. 552–561, 2006.

[6] A. Zafra-Cabeza, M. A. Ridao, I. Alvarado, and E. F. Camacho, "Applying Risk Management to Combined Heat and Power Plants," *IEEE Transactions on Power Systems*, vol. 23, no. 3, pp. 938–945, 2008.

[7] G. Anders, A. Schiendorfer, F. Siefert, J.-P. Steghöfer, and W. Reif, "Cooperative Resource Allocation in Open Systems of Systems," *ACM Transactions on Autonomous and Adaptive Systems*, vol. 10, no. 2:11, May 2015.

[8] C. Hinrichs, M. Sonnenschein, and S. Lehnhoff, "Evaluation of a Self-Organizing Heuristic for Interdependent Distributed Search Spaces," in *International Conference on Agents and Artificial Intelligence (ICAART'13)*, J. Filipe and A. L. N. Fred, Eds., vol. Volume 1 – Agents. SciTePress, 2013, pp. 25–34.

[9] A. Bar-Noy, R. Bar-Yehuda, A. Freund, J. Naor, and B. Schieber, "A Unified Approach to Approximating Resource Allocation and Scheduling," *Journal of the ACM (JACM)*, vol. 48, no. 5, pp. 1069–1090, 2001.

[10] M. Abouelela and M. El-Darieby, "Multidomain Hierarchical Resource Allocation for Grid Applications," *Journal of Electrical and Computer Engineering*, vol. 2012, Jan. 2012.

[11] A. Boudjadar, A. David, J. H. Kim, K. G. Larsen, M. Mikucionis, U. Nyman, and A. Skou, "Hierarchical Scheduling Framework Based on Compositional Analysis Using Uppaal," in *Proceedings of FACS 2013*, ser. Lecture Notes in Computer Science. Springer, 2013.

[12] J.-P. Steghöfer, P. Behrmann, G. Anders, F. Siefert, and W. Reif, "HiSPADA: Self-Organising Hierarchies for Large-Scale Multi-Agent Systems," in *Proceedings of the Ninth International Conference on Autonomic and Autonomous Systems (ICAS'13)*. IARIA, 2013.

[13] G. Anders, F. Siefert, J.-P. Steghöfer, and W. Reif, "Trust-Based Scenarios – Predicting Future Agent Behavior in Open Self-organizing Systems," in *Self-Organizing Systems*, ser. Lecture Notes in Computer Science, W. Elmenreich, F. Dressler, and V. Loreto, Eds. Springer Berlin Heidelberg, 2014, vol. 8221, pp. 90–102.

[14] G. Anders, A. Schiendorfer, J.-P. Steghöfer, and W. Reif, "Robust Scheduling in a Self-Organizing Hierarchy of Autonomous Virtual Power Plants," in *27th International Conference on Architecture of Computing Systems (ARCS'14)*, Feb 2014, pp. 1–8.

[15] G. Anders, F. Siefert, M. Mair, and W. Reif, "Proactive Guidance for Dynamic and Cooperative Resource Allocation under Uncertainties," in *IEEE Eighth International Conference on Self-Adaptive and Self-Organizing Systems (SASO'14)*, Sept 2014, pp. 21–30.

[16] P. V. Hentenryck and R. Bent, *Online Stochastic Combinatorial Optimization*. The MIT Press, 2009.

[17] J.-P. Steghöfer, G. Anders, F. Siefert, and W. Reif, "A System of Systems Approach to the Evolutionary Transformation of Power Management Systems," in *Proceedings of INFORMATIK 2013 – Workshop on "Smart Grids"*, ser. Lecture Notes in Informatics, vol. P-220. Bonner Köllen Verlag, 2013.

[18] E. Tsang, *Foundations of Constraint Satisfaction*. Academic press London, 1993, vol. 289.

[19] A. Schiendorfer, J.-P. Steghöfer, and W. Reif, "Synthesis and Abstraction of Constraint Models for Hierarchical Resource Allocation Problems," in *Proceedings of the 6th International Conference on Agents and Artificial Intelligence (ICAART'14), Vol. 2*. SciTePress, 2014, pp. 15–27.

[20] J. Branke, *Evolutionary Optimization in Dynamic Environments*, ser. Genetic Algorithms and Evolutionary Computation. Norwell, MA, USA: Kluwer Academic Publishers, 2001, vol. 3.

[21] P. Klemperer, "What Really Matters in Auction Design," *The Journal of Economic Perspectives*, vol. 16, no. 1, pp. 169–189, 2002.

[22] P. Scott, S. Thiébaux, M. van den Briel, and P. Van Hentenryck, "Residential Demand Response under Uncertainty," in *Principles and Practice of Constraint Programming*, ser. Lecture Notes in Computer Science, C. Schulte, Ed. Springer Berlin Heidelberg, 2013, vol. 8124, pp. 645–660.

[23] P. Ruiz, C. Philbrick, E. Zak, K. Cheung, and P. Sauer, "Uncertainty Management in the Unit Commitment Problem," *IEEE Transactions on Power Systems*, vol. 24, no. 2, pp. 642–651, May 2009.

[24] A. Shapiro, "On Complexity of Multistage Stochastic Programs," *Operations Research Letters*, vol. 34, no. 1, pp. 1–8, 2006.

[25] P. Ströhle, E. H. Gerding, M. M. de Weerdt, S. Stein, and V. Robu, "Online Mechanism Design for Scheduling Non-preemptive Jobs Under Uncertain Supply and Demand," in *Proceedings of the 13th International Conference on Autonomous Agents and Multi-agent Systems*, ser. AAMAS '14. Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems, 2014, pp. 437–444.

[26] R. K. Dash, N. R. Jennings, and D. C. Parkes, "Computational-Mechanism Design: A Call to Arms," *IEEE Intelligent Systems*, vol. 18, no. 6, pp. 40–47, 2003.

[27] G. Chalkiadakis, V. Robu, R. Kota, A. Rogers, and N. R. Jennings, "Cooperatives of Distributed Energy Resources for Efficient Virtual Power Plants," in *Proceedings of the 10th International Conference on Autonomous Agents and Multiagent Systems*, ser. AAMAS '11, vol. 2. Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems, 2011, pp. 787–794.

[28] P. Vytelingum, S. D. Ramchurn, T. D. Voice, A. Rogers, and N. R. Jennings, "Trading Agents for the Smart Electricity Grid," in *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems*, ser. AAMAS '10, vol. 1. Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems, 2010, pp. 897–904.

[29] R. Dash, P. Vytelingum, A. Rogers, E. David, and N. Jennings, "Market-Based Task Allocation Mechanisms for Limited-Capacity Suppliers," *Systems, Man and Cybernetics, Part A: IEEE Transactions on Systems and Humans*, vol. 37, no. 3, pp. 391–405, 2007.

[30] H. F. Wedde, "DEZENT – A Cyber-Physical Approach for Providing Affordable Regenerative Electric Energy in the Near Future," in *38th EUROMICRO Conference on Software Engineering and Advanced Applications (SEAA'12)*, 2012, pp. 241–249.

[31] K. Kok, C. Warmer, and R. Kamphuis, "PowerMatcher: Multiagent Control in the Electricity Infrastructure," in *Proceedings of the 4th International Joint Conference on Autonomous Agents and Multiagent Systems*, ser. AAMAS '05. New York, NY, USA: ACM, 2005, pp. 75–82.

[32] J. Li, G. Poulton, and G. James, "Coordination of Distributed Energy Resource Agents," *Applied Artificial Intelligence*, vol. 24, no. 5, pp. 351–380, 2010.