Natural Computing (2022) 21:155–171
https://doi.org/10.1007/s11047-020-09824-0

Classifying Metaheuristics: Towards a unified multi-level classification system

Helena Stegherr[1] · Michael Heider[1] · Jörg Hähner[1]

Accepted: 12 November 2020 / Published online: 12 December 2020

Abstract
Metaheuristics provide the means to approximately solve complex optimisation problems when exact optimisers cannot be utilised. This led to an explosion in the number of novel metaheuristics, most of them metaphor-based, using nature as a source of inspiration. Thus, keeping track of their capabilities and innovative components is an increasingly difficult task. This can be resolved by an exhaustive classification system. Trying to classify metaheuristics is common in research, but no consensus on a classification system and the necessary criteria has been established so far. Furthermore, a proposed classification system can not be deemed complete if inherently different metaheuristics are assigned to the same class by the system. In this paper we provide the basis for a new comprehensive classification system for metaheuristics. We first summarise and discuss previous classification attempts and the utilised criteria. Then we present a multi-level architecture and suitable criteria for the task of classifying metaheuristics. A classification system of this kind can solve three main problems when applied to metaheuristics: organise the huge set of existing metaheuristics, clarify the innovation in novel metaheuristics and identify metaheuristics suitable to solve specific optimisation tasks.

Keywords Nature-inspired algorithms · Metaheuristic · Classification · Stochastic optimisation

Mathematics Subject Classification 68Q05 · 90C59

1 Introduction

Metaheuristics play an important role in solving complex optimisation problems where mathematical optimisation is infeasible either due to long computation times or incomplete problem definitions. They use a combination of heuristic methods, arranged in a higher level framework, to provide more efficient search space exploration capabilities (Blum and Roli 2003) and can be described as a design pattern, rather than an algorithm (Lones 2014; Krawiec et al. 2018). However, a metaheuristic can not perform equally well on all optimisation problems, as is proven by the *No Free Lunch* theorems for optimisation (Wolpert and Macready 1997). These and other factors have lead to an explosion in the number of nature-inspired metaheuristics over the past years.

Sörensen criticises this excess in the publication of "new" metaheuristics, especially the focus on metaphors in their design (Sörensen et al. 2017). While early metaheuristics provided novel ideas and successful strategies to solve complex optimisation problems, more recent contributions lack improvements and are often too similar to already established ones (Sörensen 2013). Furthermore, the metaphors are increasingly far-fetched, with no relation to optimisation in nature, thus decreasing their quality and that of the respective algorithm (Sörensen 2013).

Additionally, the evaluation of metaheuristics is lacking diligence. They are often only insufficiently tested, i.e. without considering the degree of adaptations of the algorithms and without the appropriate statistical methods, or tested only on a small subset of problems (Sörensen 2013). This leads to problems when trying to derive more general statements on the performance of these metaheuristics. The

✉ Helena Stegherr
helena.stegherr@informatik.uni-augsburg.de

Michael Heider
michael.heider@informatik.uni-augsburg.de

Jörg Hähner
joerg.haehner@informatik.uni-augsburg.de

[1] Institute of Computer Science, Organic Computing Group, Eichleitnerstr. 30, 86159 Augsburg, Germany

testing is also increasingly competitive, as each metaheuristic is supposed to outperform previous ones but the reasons for this superiority are not analysed and it diverts attention from fundamental research. Thus, relevant insight cannot be gained by this procedure (Hooker 1995).

The most detrimental aspect of the extensive publication of novel methods is that it leads to a concentration of effort on invention. Other important research areas (e.g. further improvements, theory) in the field of metaheuristics are tended to only sparsely (Sörensen 2013; Lones 2014; Del Ser et al. 2019; Molina et al. 2020). This results in a deficit in progress, especially when fundamental research on metaheuristics is concerned.

A suitable classification system for metaheuristics may alleviate this problem. There is a necessity to provide a formal analysis of novel metaheuristics, as well as to systematically evaluate their contribution to the field (Nesmachnow 2014). There are, however, several difficulties when trying to achieve this. One is the overloaded term metaheuristic. It describes the framework in which optimisation algorithms can be developed, as well as the algorithm itself (Sörensen and Glover 2013). Thus, when analysing metaheuristics, it has to be distinguished which usage of the term applies. Another problem is the formal presentation of nature-inspired metaheuristics (Lones 2014). Especially the metaphor-related notation of most metaheuristics exacerbates the tasks of comparing and analysing them, as well as the detection of novel approaches (Sörensen 2013; Del Ser et al. 2019).

When utilised according to a defined set of criteria, a classification allows for the integration of metaheuristics into the overall context. Additionally, it can provide a basis for structuring further research in the field. Metaheuristics chosen for experimental comparisons can be selected due to the applying classification criteria. Thus, the comparison of algorithms in different stages of their development could be avoided. Theoretical approaches can gain additional information through a formal classification of metaheuristics and the algorithmic performance can be ascribed to the incorporation of specific functional components. Furthermore, finding a suitable metaheuristic for a complex optimisation problem may be facilitated by knowledge about crucial features of the metaheuristic through classification.

Classification systems, or taxonomies, for metaheuristics are an important part of research and new strategies are frequently developed and published (cf. Sect. 3). However, there is a problem inherent to these current approaches. Many expect to have provided the comprehensive classification system but none has been established among all researchers. This has several reasons: First, by definition, a classification (or taxonomy) is only complete if its classes are mutually exclusive (cf. Sect. 2). Second, such a classification system takes time to develop and requires discussion and ultimately consensus on the utilised criteria. Finally, the criteria and, thus, the whole classification system tend to be constructed focusing on a certain goal and researchers have to be aware of that.

If we look at taxonomies in biology, we can observe the procedure of their development. Over the last centuries, there were a number of different taxonomies with different criteria and goals, with some getting more attention and acceptance and some being almost ignored (Stevens 1984, 2003). Only a few publications, for example by Linnaeus or Darwin, made a lasting impression on biological taxonomy, but even those ideas were revisited and revised. The taxonomy achieved by this long progress is almost comprehensive by now, but still minor changes have to be made (Adl et al. 2012, 2018).

The process of developing a comprehensive classification system for metaheuristics will also take time, though possibly not as much as a biological taxonomy, as there are (at least by now) far less metaheuristics than life forms. But as in biology, it is necessary to continually review, summarise and refine classification systems until a comprehensive taxonomy is established. Therefore, this paper aims to give an overview of classification strategies for metaheuristics. It will also evaluate these strategies with respect to the information that can be gained by them. Furthermore, we will present a possible combination of common classification schemes aiming at a first draft for a comprehensive taxonomy. Section 2 gives the necessary definitions and Sect. 3 an extensive list of previous classification schemes. Section 4 provides an overview of important steps that should be considered when designing and evaluating novel metaheuristics. The existing classification schemes are summarised and structured to be utilised in a multi-level classification system (Sect. 5) and an example for classifying metaheuristics with it is presented in Sect. 6. We discuss our propositions in Sect. 7.

2 Definitions

Before we elucidate different classification schemes, we provide the relevant definitions that will help to find differences between them.

The central element is the definition of the term metaheuristic. We use the term as defined by Sörensen and Glover (2013) which includes the differentiation between metaheuristic algorithms and frameworks:

> A metaheuristic is a high-level problem-independent algorithmic framework that provides a set of guidelines or strategies to develop heuristic optimization algorithms. The term is also used to refer to a problem-specific implementation of a heuristic

> optimization algorithm according to the guidelines expressed in such a framework.

In the following we try to distinguish classification schemes referring to algorithms from those referring to frameworks. This is also important when considering the novelty of metaheuristics, as new algorithms are easier to design than new frameworks (Lones 2019).

Special considerations have to be made regarding extensions and variants of metaheuristics, especially in the context of classification. In most of those variants, the framework remains unchanged, whereas the algorithm contains new features. Thus, in our classification we will treat these variants as different metaheuristic algorithms.

Classification schemes vary immensely in their complexity and in the information that can be gained by applying them. They are designed with different objectives in mind, from a simple overview of metaheuristics and their inspirations to a detailed analysis of their components. Therefore, it is vital to define the term classification and dissociate it from the concept of categorisation.

Our definition of classification is the one provided by Jacob (2004):

> Classification as a process involves the orderly and systematic assignment of each entity to one and only one class within a system of mutually exclusive and nonoverlapping classes.

Thus, a classification scheme sorts metaheuristics according to their similarities using a set of predefined criteria. The most important part of this definition is that each classified entity can only be a member of one single class. Classification can be used to gain knowledge or understanding of the classified entity (Jacob 2004). Taxonomy can be used synonymously to classification. It describes the procedure of developing a classification system, usually one with an internal hierarchy (Jacob 2004).

Categorisation describes a similar but in some aspects fundamentally different concept. The definition provided by Jacob is (2004):

> Categorisation is the process of dividing the world into groups of entities whose members are in some way similar to each other.

Categorisation is used to establish an organisation in a complex system. The entities in the system are divided into predefined categories according to specific attributes inherent to the entities. This process can also provide information that exceeds the simple grouping as new relationships can be formed. In contrast to classification, an entity can be part of more than one category, depending on the underlying context (Jacob 2004).

Based on these definitions, we use the term categorisation for organising systems with criteria allowing for a sorting of metaheuristics in several groups at the same time. Schemes are also deemed a categorisation if its criteria do not allow for a comprehensive sorting, i.e. many metaheuristics belong to the same group due to missing criteria.

Definition 1 (*Categorisation*) An ordering system with criteria allowing for a sorting of metaheuristics in several groups at the same time or several metaheuristics in one group.

Conversely, we will use the term classification if metaheuristics can be distinctly characterised by the defined criteria. Additionally, there should not be groups of metaheuristics at the end of a classification scheme. If groups remain, it can be assumed that the metaheuristics within the group are identical, at least with respect to the classification criteria.

Definition 2 (*Classification*) A comprehensive ordering system with criteria allowing for unambiguous assignment of distinct metaheuristics in individual classes.

The goals of classifying metaheuristics are manifold, with two aspects accentuated. One is the integration of metaheuristics into a comprehensive metaheuristic context. Novelty and exceptionality of the metaheuristic algorithms and the underlying frameworks can be evaluated and unique features, as well as the overall benefit for the whole field, can be highlighted. Furthermore, classification provides a basis for the selection of metaheuristics that are used in a meaningful comparison. The other important aspect is to gain an understanding of the metaheuristic-problem relations. Each metaheuristic has specific problems it performs especially well on. To know these problems in advance and to understand why these relationships exist is essential for the purposeful application of metaheuristics (Woodward and Swan 2010).

There are, however, some major problems when trying to classify metaheuristics. One is the lack of a consistent notation in their description. Most nature-inspired metaheuristics use a notation based on their metaphor instead of using common optimisation terms (Sörensen 2013). This causes difficulties in formulating common classification criteria and in extracting the necessary information from the metaheuristic to apply those criteria. Another important problem is the lack of guidelines for classification criteria (Fister et al. 2013). Current classification criteria are chosen according to what seems most suitable by the respective researchers. This, however, leads to many different and often very coarse-grained classification schemes resulting in only little information gain. Some schemes are more detailed but utilising them to classify metaheuristics is

rather difficult. Additionally, the criteria for a classification scheme are often selected regarding their easy applicability instead of their meaningfulness. Thus, complex criteria with a high information gain are used only infrequently because the analysis takes time and effort.

3 Classification schemes in literature

We now provide an overview of classification approaches in literature. These range from different taxonomic structures to detailed component analysis. We focus on the applied criteria, not on the metaheuristics classified. Furthermore, we try to present them ordered from simple to more complex and detailed classification schemes whose criteria can be utilised for a comprehensive classification system and we distinguish those who rather present a categorisation according to the definitions given in Sect. 2. We also differentiate approaches classifying metaheuristic algorithms from those classifying their frameworks. A summary of this overview is given in Tables 1 and 2.

Almufti et al. (2019) present a taxonomy based on the biological fields the metaheuristics take their inspiration from. They make a distinction between evolutionary-based, swarm-based and ecology-based metaheuristics. According to our definitions, this approach can be applied on frameworks as well as on algorithms. It is, however, a categorisation, as no criteria are applied that allow for an unambiguous distinction of metaheuristics.

Sörensen and Glover (2013) also provide a taxonomic structure, grouping local search metaheuristics, constructive metaheuristics and population-based metaheuristics. The approach is directed at frameworks and presents a categorisation, as they stress that a metaheuristic can be part of more than one group.

Another taxonomic approach is given by Binitha and Sathya (2012). They divide frameworks or algorithms according to their inspiration, i.e. evolution, swarms and ecology. Furthermore they add a comparative analysis regarding the terms of representation, operators, areas of application and control parameters. Though some of the criteria in this analysis can be used for a classification of metaheuristic algorithms, the taxonomy represents a categorisation, as no clear distinction can be made.

Rajpurohit et al. (2017) also divide metaheuristics into groups according to their source of inspiration. They differentiate between biology-based, physics-based, social-based, music-based, chemical-based, sport-based, mathematics-based and swarm-based systems. This represents a categorisation approach and is applicable on metaheuristic frameworks and algorithms.

Fister et al. (2013) use a categorisation approach, dividing nature-inspired metaheuristics into bio-inspired, physics- or chemistry-inspired and all "other". Bio-inspired metaheuristics can be swarm-based or not. Additionally, it is taken into account whether the metaheuristic is trajectory-based or population-based, attraction-based or not and rule-based or equation-based. Though there are more criteria taken into account than in previously described approaches, it is still a categorisation as there is no distinctive representation of a metaheuristic after application. This categorisation can be applied to frameworks and algorithms.

The criteria presented by Nesmachnow (2014) partition metaheuristics into trajectory based (exploitation oriented) and population based (exploration oriented). Furthermore, a distinction is made between local search based and constructive and memory-based and memory-less approaches. These criteria can be applied to categorise frameworks and algorithms, but they do not allow for an unambiguous classification.

Pazhaniraja et al. (2017) provide an extensive set of classification criteria for metaheuristic algorithms. It contains the existence of constraints, the physical structure of the problem (optimal control or non-optimal control), the nature of the equation (linear/quadratic, polynomial or non-linear), the values the decision variable can take (integer or real-valued), the nature of the variable (deterministic or stochastic), the separability of the function and the number of objective functions. These criteria cannot (or only with limitations) be applied when classifying a metaheuristic framework because they depend strongly on the respective implementation of the metaheuristic algorithm. They are, however, sufficiently detailed to be utilised in a classification, not only in a categorisation.

Birattari et al. (2001) published one of the first classification schemes. They differentiate between trajectory and discontinuous methods, population-based and single-point search approaches, memory-usage or no memory, one or various neighbourhood structures, a dynamic or static objective function and nature- or non-nature inspired metaheuristics. Though there is only a small number of criteria, the combination of those can provide a significant though not complete classification of metaheuristic frameworks and algorithms.

Nabaei et al. (2016) present a topology system containing important features of metaheuristic algorithms and frameworks, although the application on frameworks is somewhat restricted since some features are implementation-dependent. These features can be used as criteria for a classification approach. They distinguish whether a metaheuristic does or does not contain an evolutionary operator, multiplier updating, self-adaptive adjusting, parallel learning, fuzzy-adjusting, a quantum model, real-code possibilities, variable adjusting, the Nelder-Mead algorithm, hybridised versions, rank-based schemes, orthogonal

Table 1 Summary of selected classification systems—part 1: the table lists the publication, whether the criteria reference framework and/or algorithm classification (and which is in focus), our assessment of the type (classification or categorisation) of criteria in the system and the relevant criteria mentioned in the respective publication. Continued in Table 2 where applicable, criteria are treated as logical values ($\oplus$: criteria are mutually exclusive). More complex criteria are separated via semicolon and then listed

Publication	Applicable to	Type	Criteria
Almufti et al. (2019)	Framework/ algorithm	Categorisation	Evolutionary $\oplus$ swarm-based $\oplus$ ecology-based
Sörensen and Glover (2013)	Framework	Categorisation	Local search-based $\vee$ constructive $\vee$ population-based
Binitha and Sathya (2012)	Framework/ algorithm	Categorisation	(Evolutionary $\oplus$ swarm-based $\oplus$ ecology-based) $\vee$ (single solution $\oplus$ population); operators, control parameter
Rajpurohit et al. (2017)	Framework/ algorithm	Categorisation	Biology-based $\oplus$ physics-based $\oplus$ social-based $\oplus$ music-based $\oplus$ chemical-based $\oplus$ sport-based $\oplus$ mathematics-based $\oplus$ swarm-based
Fister et al. (2013)	Framework/ algorithm	Categorisation	(Trajectory-based $\oplus$ population-based) $\vee$ attraction-based $\vee$ (rule-based $\oplus$ equation-based) $\vee$ nature-inspired $\vee$ bio-inspired $\vee$ swarm-based $\vee$ physics-inspired $\vee$ chemistry-inspired $\vee$ other
Nesmachnow (2014)	Framework/ algorithm	Categorisation	(Trajectory-based/exploitation-oriented $\oplus$ population-based/exploration-oriented) $\vee$ (local search based $\oplus$ constructive) $\vee$ memory-based
Pazhaniraja et al. (2017)	Algorithm	Classification	Constrained $\vee$ function separability $\vee$ (deterministic $\oplus$ stochastic) $\vee$ (decision variable: integer $\oplus$ real-valued) $\vee$ (single-objective $\oplus$ multi-objective); physical problem structure, nature of equation
Birattari et al. (2001)	Framework	Classification	(Trajectory $\oplus$ discontinuous) $\vee$ (population-based $\oplus$ single-point search) $\vee$ (memory $\oplus$ memoryless) $\vee$ (one $\oplus$ various neighbourhood structures) $\vee$ (dynamic $\oplus$ static objective function) $\vee$ (nature-inspired $\oplus$ non-nature inspired)
Nabaei et al. (2016)	Algorithm/ framework	Classification	Evolutionary operator $\vee$ multiplier updating $\vee$ self-adaptive adjusting $\vee$ parallel learning $\vee$ fuzzy adjusting $\vee$ quantum model $\vee$ real-code $\vee$ variable adjusting $\vee$ Nelder-Mead $\vee$ existing hybrids $\vee$ rank-based $\vee$ orthogonal learning $\vee$ θ-search $\vee$ quantum-behaved $\vee$ opposition-based learning $\vee$ chaos theory
Fausto et al. (2019)	Framework/ algorithm	Categorisation	Evolution-based $\oplus$ swarm-based $\oplus$ physics-based $\oplus$ human-based
	Framework/ algorithm	Classification	Characteristics of metaheuristics on exploration and exploitation (selection mechanism, attraction operators, iteration dependency), computational complexity (population sorting, population-related measurements, variable fitness function computation), requirement of additional memory, parameter tuning process, implementation difficulty

learning, θ-search, quantum-behaviour, opposition-based learning and chaos theory.

A categorisation as well as characteristics of metaheuristics algorithms and frameworks are presented by Fausto et al. (2019). They divide nature-inspired metaheuristics into evolution-, swarm-, physics- and human-based approaches. Furthermore, they provide a list of characteristics that can apply to metaheuristics which can be utilised as classification criteria. Those characteristics related to the exploration and exploitation capabilities are the selection mechanism (greedy, individual greedy or non-greedy), the attraction operator (e.g. none, global best, multiple, global, personal) and the iteration dependency. Characteristics determining the computational complexity are the presence of population-sorting, of population-related measurements and of a variable fitness function computation. In addition to those, metaheuristics can require additional memory, have different parameter tuning processes (tuneless, easy or hard) and pose variable degrees of difficulty in implementation (low, medium or high).

Sergienko et al. (2009) provide schemes designed for the classification of combinatorial optimisation algorithms. One scheme is suitable for framework categorisation, the other one is appropriate for extracting classification criteria. The categorisation approach divides combinatorial optimisation algorithms into sequential algorithms, deterministic local search algorithms, stochastic local search algorithms, evolutionary algorithms, swarm algorithms,

Table 2 Summary of selected classification systems—part 2: the table lists the publication, whether the criteria reference framework and/or algorithm classification (and which is in focus), our assessment of the type (classification or categorisation) of criteria in the system and the relevant criteria mentioned in the respective publication. Where applicable, criteria are treated as logical values ($\oplus$: criteria are mutually exclusive). More complex criteria are separated via semicolon and then listed

Publication	Applicable to	Type	Criteria
Continuation of Table 1			
Sergienko et al. (2009)	Framework	Categorisation	Sequential algorithm $\oplus$ deterministic local search $\oplus$ stochastic local search $\oplus$ evolutionary algorithm $\oplus$ swarm algorithm $\oplus$ scanning method $\oplus$ special method
	Framework/ algorithm	Classification	(Deterministic $\oplus$ stochastic) $\vee$ (trajectory-continuous $\oplus$ trajectory-discontinuous) $\vee$ [sequential $\oplus$ (iterative: one-point $\oplus$ population-based)] $\vee$ [unchanged search landscape $\oplus$ (search space change $\oplus$ objective function change $\oplus$ neighbourhood change)] $\vee$ [without memory $\oplus$ (storage memory $\oplus$ control memory $\oplus$ adaptive memory)] $\vee$ [without adaptation $\oplus$ (parameter adjustment $\oplus$ component choice $\oplus$ special model of the problem)] $\vee$ (problem-oriented $\oplus$ model-oriented); structural complexity
Blum and Roli (2003)	Algorithm/ framework	Categorisation	nature-inspired $\vee$ (search: population-based $\oplus$ single point) $\vee$ (objective function: dynamic $\oplus$ static) $\vee$ (neighbourhood structures: one $\oplus$ various) $\vee$ memory usage
	Framework/ algorithm	Classification	Intensification and diversification capabilities with I&D framework
Dokeroglu et al. (2019)	Algorithm/ framework	Classification	Number of hyperparameters, stages with exploration/exploitation balancing, hybridisation possibilities, inclusion of local search mechanism
Lones (2014)	Framework	Classification	(Local search $\oplus$ EA $\oplus$ PSO $\oplus$ ACO)-based metaheuristics
Molina et al. (2020)	Framework	Categorisation	Breeding-based $\oplus$ swarm intelligence (aquatic animals $\oplus$ terrestrial animals $\oplus$ flying animals $\oplus$ microorganisms $\oplus$ other) $\oplus$ physics and chemistry (physics $\oplus$ chemistry) $\oplus$ plants based $\oplus$ miscellaneous
	Framework	Classification	Differential vector movement [guided by whole population $\oplus$ guided by representative $\oplus$ guided by group (subpopulation $\oplus$ neighbourhood)] $\oplus$ solution creation (combination $\oplus$ stigmergy)
Li et al. (2020)	Framework	Classification	Single-elite $\oplus$ multi-elites $\oplus$ none-elite
Sergeyev et al. (2018)	Algorithm/ framework	Classification	Visualisations of performance evaluation
Woodward and Swan (2010)	Algorithm	Classification	Performance on problems

scanning methods and special methods. Additionally, they present a more extensive classification scheme. The main characteristics of combinatorial optimisation algorithms are the decision-making principle (deterministic or stochastic), the structure complexity (simple, metaheuristic, hyperheuristic, hybrid algorithm or hybrid metaheuristic), the type of trajectory (continuous or discontinuous), the type of spaces (sequential or iterative), the influence on the search landscape (do or do not change search landscape), the use of memory (with or without), the incorporation of adaptation or training (with adaptation or without) and the special model of the problem (problem-oriented or model-oriented). Some of these are subdivided into further criteria. Iterative algorithms can be either one-point algorithms or population-based algorithms. Algorithms changing the search landscape can be partitioned according to this change, e.g. the search space, the objective function or the system of neighbourhoods. The type of memory can also be further characterised, e.g. storage, control or adaptive memory. Finally, the algorithms with adaptation can be divided into those with automatic

adjustment of parameters, those with choice of components and those with a special model of the problem.

According to Blum and Roli (2003), the most common criteria for classifying metaheuristics are the source of inspiration (nature or not), population-based or single point search, the objective function (dynamic or static), neighbourhood structures (one or various) and memory usage. These, however, do not suffice for a detailed classification and we would rather consider them criteria for categorisation. However, Blum and Roli provide an approach for classifying a metaheuristic framework according to its intensification and diversification capabilities. They call their underlying scheme I&D framework. They review the metaheuristic components that influence intensification or diversification during the search process and evaluate if they are guided by the objective function, guided by other functions or completely random. Thus, the components of the metaheuristic and their respective I&D evaluation provide criteria for a classification approach.

The stages where exploration and exploitation (and diversification and intensification) can be balanced is also one important feature of metaheuristics as described by Dokeroglu et al. (2019). The other important features are the number of hyperparameters, the possibility of hybridisation and the inclusion of a local search mechanism. These features can be used as classification criteria for metaheuristic algorithms and frameworks, though the number of hyperparameters is implementation-dependent and thus only applicable when classifying algorithms.

Lones (2014) provides a unique approach to classification. It focusses on the use of specific heuristics in the metaheuristics. Thus, a local search metaheuristic utilises *Neighbourhood Search*, *Hill Climbing*, *Accepting Negative Moves*, *Multi-Start* and/or *Adaptive Memory Programming*. Evolutionary algorithm (EA) metaheuristics contain *Population-based Search* and/or *Intermediate Search* as their basic heuristics. Particle Swarm Optimisation (PSO) metaheuristics contain *Directional Search* and/or *Variable Neighbourhood Search* and Ant Colony Optimisation (ACO) metaheuristics also contain *Search Space Mapping*. Lones (2019) applies this strategy to 32 metaheuristics to find their common heuristic bases. This approach can be used for framework classification schemes in general, as it is an efficient way to find similarities in heuristic usage and highlight different strategies when selecting the use of these heuristics.

Molina et al. (2020) provide extensive and insightful new approaches on taxonomies of metaheuristics. The first proposed taxonomy classifies metaheuristics according to their metaphor and represents a categorisation. They distinguish breeding-based, swarm intelligence, physics and chemistry, plants based and other metaheuristics. Furthermore, swarm intelligence metaheuristics can be further divided into ones inspired by aquatic animals, terrestrial animals, flying animals, microorganisms or other. Physics and chemistry metaheuristics are either physics based or chemistry based. This structure provides a basic overview on different kinds of metaphor-inspiration. In addition to this categorisation, Molina et al. (2020) provide a classification system based on the behaviour of the metaheuristics. This may be the first attempt to use criteria relating to the overall behaviour of metaheuristic frameworks. They distinguish two types of procedures metaheuristics utilise to generate new solutions: differential vector movement and solution creation. Differential vector movement can be guided by the whole population, by a representative of the population or by a group (either a subpopulation or a neighbourhood). Solution creation can be performed by combination or by stigmergy. This taxonomy allows for a detailed but again not complete classification of metaheuristics according to their behaviour.

Another classification approach based on the behaviour of metaheuristics is provided by Li et al. (2020). They present a taxonomy distinguishing the number of high-fitness individuals in the metaheuristic, single-elite, multi-elite or none-elite. This approach is similar to the classification of Molina et al. (2020), especially to their description of differential vector movement metaheuristics.

The last two classification approaches are based on the performance of a metaheuristic on specific problems. Sergeyev et al. (2018) provide a systematic comparison of metaheuristics based on the visualisation of their respective performance. It requires extensive testing of metaheuristics on different problems and a detailed evaluation. The results are visualised and can be used to compare the metaheuristics. These performance relations to different problems can be used to classify metaheuristics, though it is quite time-consuming and complex to apply on every metaheuristic algorithm–problem combination.

The other approach based on the performance of a metaheuristic on specific problems or problem classes is presented by Woodward and Swan (2010). They propose a preferably complete mapping of metaheuristics to problem classes, as the *No Free Lunch* theorem states that no metaheuristic can perform well on all problem classes, but, on the other hand, it should be good at some problem classes. The difficulty is, however, the selection of the respective problems. Which problems are encompassed in one class is not readily detected. Still, defining classification criteria with this approach could provide extensive insights.

This summary of classification approaches may very well be incomplete as we primarily focussed on including classification and categorisation systems for metaheuristics that have been proposed directly as "classification systems" in recent years. Classification schemes not listed

here as well as novel criteria should, however, be researched and considered when attempting classification of metaheuristics.

There are, of course, many surveys comparing metaheuristics but they often only provide a list of the respective algorithms. Comparisons of different swarm-based methods are common (Selvaraj et al. 2014), as are evaluations on a few benchmark functions (Krishnanand et al. 2009; Can and Alatas 2017) or conceptual comparisons listing algorithmic details (Binitha and Sathya 2012; Al-Amry and Al-Gaphari 2018). However, most such surveys do not entail a critical view on metaheuristics and their results. There is also almost no information gain, as the comparison as well as the evaluation are either too superficial or too specialised on a certain implementation.

An interesting observation can be made with hyperheuristics, which are used to construct and improve metaheuristics for specific problems (Krawiec et al. 2018; Hong et al. 2018). While an extensive classification system for those algorithms falls out of the scope of this work, we would like to acknowledge their ability to determine favourable combinations of metaheuristic components and mapping those to specific problems, thereby, performing an intrinsic categorisation.

4 How metaheuristics should be treated

We now want to give an overview of ideas on what the procedure of designing and testing a metaheuristic should look like. If these steps are considered consequently, it will facilitate dealing with metaheuristics and help to show their full potential. Classification is an important tool to this end.

The central aspect is that the focus of metaheuristic research, for existing ones as well as for new designs, should lie on gaining an overall scientific understanding (Sörensen et al. 2017). Gaps in research should be closed and there should be an emphasis on theoretical developments (Del Ser et al. 2019; Nesmachnow 2014).

Thus, several key aspects have to be considered. First a uniform notation is required to make metaheuristics understandable and reusable (Del Ser et al. 2019). Furthermore, a standardised modelling language would provide explicit and machine-readable descriptions (Sörensen et al. 2017; Swan et al. 2015). This would also yield advantages when framing novel approaches in previous research (Sörensen 2013).

Second, evaluation and comparison of metaheuristics must be based on a standardised procedure, including theoretical as well as empirical approaches. A theoretical comparison can highlight the properties of the metaheuristic and identify superior behaviour and novelty attributable to those (Del Ser et al. 2019). This may be facilitated by deconstructing the new algorithms, thus providing a component-based view as proposed by Sörensen (2013). Empirical approaches must also be standardised. This encompasses the development and application of appropriate testing protocols, a meaningful statistical analysis and the thoughtful selection of benchmark problems (Sörensen et al. 2017; Sörensen 2013; Del Ser et al. 2019; Nesmachnow 2014).

Preferably, a general and unified framework is devised, in which all metaheuristics can be included (Nesmachnow 2014; Del Ser et al. 2019; Molina et al. 2020). This would facilitate direct comparisons and evaluations, as well as the development and improvement of metaheuristics.

These aspects are important for available metaheuristics and for novel approaches. Novel metaheuristics, however, need additional considerations. They must provide a benefit in comparison to previous approaches. Thus, they have to show a decrease in time, cost or complexity when solving complex problems (Almufti et al. 2019). Novel metaheuristics should also be able to handle different function types, show good convergence properties and provide the possibility for parallelisation (Almufti et al. 2019). They should include only few control parameters, which should be set according to specific rules or be adaptive (Binitha and Sathya 2012; Almufti et al. 2019; Sörensen 2013; Dokeroglu et al. 2019). Novel metaheuristics can improve both their overall, as well as their component-wise, reusability and comprehensiblity for a wider audience by employing sensible design patterns (Krawiec et al. 2018). Furthermore, it is important to consider the problems metaheuristics are supposed to solve. The problem representation is an important factor in the design of a metaheuristic and it would be even more beneficial if novel approaches would be based on insights into those problem structures (Binitha and Sathya 2012; Sörensen 2013).

This list of prerequisites for handling metaheuristics and especially novel approaches entails several difficulties. In order to integrate a novel metaheuristic into current research, a number of important steps have to be considered, which is quite time-consuming. A larger problem, however, results from such steps still lacking a generally accepted procedure. Furthermore, it is difficult to find suitable metaheuristics for comparisons, as there is no concise survey of all metaheuristics listing every important feature and, for most metaheuristics, not even publicly available source code. A comprehensive classification scheme can help to alleviate some of these problems.

5 How to use classification for metaheuristic research

In this section, we formulate categorisation and classification strategies for analysing metaheuristics with a focus on providing a basis to some of the design steps in Sect. 4. We use the criteria gathered by literature research and combine them to provide classification schemes for several purposes. We also point out where additional criteria are necessary. Furthermore, we elaborate on the necessity of evaluation strategies and optimisation problem classification when classifying metaheuristics.

5.1 Categorisation

Although categorisation schemes can not provide the insights of a comprehensive classification, they can give an overview and indications on how to proceed with the handling of a metaheuristic. For example, when using the sources of inspiration as criteria for categorisation, it is possible to deduce what the overall procedure of the algorithm might look like. This can help to find metaheuristics using a similar idea or metaphor, which can then be selected for a more detailed comparison. Further criteria can provide first insights into the exploration and exploitation capabilities of the metaheuristic (e.g. if it is trajectory-based or population-based).

A categorisation of metaheuristic algorithms can also help to find metaheuristics with the same adaptations. Especially established metaheuristics like the *Genetic Algorithm* (GA) have dozens of extensions and variants (Dokeroglu et al. 2019). Some novel metaheuristics contain these extensions in their basic version. Thus, comparing them to a basic GA would not accurately depict the potential the GA can provide. A short categorisation before starting this kind of comparison alleviates this problem.

Altogether, categorisation takes less time and fewer criteria than classification. However, it can only provide basic insights into the categorised metaheuristics. Categorisation is more suitable as a simple system of ordering before starting with a detailed evaluation or extensive classification.

5.2 Classification

When classifying metaheuristics, we have to differentiate between metaheuristic frameworks and algorithms. The classification of frameworks is important to detect novelty and to link framework-specific factors to optimisation success and to those problems the metaheuristic can solve best. The algorithm classification can be used to detect implementation details helpful for solving specific problems. These details can readily be transferred to other algorithms and be used for their improvement. Overall, classification can be used to specifically select metaheuristics for comparison, depending on their features described in the classification criteria.

Which classification criteria are selected is dependent on the intention of the classification. It is, however, indispensable that each metaheuristic is assigned to one class only at the bottom of the classification scheme. If two metaheuristics belong to the same class, they can be considered identical with respect to the intention of the scheme. If the intention is the complete classification of all metaheuristics, it is more difficult to select the appropriate criteria. They should not be too universal, as metaheuristics would not be properly classified into distinct classes. However, they must not be too specific, otherwise basically identical metaheuristics would be separated into different classes by irrelevant criteria.

A classification system can be used to find suitable metaheuristics for comparison. This is an important step, as the selection of an inadequate set of metaheuristics leads to a bias in the evaluation (LaTorre et al. 2020). There are two possibilities, either the selection of similar metaheuristics or the selection of inherently different metaheuristics. In this case, ‘similar’ means a high overlap of classification criteria, whereas ‘different’ means a small overlap. These criteria have to include, but are not limited to, changes in the overall metaheuristic structure, the presence of functional components and their manifestation, specific adaptations and extensions. Details of these criteria have to be provided by the classification system. The comparison of similar metaheuristics leads to the identification of small factors that are responsible for differences in performance or applicability. Alternatively, comparing different metaheuristics can help to point out their general functional elements and identify the main features that make them applicable to certain problems. Having to give this kind of justification for the metaheursitics chosen for a comparison can prevent the testing of (novel) metaheuristics against those versions of other (established) ones that do not contain the same improvements or additional features.

Providing a general and complete classification system is not in the scope of this paper. Discussion of the existing classification criteria is required as we consider them not sufficient to provide an unambiguous classification. We will, however, present a classification system for metaheuristics which incorporates basic classification. For each of the steps, we provide example criteria which can be elaborated on.

We first want to present a comprehensive structure of a classification system (Fig. 1). This system classifies metaheuristics on seven different levels. We call those levels

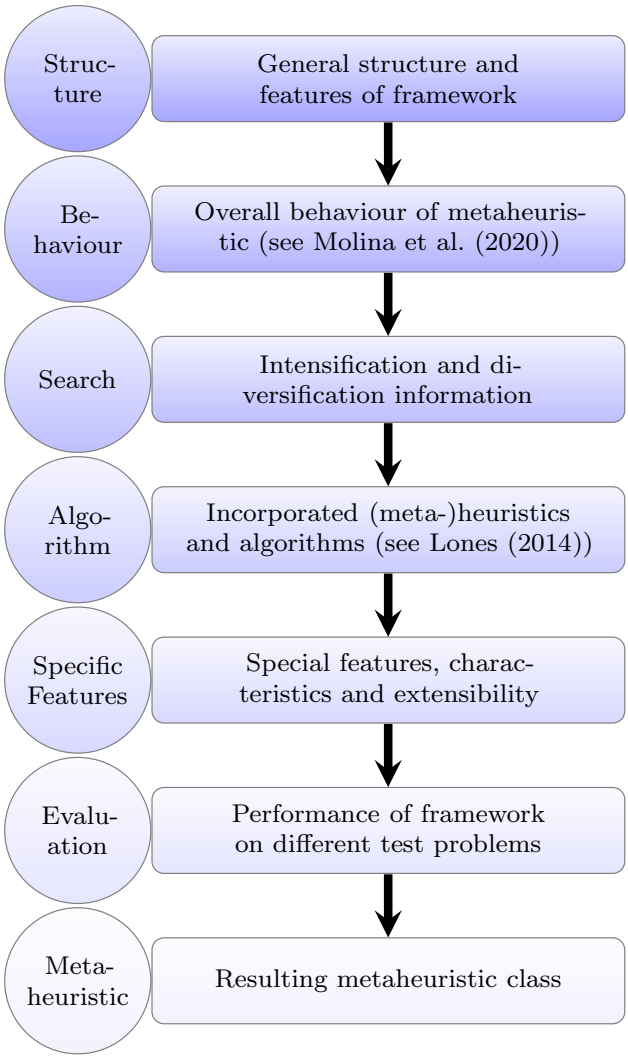

Fig. 1 Classification system for metaheuristics

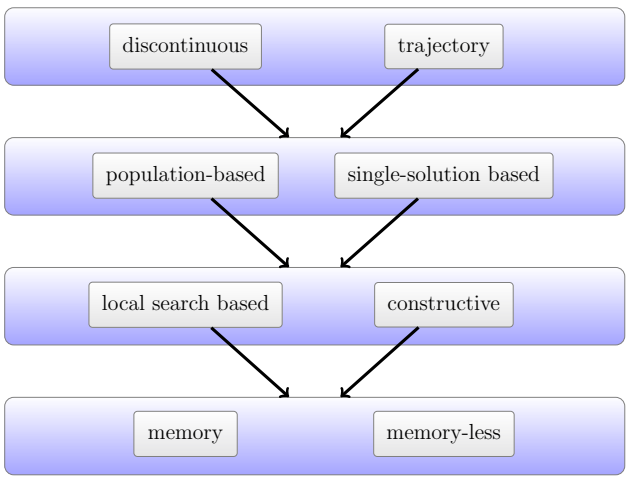

Fig. 2 Classification System: Detailed view on *Structure*

Structure, *Behaviour*, *Search*, *Algorithm*, *Specific Features*, *Evaluation* and *Metaheuristic*. Each level contains a specific set of related criteria, which are commonly used in previous classification or categorisation schemes. For some levels, however, the current criteria are not sufficient for a detailed classification or there is no consensus on their expressiveness. We will annotate these levels and provide suitable means to find acceptable criteria. Furthermore, we do not include criteria referring to the source of inspiration. Though these might help to gain an overview on metaheuristics, they do not provide any information on their quality, functionality or entailed features.

Structure: The first level will incorporate criteria related to the overall structure and the general features of the metaheuristic framework. The detailed classification steps are shown in Fig. 2 as a multi-layer decision process. Each layer represents a decision between two mutually exclusive criteria. The first two layers are closely related, as most discontinuous metaheuristics are population-based and trajectory methods are commonly single-solution based. In the next layer, local-search based methods are distinguished from constructive methods and in the last layer the distinction between memory-including and memory-less metaheuristics is made. This process provides initial structural information on the metaheuristic to be classified. Most previous classification schemes stop at this point or further include only inspiration-based criteria. However, we consider the information gained in this step valuable but insufficient for an extensive classification. Thus, further steps with additional criteria are required.

Behaviour: The second level of classification is based on the recently published taxonomy by Molina et al. (2020). This approach is the first taxonomy based on the overall search behaviour of the metaheuristic (Molina et al. 2020). Thus, it provides valuable new insights when applied to classifying metaheuristics.

The steps of this taxonomy are shown in Fig. 3. A first differentiation is made between new solutions being

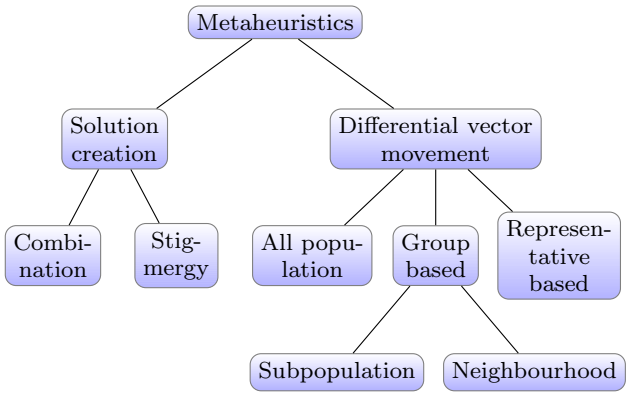

Fig. 3 Behaviour taxonomy described by Molina et al. (2020)

generated from a combination of several solutions (*Solution creation*) or from a previous solution (*Differential vector movement*) that may be replaced. Solution creation can be based on the combination of solution or on stigmergy. Differential vector movement can use all of the population as a basis for a new solution, but also a representative of the population or a group within the population. The group can contain a subpopulation or a neighbourhood (Molina et al. 2020).

This taxonomy presents a useful first distinction of metaheuristics. Its application provides a meaningful grouping of metaheuristics and facilitates evaluating their behaviour. Thus, it should be a standard for all metaheuristic publications. However, there are some issues when this taxonomy is used to compare metaheuristics in more detail. Molina et al. (2020) presented the most influential algorithms combined with those showing a high degree of similarity. For the *Particle Swarm Optimisation* algorithm, for example, there are 57 similar other metaheuristics (Molina et al. 2020). If differences between those have to be highlighted to understand the respective performances, to select them for a comparison or to analyse their features, it is still necessary to review all of them individually. This requires a lot of time and effort. Thus, we deem it necessary to extent this taxonomy by more specific criteria described in the next paragraphs.

Search: Classification on the third level is concerned with the intensification and diversification (and exploitation and exploration) capabilities of the metaheuristic framework. The criteria have to allow distinguishing which mechanisms are responsible for the capabilities and also quantifying those in some way.

Examples on how criteria reflecting the intensification and diversification mechanisms could look are given by Fausto et al. (2019), Dokeroglu et al. (2019) and Blum and Roli (2003). Fausto et al. relate the exploration and exploitation behaviour to certain characteristics of the metaheuristic. They focus on the selection mechanism, the attraction operators and the iteration dependency. These factors can be utilised as criteria for the classification in this step. Dokeroglu et al. provide stages of exploration and exploitation for several algorithms. However, they describe those stages in terms of the underlying metaphors of the metaheuristics. This results in the necessity to find standardised descriptions for those stages, so similarities between metaheuristics can be detected. The I&D components from the I&D framework by Blum and Roli present more readily usable classification criteria. Especially the basic (or intrinsic) components are relevant for the classification of a metaheuristic framework, as they are inherent to it. The strategic components are algorithm-dependent, thus providing means to classify metaheuristic algorithms. Again, some of the intrinsic components are presented in metaphor-specific terms and have to be translated to be generally applicable. Examples are the cooling schedule in *Simulated Annealing*, mutation, recombination and selection in *Evolutionary Computation* and the pheromone update in *Ant Colony Optimisation*. The remaining components, however, can be directly used as criteria in the classification system, e.g. the acceptance criterion and black box local search.

The second part of classification on this level is the quantification of intensification and diversification in metaheuristics. Blum and Roli (2003) provide a kind of quantification for the components in their I&D framework. They are weighted according to their guidance by the objective function, other functions or randomness. Thus, components guided solely by the objective function provide the maximum intensification possible, whereas components with sufficient guidance by other functions than the objective functions and randomness provide the maximum diversification.

The combination of the intensifying and diversifying components of a metaheuristic and the weighting of their effects presents adequate criteria for classification. However, it is important to use standardised descriptions of these components. It must also be evaluated if the weighting of their effects is sufficiently described by the

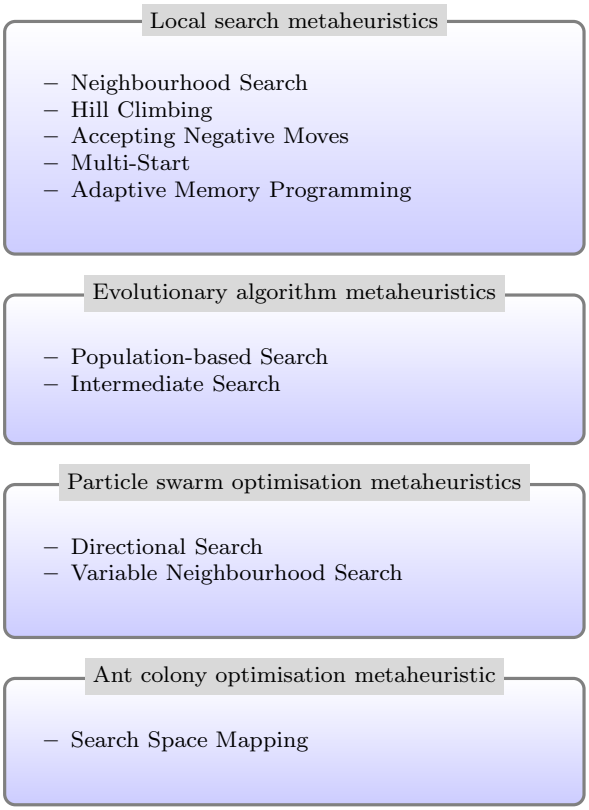

Fig. 4 Basic metaheuristic components as described by Lones (2014)

I&D framework or if another more detailed metric has to be found.

Algorithm: The fourth level of the classification system uses basic metaheuristic components incorporated in a metaheuristic framework as criteria for classification. A basis for these criteria was established by Lones (2014, 2019). His work analysed metaheuristics in order to find basic algorithmic structures incorporated in them. Thus, he started by analysing local search metaheuristics, evolutionary algorithms, particle swarm optimisation and ant colony optimisation to distinguish their basic metaheuristic components. They are summarised in Fig. 4.

Local search metaheuristics make use of neighbourhood search, hill climbing, accepting negative moves, multi-start and adaptive memory programming. Evolutionary algorithm metaheuristics also often incorporate some of these components, but additionally use population-based search and intermediate search. Particle swarm optimisation metaheuristics add directional search and variable neighbourhood search to the list of integrated components and ant colony optimisation metaheuristics often incorporate search space mapping. Most metaheuristics use one or more of the basic components. Their combination is responsible for the overall functioning of the metaheuristic. Thus, classification by the incorporated components provides information on similarities or differences. This list of metaheuristic components may not be complete yet. Therefore, all metaheuristics classified according to these criteria have to be analysed with respect to additional components.

Specific Features: The fifth level of our classification system relates to the special features and extensibility of the metaheuristic framework. Finding criteria suitable for this is far more intricate than for the previous levels, as they have to relate to the capabilities of the framework. However, it is possible to utilise some criteria common to metaheuristic algorithm classification. One example would be if the metaheuristic uses adaptive parameters. Some metaheuristic frameworks include algorithms in which this extension has already been made. In others, the extension is possible but was not incorporated yet and there might as well be metaheuristics where it is not possible to make parameters adaptive. This characteristic can be used as a classification criterion. Finding new criteria based on special or unique characteristics of metaheuristic frameworks is also possible. However, this requires extensive knowledge on metaheuristics and their functional features.

A first set of universally applicable criteria in this step are some of the characteristics provided by Fausto et al. (2019). The factors determining the computational complexity can readily be applied. However, finding criteria suitable for this step is a significant problem. All criteria gained by incorporating algorithm-specific criteria or by analysing the metaheuristic itself need to be carefully reviewed. Furthermore, a consensus is needed on the necessity and the suitability of those criteria. As long as there is no established classification procedure on this level, one has to either choose appropriate criteria for the respective classification scheme and provide an explanation on why they were chosen, or skip this step in the classification. Skipping a step might be sufficient for the respective aim of a metaheuristic framework classification but has to be carefully considered and the result of the classification has to be analysed. For metaheuristic algorithm classification, however, this level is important to provide a detailed distinction.

Evaluation: Level six of the classification system is concerned with the performance of metaheuristic frameworks on specific problems. This kind of matching can provide valuable insights when choosing a metaheuristic for the optimisation problem at hand. However, it is also one of the most difficult classification steps, as it is hard to define criteria and the effort of evaluating the metaheuristic according to these criteria is enormous.

Adequate criteria would be, for example, the problem structure and the dimensionality of the problem. For a more specific classification, however, the performance of metaheuristics on certain problem classes has to be known. This presents two problems: a standardised evaluation procedure of metaheuristics is necessary, as well as the knowledge of those problem classes. We will elaborate on these problems in Sect. 5.3 and 5.4.

Thus, this classification step needs the most research effort to become applicable in the overall system. However, it can also provide the greatest benefit. Knowledge about the performance of metaheuristics on diverse problem classes can facilitate the selection of a metaheuristic to solve a new problem. It can also help to determine which metaheuristics show identical behaviour on the same problem classes. Thus, it becomes possible to compare the structure of these metaheuristics and find the relevant components inducing these effects.

Metaheuristic: The last level of the classification system contains the distinct metaheuristic classes. All metaheuristics in one class are identical with respect to the applied classification criteria. Thus, a complete classification system containing all relevant criteria can show if different metaheuristic frameworks present unique ideas or if they utilise the same principles. On one hand, this facilitates establishing an order in the vast amount of metaheuristics, especially of those inspired by natural phenomena. On the other hand, it will help to evaluate novel metaheuristics and the ideas behind them faster and more precisely. Additionally, it is an important step towards resolving the problem of finding the most

suitable metaheuristics to tackle new complex optimisation problems.

This classification system can be utilised to classify metaheuristic algorithms as well as frameworks, though the algorithm classification will need a few adaptations. The basic levels will stay the same, whereas new criteria have to be found. These criteria should aim at finding the differences between the algorithmic designs. However, there are most likely more criteria necessary and they have to be carefully chosen. While metaheuristic framework classification presents a universal necessity to manage their large variety, algorithm classification is of interest in explicit problem formulations. Thus, it is essential to reach a consensus on the procedure and criteria for both cases.

5.3 Evaluating performance on problems

An important step towards creating exclusive and disjunct classes of metaheuristics is evaluating their performance on certain problems (cf. Sect. 5.2—*Evaluation*). In many cases metaheuristics have been and will be engineered to perform well on an individual problem at hand. While it is crucial that relations between individual problems are understood (cf. Sect. 5.4) in order to be able to classify metaheuristics on the basis of performance on similar problems, rather than having to test every metaheuristic on every specific problem, the way in which performance evaluations are conducted and which metrics are evaluated has to be equally deliberate and reproducible (Barr et al. 1995).

An important part of performance evaluation is the structured planning of the experimental setup that goes beyond mere selection of benchmarking problems (Hooker 1994, 1995). Birattari et al. (2001) suggest relying on design of experiments techniques such as randomization, blocking and factor analysis and using the checklist for planning experiments previously proposed by Dean et al. (2017). The checklist is composed of nine individual steps (slightly adjusted to fit the metaheuristics domain):

1. Define objectives;
2. Identify variation sources;
3. Choose combinations of hyperparameters and context values;
4. Specify measurements (e.g. sensitivity), procedure and anticipated difficulties (additional sources of variation);
5. Run a pilot experiment;
6. Specify the explanatory model;
7. Outline the analysis of data;
8. Determine the number of observations;
9. Review decisions.

After concluding the design, the experiment should be executed. The obtained results are then to be interpreted with regard to statistical and practical significance (Barr et al. 1995). When comparing performances of multiple algorithms, methods like hypothesis testing on widely-used benchmarking problems should be used (García et al. 2010; Eftimov et al. 2017). A typical null hypothesis when performing our classifications fifth level, *Evaluation*, on metaheuristics that are regarded as similar up until then, would be that those fall into the same class. If this hypothesis can not be validated, the metaheuristics will be divided into distinct classes.

LaTorre et al. (2020) recently summarised the important steps of evaluating metaheuristics with a focus on the experimental validation of new algorithms. They present a set of guidelines and techniques that should ensure a fair, coherent and sound evaluation process and that should provide a common methodological basis. The guidelines are structured into four levels, *Benchmarks*, *Validation of Results*, *Components analysis and parameter tuning* and *Why is my algorithm useful?* (LaTorre et al. 2020). Each level contains techniques, recommendations and requirements that should be considered in a metaheuristic evaluation process. In this procedure, a comprehensive classification system can provide valuable information for the selection of the reference algorithms of the *Benchmark* guidelines. Furthermore, it provides a basis for the component analysis of the third level of guidelines as it includes relevant information on the performance of metaheuristics with specific components.

5.4 Optimisation problem classification

A classification of optimisation problems is required when classifying metaheuristics according to their performance on problems, as good algorithmic performance on classes of problems is more desirable than good performance on special cases (Barr et al. 1995). Woodward and Swan accentuate the necessity of testing the algorithms on all problems of a class (Woodward and Swan 2010). This, however, means that problem classes need to be defined. We do not address classification of problems directly, but we do want to draw attention to the necessity of problem classification in order to facilitate metaheuristic classification and expedite comparative testing of metaheuristics in future publications. Typical criteria for sorting optimisation problems are continuous or discrete, unconstrained or constrained and no, one or many objectives. Furthermore, the dimensionality, structure and parametric distributions of the problem can influence the performance of the metaheuristic (Barr et al. 1995).

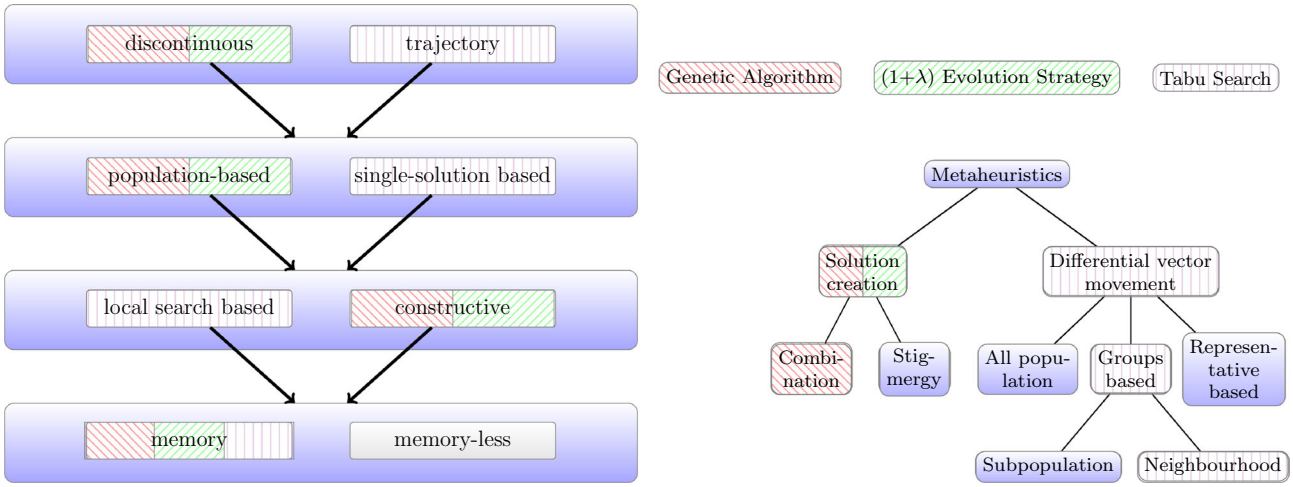

Fig. 5 Example classification of GA, $(1 + \lambda)$-ES and TS: *Structure* and *Behaviour*

Table 3 Example classification of GA, $(1 + \lambda)$-ES and TS: *Search* and *Algorithm*

Classification level	Genetic algorithm	$(1 + \lambda)$ Evolution strategy	Tabu search
Search	Mutation: I&D OG-R	Mutation: I&D OG-R	Tabu list: I&D OG-NOG
	Recombination: I&D OG-NOG-R		Aspiration criterion: I&D OG-R
	Selection: I&D OG-NOG-R		
Algorithm	Mutation: neighbourhood search	Mutation: neighbourhood search	Tabu list: adaptive memory programming
	Recombination: population-based search, intermediate search		Aspiration criterion: hill climbing, accepting negative moves
	Selection: population-based search		
	Population usage: population-based search, adaptive memory programming, multi-start	Population usage: population-based search	
	Generational replacement: hill climbing	Generational replacement: hill climbing	

6 Classification example

We illustrate the application of the proposed classification system with an example for three metaheuristics in their basic versions: the *Genetic Algorithm*, the $(1 + \lambda)$ *Evolution Strategy* and *Tabu Search*. After giving a short description, we will classify each of them according to the criteria of the classification levels *Structure*, *Behaviour*, *Search* and *Algorithm*. These four levels suffice to divert the frameworks into distinct classes. Fig. 5 and Table 3 illustrate the differences in each level. Criteria from the other levels are omitted as they are to some extent implementation-dependent and require a very detailed analysis of the metaheuristics and extensive experimental results, which is not in the scope of this primary example demonstrating a proof of concept.

Genetic Algorithm: The Genetic Algorithm (GA) was chosen for this example as it represents one of the most intensively studied and applied metaheuristics. Though there are many adaptations and extensions (García-Martínez et al. 2016), we will focus on a basic version suited for combinatorial optimisation problems. This basic GA generates its initial population randomly in the search space. It will repeat the following steps until a stopping criterion (in this case, a number of evaluations) is met:

1. Parents are selected utilising tournament-selection.
2. Offspring is generated via uniform crossover.
3. Offspring is mutated by flipping one random bit in the solution.
4. Offspring is evaluated and generational replacement takes place considering elitist selection.

The GA uses a discontinuous population-based structure where it constructs solutions using memory. New solutions are created by combining previously found solutions (cf. Fig. 5). The GA employs three different search strategies (cf. Table 3), where both recombination and selection are used for intensification and diversification while implementation (or self-adaptation) can determine the degree to which they are objective function or otherwise guided or random. The mutation operator enables I&D and searches randomly guided by an objective function. On the algorithmic level, mutation employs neighbourhood search, recombination uses intermediate population-based search and selection performs population-based search. The population is used for search while utilizing adaptive memory programming and multi-starts. Generational replacement follows a hill climbing scheme with regards to the fitness landscape.

$(1+\lambda)$ *Evolution Strategy*: The $(1+\lambda)$ Evolution Strategy (ES) is a simple form of ES and was selected as it belongs to the group of Evolutionary Algorithms, as does the GA (Emmerich et al. 2018; Corne and Lones 2018). We consider again a version suitable for combinatorial optimisation. In the $(1+\lambda)$-ES, only one parent is randomly initialised. The following steps are repeated, until a stopping criterion (again, the number of evaluations) is met:

1. λ offspring are generated by mutation of the parent (random bit flip).
2. Offspring is evaluated and the best individual of parent and offspring is selected as parent of the next generation.

Like the GA, $(1+\lambda)$-ES uses a discontinuous population-based structure where it constructs solutions using memory. New solutions are created by adapting previously found solutions (cf. Fig. 5). We tend to disagree with the classification as combination based solution creation as proposed by Molina et al. (2020) in that $(1+\lambda)$-ES does not show combinatory properties. We would like to note that this is, however, correctly classifying the $(\mu+\lambda)$-ES. To accommodate the $(1+\lambda)$-ES we relax the requirement that all metaheuristics have to be subdivided to fit into a leaf node in our behaviour classification. As there is only one parent, the $(1+\lambda)$-ES neither utilises parent selection nor crossover, limiting search to the mutation operator (cf. Table 3), which enables I&D and operates based on an objective function and randomness. On the algorithmic level, the population and mutation operator are used for population-based search and generational replacement follow a gradient ascent in the fitness landscape.

Tabu Search: Tabu Search (TS) coined the term metaheuristic and is one of the few approaches not considered nature-inspired (Glover and Laguna 1997; Laguna 2017). For this example we describe a basic version suitable for combinatorial optimisation that includes a short-term memory as tabu list and improved-best as aspiration criterion (Glover 1990; Glover and Laguna 1997). It starts at a randomly initialised solution, for which a list of candidate moves is created, i.e. neighbouring solutions. The tabu list is initialised with the starting solution and each visited solution is added. Candidates are evaluated and their respective tabu status is verified:

1. If the candidate is not tabu and there is no better candidate, it is chosen as best admissible move and the list of candidate moves and the tabu list are updated.
2. If the candidate is tabu and there is no better candidate, the aspiration criterion determines its admissibility or if the candidate list has to be extended. Both lists are then updated.

This is repeated until the stopping criterion, for this example a number of function evaluations, is met. TS is a single-solution trajectory-based metaheuristic that utilises local search and memory. Its behaviour is based on a neighbourhood guided differential vector movement (cf. Fig. 5). The aspiration criterion controls I&D in search in conjunction with the tabu list. It is guided by the objective function and randomness, while the tabu list can be guided by both the objective function and other metrics while being deterministic. On the algorithmic level, tabu list uses adaptive memory programming and the aspiration criterion can be described as hill climbing in the fitness landscape, although it is accepting negative moves (cf. Table 3).

Altogether, this more complex classification system enables a more detailed comparison and analysis of metaheuristics. Primary questions on the classified algorithms can be answered without having to research them in detail. For example, if the metaheuristics intensification and diversification capabilities have to be determined to get a first impression of its behaviour on multi-modal objective functions, the classification system shows that the GA provides more components to balance the search than the others and thus is probably advantageous. With the incorporation of the *Evaluation* level, this assessment can be further improved.

7 Conclusion

Many publications in metaheuristic research are focussed on the presentation of novel metaphor-inspired approaches. Other areas of research necessary for a holistic understanding are neglected due to the excessive publication of these ostensibly novel metaheuristics and their competitive evaluation. Additionally, it is increasingly difficult to maintain an overview of all existing metaheuristics, their

capabilities and their advantages or disadvantages. A formal classification strategy would prove beneficial. It provides the means to gain this overview, facilitate the integration of novel approaches into this overall system and help to identify differences between metaheuristics and the features that cause them. Furthermore, it enables a fast and directed selection of metaheuristics for comparisons and evaluations, as it contains information on their features and their applicability on specific problems.

There have been several approaches to the classification of metaheuristics. They vary strongly in their extensiveness, their criteria selection and their applicability on general metaheuristic frameworks and specialised metaheuristic algorithms. There is, however, no standardised procedure and common scheme for the task of classification. This leads to the problem that most approaches are rather incomplete and can only be applied with a specific classification goal in mind, not to a comprehensive classification of all metaheuristics. Only recently, Molina et al. (2020) presented a more detailed taxonomy, focussing on the overall behaviour of the metaheuristics. While this is most helpful for finding criteria to classify metaheuristics according to their general behaviour, it is still necessary to analyse them in detail when the cause of this behaviour is sought or if suitable metaheuristics for a comparison are wanted.

Thus, we propose a multi-level classification system capable of classifying frameworks as well as algorithms and applicable to present a holistic classification. Initially we defined the term classification for our purpose and made the necessary distinction to the similar but less informative term categorisation. We then summarised metaheuristic classification approaches in literature and discussed their application domain and their included criteria. Furthermore, we gave an overview of open research questions on metaheuristics, especially on the handling of novel approaches. We then presented a multi-level system suitable for different classification tasks. It consists of seven levels, each referring to different characteristics of a metaheuristic and containing the respective criteria. The basic structures of the individual levels and their criteria are based on the combination of existing classification schemes. On some levels, however, the criteria do not suffice for a comprehensive classification. We elaborated on these problems and presented procedures necessary to fill these gaps.

Though not yet complete, our classification system provides a basis for a comprehensive classification of metaheuristics. It can assist in ordering existing metaheuristics, in the integration of novel approaches and be utilised to identify characteristic features. However, it is vital to find additional criteria to complete the system. Those criteria have to be commonly accepted so classification can become a general step in the design and integration of novel metaheuristics and in performance or conceptual comparisons between different approaches.

Funding Open Access funding enabled and organized by Projekt DEAL. The funding was partially provided by the Bundesministerium für Wirtschaft und Energie.

References

Adl SM, Simpson AGB, Lane CE, Lukeš J, Bass D, Bowser SS, Brown MW et al (2012) The revised classification of eukaryotes. J Eukaryot Microbiol 59(5):429–514. https://doi.org/10.1111/j.1550-7408.2012.00644.x

Adl SM, Bass D, Lane CE, Lukeš J, Schoch CL, Smirnov A, Agatha S et al (2018) Revisions to the classification, nomenclature, and diversity of eukaryotes. J Eukaryot Microbiol. https://doi.org/10.1111/jeu.12691

Al-Amry RA, Al-Gaphari G (2018) Survey on recent bio-inspired optimization algorithms. Int J Comput Sci Netw (IJCSN) 7(6)

Almufti SM, Marqas RB, Saeed VA (2019) Taxonomy of bio-inspired optimization algorithms. J Adv Comput Sci Technol 8(2):23. https://doi.org/10.14419/jacst.v8i2.29402

Barr RS, Golden BL, Kelly JP, Resende MGC, Stewart WR (1995) Designing and reporting on computational experiments with heuristic methods. J Heuristics 1(1):9–32. https://doi.org/10.1007/bf02430363

Binitha S, Sathya SS (2012) A survey of bio inspired optimization algorithms. Int J Soft Comput Eng (IJSCE) 2(2):137–151

Birattari M, Paquete L, Stützle T, Varrentrapp K (2001) Classification of metaheuristics and design of experiments for the analysis of components. Tech. rep., Tech. Rep. AIDA-01-05

Blum C, Roli A (2003) Metaheuristics in combinatorial optimization. ACM Comput Surv 35(3):268–308. https://doi.org/10.1145/937503.937505

Can U, Alatas B (2017) Performance comparisons of current metaheuristic algorithms on unconstrained optimization problems. Period Eng Nat Sci (PEN). https://doi.org/10.21533/pen.v5i3.120

Corne D, Lones MA (2018) Evolutionary algorithms. Springer, Berlin, pp 1–22. https://doi.org/10.1007/978-3-319-07153-4_27-1

Dean A, Voss D, Draguljić D (2017) Design and analysis of experiments. Springer, Berlin. https://doi.org/10.1007/978-3-319-52250-0

Del Ser J, Osaba E, Molina D, Yang XS, Salcedo-Sanz S, Camacho D, Das S, Suganthan PN, Coello Coello CA, Herrera F (2019) Bio-inspired computation: where we stand and what's next. Swarm Evol Comput 48:220–250. https://doi.org/10.1016/j.swevo.2019.04.008

Dokeroglu T, Sevinc E, Kucukyilmaz T, Cosar A (2019) A survey on new generation metaheuristic algorithms. Comput Ind Eng 137:106040. https://doi.org/10.1016/j.cie.2019.106040

Eftimov T, Korošec P, Seljak BK (2017) A novel approach to statistical comparison of meta-heuristic stochastic optimization algorithms using deep statistics. Inf Sci 417:186–215. https://doi.org/10.1016/j.ins.2017.07.015

Emmerich M, Shir OM, Wang H (2018) Evolution strategies. Springer, Berlin, pp 1–31. https://doi.org/10.1007/978-3-319-07153-4_13-1

Fausto F, Reyna-Orta A, Cuevas E, Andrade ÁG, Perez-Cisneros M (2019) From ants to whales: metaheuristics for all tastes. Artif Intell Rev 53(1):753–810. https://doi.org/10.1007/s10462-018-09676-2

Fister IJ, Yang XS, Fister I, Brest J, Fister D (2013) A brief review of nature-inspired algorithms for optimization. Elektrotehniški vestnik 80(3). arxiv:1307.4186v1

García S, Fernández A, Luengo J, Herrera F (2010) Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: experimental analysis of power. Inf Sci 180(10):2044–2064. https://doi.org/10.1016/j.ins.2009.12.010

García-Martínez C, Rodriguez FJ, Lozano M (2016) Genetic algorithms. Springer, Berlin, pp 1–34. https://doi.org/10.1007/978-3-319-07153-4_28-1

Glover F (1990) Tabu search: a tutorial. Interfaces 20(4):74–94. https://doi.org/10.1287/inte.20.4.74

Glover F, Laguna M (1997) Tabu search. Springer, Berlin. https://doi.org/10.1007/978-1-4615-6089-0

Hong L, Drake JH, Woodward JR, Özcan E (2018) A hyper-heuristic approach to automated generation of mutation operators for evolutionary programming. Appl Soft Comput 62:162–175. https://doi.org/10.1016/j.asoc.2017.10.002

Hooker JN (1994) Needed: an empirical science of algorithms. Oper Res 42(2):201–212. https://doi.org/10.1287/opre.42.2.201

Hooker JN (1995) Testing heuristics: we have it all wrong. J Heuristics 1(1):33–42. https://doi.org/10.1007/bf02430364

Jacob E (2004) classification and categorization: a difference that makes a difference. Library Trends 52

Krawiec K, Simons C, Swan J, Woodward JR (2018) Metaheuristic design patterns: new perspectives for larger-scale search architectures. IGI Global, pp 1–36

Krishnanand KR, Nayak SK, Panigrahi BK, Rout PK (2009) Comparative study of five bio-inspired evolutionary optimization techniques. In: 2009 World congress on nature & biologically inspired computing (NaBIC). IEEE. https://doi.org/10.1109/nabic.2009.5393750

Laguna M (2017) Tabu search. Springer, Berlin, pp 1–18. https://doi.org/10.1007/978-3-319-07153-4_24-1

LaTorre A, Molina D, Osaba E, Del Ser J, Herrera F (2020) Fairness in bio-inspired optimization research: a prescription of methodological guidelines for comparing meta-heuristics. To be published arxiv:2004.09969v1

Li H, Liu X, Huang Z, Zeng C, Zou P, Chu Z, Yi J (2020) Newly emerging nature-inspired optimization—algorithm review, unified framework, evaluation, and behavioural parameter optimization. IEEE Access 8:72620–72649. https://doi.org/10.1109/access.2020.2987689

Lones MA (2014) Metaheuristics in nature-inspired algorithms. In: Proceedings of the 2014 conference companion on Genetic and evolutionary computation companion—GECCO Comp'14. ACM Press. https://doi.org/10.1145/2598394.2609841

Lones MA (2019) Mitigating metaphors: a comprehensible guide to recent nature-inspired algorithms. SN Comput Sci 1(49). https://doi.org/10.1007/s42979-019-0050-8. arxiv:1902.08001v1

Molina D, Poyatos J, Del Ser J, García S, Hussain A, Herrera F (2020) Comprehensive taxonomies of nature- and bio-inspired optimization: inspiration versus algorithmic behavior. Cogn Comput Crit Anal Recomm. https://doi.org/10.1007/s12559-020-09730-8

Nabaei A, Hamian M, Parsaei MR, Safdari R, Samad-Soltani T, Zarrabi H, Ghassemi A (2016) Topologies and performance of intelligent algorithms: a comprehensive review. Artif Intell Rev 49(1):79–103. https://doi.org/10.1007/s10462-016-9517-3

Nesmachnow S (2014) An overview of metaheuristics: accurate and efficient methods for optimisation. Int J Metaheuristics 3(4):320. https://doi.org/10.1504/ijmheur.2014.068914

Pazhaniraja N, Paul PV, Roja G, Shanmugapriya K, Sonali B (2017) A study on recent bio-inspired optimization algorithms. In: 2017 Fourth international conference on signal processing, communication and networking (ICSCN). IEEE. https://doi.org/10.1109/icscn.2017.8085674

Rajpurohit J, Sharma TK, Abraham A Vaishali (2017) Glossary of metaheuristic algorithms. Int J Comput Inf Syst Ind Manag Appl :181–205

Selvaraj C, Kumar S, Karnan M (2014) A survey on application of bio-inspired algorithms. Int J Comput Sci Inf Technol (IJCSIT) 5(1):366–370

Sergeyev YD, Kvasov DE, Mukhametzhanov MS (2018) On the efficiency of nature-inspired metaheuristics in expensive global optimization with limited budget. Sci Rep 8(1). https://doi.org/10.1038/s41598-017-18940-4

Sergienko IV, Hulianytskyi LF, Sirenko SI (2009) Classification of applied methods of combinatorial optimization. Cybern Syst Anal 45(5):732–741. https://doi.org/10.1007/s10559-009-9134-0

Sörensen K, Sevaux M, Glover F (2017) A history of metaheuristics. Handbook of heuristics. arxiv:1704.00853v1

Sörensen K (2013) Metaheuristics-the metaphor exposed. Int Trans Oper Res 22(1):3–18. https://doi.org/10.1111/itor.12001

Sörensen K, Glover FW (2013) Metaheuristics. Encyclopedia of operations research and management science. Springer, New York, pp 960–970

Stevens PF (1984) Metaphors and typology in the development of botanical systematics 1690–1960, or the art of putting new wine in old bottles. Taxon 33(2):169–211. https://doi.org/10.2307/1221161

Stevens PF (2003) History of taxonomy. American Cancer Society. https://doi.org/10.1038/npg.els.0003093

Swan J, Adriaensen S, Bishr M, Burke EK, Clark JA, Causmaecker PD, Durillo J, et al (2015) A research agenda for metaheuristic standardization. In: MIC 2015 : the XI metaheuristics international conference

Wolpert DH, Macready WG (1997) No free lunch theorems for optimization. IEEE Trans Evol Comput 1(1):67–82. https://doi.org/10.1109/4235.585893

Woodward JR, Swan J (2010) Why classifying search algorithms is essential. In: 2010 IEEE international conference on progress in informatics and computing, IEEE