COHDA: A Combinatorial Optimization Heuristic for Distributed Agents

Christian Hinrichs[1], Sebastian Lehnhoff[2], and Michael Sonnenschein[1]

[1] University of Oldenburg, Germany,
christian.hinrichs@uni-oldenburg.de
sonnenschein@informatik.uni-oldenburg.de
[2] OFFIS Institute for Information Technology, Oldenburg, Germany,
sebastian.lehnhoff@offis.de

Abstract. Solving Distributed Constraint Optimization Problems has a large significance in today's interconnected world. Complete as well as approximate algorithms have been discussed in the relevant literature. However, these are unfeasible if high-arity constraints are present (i. e., a fully connected constraint graph). This is the case in distributed combinatorial problems, for example in the provisioning of active power in the domain of electrical energy generation. The aim of this paper is to give a detailed formalization and evaluation of the COHDA heuristic for solving these types of problems. The heuristic uses self-organizing mechanisms to optimize a common global objective in a fully decentralized manner. We show that COHDA is a very efficient decentralized heuristic that is able to tackle a distributed combinatorial problem, without being dependent on centrally gathered knowledge.

Keywords: Self-Organization, Cooperation, Smart Grid.

1 Introduction

In decentralized systems, where the search space of a given optimization problem is distributed into disjoint subspaces, centralized optimization approaches often cannot be applied. For example, the global collection of data might violate privacy considerations or bandwidth restrictions. The gathering of such data might even be impossible, as it is the case if local search spaces are partially unknown or cannot be enumerated (i. e. due to infiniteness). Another limitation is that distributed search spaces are often not independent. Such interdependencies require to evaluate search spaces with relation to each other. Thus, a parallel search for optimal solutions would require a large communication overhead.

For instance, this type of problem is present in the transition of today's electricity grid to a decentralized smart grid. Here, we have to cope with an increasing number of distributed energy resources (DER). Usually, the operation of these DER is individually configured, according to the constraints given at the place of installation. For example, a combined heat and power plant (CHP) primarily has to satisfy the (varying) thermal energy needs of a building. Electrical

J. Filipe and A. Fred (Eds.): Agents and Artificial Intelligence, ICAART 2013, CCIS 449, DOI: 10.1007/978-3-662-44440-5_2, pp. 1–17, 2014.

energy is produced only as a side-product, so that e. g. the provisioning of active power by this unit is difficult to request directly. Because of this rather dynamical behavior of such DER, an adaptive, decentralized control scheme is crucial for a reliable operation of the system (see [17,22] for more details regarding ongoing work in this domain).

In the contribution at hand, we focus on day-ahead planning of the provisioning of active power, which can be expressed as a distributed combinatorial problem: Given a set of DER and a global target power profile, each unit has to select its own mode of operation for the planning horizon in such a way, that the resulting individual power profiles of all units jointly match the global target profile as close as possible. For this purpose, the distributed heuristic COHDA is developed which makes use of self-organization strategies. In hitherto existing population-based heuristics, each individual represents a complete solution to the given problem within a common search space. In our approach, however, an individual incorporates a local, dependent search space, and thus defines a *partial* solution that can only be evaluated with respect to all other individuals. The task of each individual is to find a partial local solution that, if combined with the local solutions of the other individuals, yields the optimal global solution.

The problem stated in this contribution is formulated as an instance of a *distributed constraint optimization problem* (DCOP). In [4], a thorough examination of heuristic approaches from the DCOP domain as well as from the context of game theory is presented, and [12] puts DCOPs into perspective of cooperative problem solving in multi-agent systems. All mentioned approaches therein rely on communication between individuals which are directly connected in the constraint graph of the problem to solve. However, the problem considered in the present contribution induces a fully connected constraint graph, which renders these approaches unfeasible due to communication complexity. The consequence would be a broadcasting of messages, which has been realized in the COBB approach [18]. Technically, broadcasting can be done in two ways: by sending messages directly to all other existing individuals, or by using a central black board where the relevant information is posted publicly. The former method would lead to an explosion of the number of transferred messages. The latter method, as used in [14], is able to avoid this (if a suitable *adjustment schedule* is used, c. f. [4]), but introduces the problem of a centralized information repository with the drawbacks mentioned earlier. In contrast, the EPOS approach, as introduced in [20,19], uses a *tree overlay* organization structure and thus is based on a partial representation of the constraint graph. Following, EPOS does fulfill the demand for an algorithm that can handle fully connected constraint graphs in the context of distributed constraint optimization problems. The tree overlay, however, imposes hierarchical relations on the agents. Hence, there are still centralized components present in the architecture of this approach. Furthermore, the optimization process is carried out in an iterative bottom-up fashion, which leads to a rather synchronous execution paradigm. We believe that a more convenient approach with regard to decentralized settings is possible. Therefore, we introduced the COHDA heuristic in [7].

In order to compensate for the condensed presentation in [7], the aim of this paper is to evaluate the COHDA heuristic in more detail. Hence, we will give a detailed formalization of the heuristic and a thorough description of the approach first. In addition, we extend COHDA to the multi-objective case. Subsequently, we evaluate the heuristic with respect to different parameters: message delay, network density, planning horizon, search space complexity and population size. Note that the first two are user-defined parameters, while the last three are problem-specific. From the results, the properties adaptivity, robustness, scalability, and anytime behavior are derived.

The contribution at hand is a revised version of [8].

2 Method

Constraint optimization problems (COP) can be formulated with an integer programming model, if we assume that each search space is discrete by nature, and that the elements within are known and may be enumerated. Let $c \in \mathbb{R}^q$ be the *target* that should be matched, where q is the number of dimensions (i. e. in the context of power provisioning, q denotes the planning horizon). We now assume that there are m disjoint search spaces (i. e. electrical generators). Now let the ith search space contain n_i elements. The jth element in such a search space describes a partial solution to the problem, denoted with $w_{ij} \in \mathbb{R}^q$ (i. e. a feasible power profile). The goal is to *select* an element w_{ij} from each search space i, such that the sum of these selected values approaches c as close as possible (c. f. [7]):

$$\min \left\| c - \sum_{i=1}^{m} \sum_{j=1}^{n_i} (w_{ij} \cdot x_{ij}) \right\|_1 \tag{1}$$

$$\text{subject to } \sum_{j=1}^{n_i} x_{ij} = 1,\ i = 1 \dots m,$$

$$x_{ij} \in \{0, 1\},\ i = 1 \dots m,\ j = 1 \dots n_i,$$

Here, each search space has an associated selection variable x_{ij}, which defines whether an element has been chosen ($x_{ij} = 1$) or not ($x_{ij} = 0$). This model is a generalization of the well-known subset-sum problem, where solutions exceeding the target are not allowed (whereas our model approximates c from any side).

2.1 Mapping to a Distributed System

So far, the above formulation is only suitable from a central perspective. In the aimed setting however, each search space is represented by an autonomous decision maker, which we call *agents*. The task of each agent a_i is to select one of its elements w_{ij} with respect to the common global target c. More formally, an agent a_i has to find an assignment of *its own* selection variables x_{ij}, such that the objective function in (1) is minimized globally.

Definition 1. *A selection of an agent a_i is a tuple $\gamma_i = \langle i, j \rangle$ where i is the identifier of a_i, and j identifies the selected element w_{ij} such that $x_{ij} = 1$ and $\sum_{j=1}^{n_i} x_{ij} = 1$.*

Agents are autonomous, so they may change their selection at any time. Therefore, we need to define the *state* of an agent.

Definition 2. *The state of an agent a_i is given by $\sigma_i = \langle \gamma_i, \lambda_i \rangle$, where γ_i is a selection containing an assignment of a_i's decision variables x_{ij}, and λ_i is a unique number within the history of a_i's states. Each time an agent a_i changes its current selection γ_i to $\acute{\gamma}_i$, the agent enters a new state $\acute{\sigma}_i = \langle \acute{\gamma}_i, \lambda_i' \rangle$ where $\lambda_i' = \lambda_i + 1$. This imposes a strict total order on a_i's selections, hence λ_i reflects the "age" of a selection.*

In order to decide which of its local elements w_{ij} to select optimally, an agent has to take the current selections (i.e. states) of the other agents in the system into account.

Definition 3. *A configuration $\Sigma = \{\sigma_i, \sigma_k, \dots\}$ is a set of states. A state belonging to an agent a_i can appear in a configuration no more than once:*

$$\sigma_i \in \Sigma \ \wedge \ \sigma_k \in \Sigma \ \Rightarrow \ i \neq k$$

Note that this definition allows a configuration to be incomplete with regard to the population of agents in the system. A configuration that contains states for all existing agents is called *global*:

Definition 4. *A global configuration regarding the whole system is denoted by $\Sigma_{global} = \{\sigma_i \mid i = 1 \dots m\}$.*

On the other hand, Definition 3 enables us to model a local view that an agent a_i has on the system. This is quite similar to the definition of *context* in [16]. We call such a local view a *perceived configuration*:

Definition 5. *A perceived configuration of an agent a_i is a configuration $\Sigma_i = \{\sigma_k \mid a_i$ is aware of $a_k\}$.*

Following, if we assume that an agent a_i is able to somehow perceive a configuration Σ_i containing information about other agents that a_i is aware of (we will address this later), it may now select one of its own elements w_{ij} with respect to the currently chosen elements of other agents in Σ_i and the optimization goal c.

2.2 Introducing Local Constraints

Furthermore, we introduce local constraints, which impose a penalty value p_{ij} to each element w_{ij} within the search space of an agent a_i.

Definition 6. *The penalty function $\Pi_i : \mathbb{R}^q \mapsto \mathbb{R}$ of an agent a_i maps an element w_{ij} to a penalty value p_{ij}.*

These local constraints are known to the corresponding agent only, as described in the introductory example (i. e. for a CHP unit, heating preferences defined by residents). Thus, each agent has two objectives: minimizing the common objective function as given in (1), and minimizing its local penalties that are induced by contributing a certain element w_{ij}. This compound optimization goal at agent level may be expressed with a utility function:

$$z_i = \alpha_i \cdot z_i^1 + (1 - \alpha_i) \cdot z_i^2 \tag{2}$$

Here, z_i^1 represents the common global objective function and z_i^2 incorporates the local constraints. The parameter α_i allows an agent a_i to autonomously adjust its preference for optimizing the global goal versus optimizing its local constraints. Note that the domains of z_i^1 and z_i^2 must be carefully defined in this model (i. e. normalized to $[0.0,\ 1.0]$), so that α_i gains the desired effect.

From a global point of view, this yields the *distributed-objective multiple-choice combinatorial optimization problem* (DO-MC-COP):

$$\begin{aligned}
\min\ & \sum_{i=1}^{m} z_i && (3)\\
\text{where}\ & z_i = \alpha_i \cdot z_i^1 + (1 - \alpha_i) \cdot z_i^2\ , \\
& z_i^1 = \left\| c - \left(\sum_{j=1}^{n_i} (w_{ij} \cdot x_{ij}) + \sum_{w \in \phi(\Sigma_i)} w \right) \right\|_1\ , \\
& z_i^2 = \sum_{j=1}^{n_i} \Pi_i (w_{ij}) \cdot x_{ij}\ , \\
& \phi(\Sigma) = \{ w_{ij} \mid \langle \langle i, j \rangle, \lambda \rangle \in \Sigma \}\ , \\
\text{subject to}\ & \sum_{j=1}^{n_i} x_{ij} = 1,\ i = 1 \ldots m\ , \\
& x_{ij} \in \{0, 1\},\ i = 1 \ldots m,\ j = 1 \ldots n_i\ , \\
& \alpha_i \in \mathbb{R},\ 0 \leq \alpha_i \leq 1,\ i = 1 \ldots m\ .
\end{aligned}$$

Summarizing, in this model there are m decision makers (agents) a_i, that pursue a common goal by each contributing one solution element w_{ij} from their associated local search space, while at the same time minimizing the resulting local penalty $\Pi_i (w_{ij})$.

Obviously, if an agent a_i changes its state σ_i, this should have an effect on the decision making of the other agents in the system. Thus, the definition of how an arbitrary agent a_k perceives a configuration Σ_k, and how this relates to Σ_{global}, is crucial for solving the DO-MC-COP in a distributed way. The following section addresses these questions and describes a self-organizing approach to this distributed-objective problem.

2.3 COHDA

In nature, we find many examples of highly efficient systems, which perform tasks in a completely decentralized manner: swarming behavior of schooling fish or flocking birds [23], foraging of ants [9] and nest thermoregulation of bees [10]. Even processes within single organisms show such astonishing behavior, for instance the neurological development of the fruit fly [13] or the foraging of *Physarum polycephalum*, a single-celled slime mold [26], which both exhibit rules for adaptive network design. One of the core concepts in these examples is self-organization. From the perspective of multi-agent systems, this term can be defined as *"the mechanism or the process enabling a system to change its organization without explicit external command during its execution time"* [24]. From the perspective of complex systems theory, this is related to emergence, which can be defined as *"properties of a system that are not present at the lower level [...], but are a product of the interactions of elements"* [5]. Such systems usually exhibit a number of desirable properties like adaptivity, robustness, scalability, and anytime behavior [21,2].

The COHDA heuristic, as originally proposed in [7], applies these concepts to create a self-organizing heuristic for solving distributed combinatorial problems. The key concept in COHDA is a partial representation of the (usually fully connected) constraint graph of the problem to solve, in order to reduce coordination complexity. Note that this graph induces the communication network of the system. But unlike other approaches mentioned in the introduction, a specific graph topology is not required. Instead, the heuristic adapts to whatever topology is given by real-world requirements (i. e. physical communication lines in power grids), or is defined by the system operator. This is combined with an information spreading strategy that, despite the partial constraint graph, allows the heuristic to converge rapidly to a global solution.

As described above, the heuristic has to cope with an arbitrary communication topology. This can be expressed with a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where each agent is represented by a vertex $a_i \in \mathcal{V}$. Edges $e = (a_i, a_k) \in \mathcal{E}$ depict communication links. Thus, we can define the *neighborhood* of an agent:

Definition 7. *Given a set of edges $\mathcal{E}$, the neighborhood of an agent a_i is defined as $\mathcal{N}_i = \{a_k \mid (a_i, a_k) \in \mathcal{E}\}$.*

An agent may not communicate with any other agent outside of its neighborhood. Just like flocking birds, the agents now observe their local environment and react to changes within their perception range. For that purpose, each agent a_i maintains a configuration Σ_i, which reflects the knowledge of a_i about the system. This configuration is initially empty, but is updated during the iterative process through information exchange with other agents (hence, Σ_i is called the *perceived configuration* of a_i, see Definition 5). Now, whenever an agent a_i enters a new state $\acute{\sigma}_i$ by changing the assignment of its decision variables x_{ij}, its neighboring agents $a_k \in \mathcal{N}_i$ perceive this event. These agents now each update their current local view Σ_k on the system, and react to this event by re-evaluating their search spaces and subsequently adapting their own decision variables.

However, usually $\Sigma_k \neq \Sigma_{global}$, hence an agent has to deal with incomplete, local knowledge. Thus, for improving the local search at agent level, the COHDA heuristic uses an information spreading strategy besides this reactive adaptation. Whenever a local change is published to the neighborhood, the publishing agent a_i not only includes information about its updated state σ_i, but publishes its whole currently known perceived configuration Σ_i as well. A receiving agent a_k now updates its existing knowledge base Σ_k with this two-fold information $(\Sigma_i \cup \{\sigma_i\})$. In this update procedure, an element $\sigma_y = \langle \gamma_y, \lambda_y \rangle \in \Sigma_i$ of the sending agent a_i is added to Σ_k of the receiving agent a_k if and only if any of the following conditions hold:

1. Σ_k does not already contain a state from a_y, such that $\forall \sigma_z \in \Sigma_i \,:\, z \neq y$.
2. Σ_k already contains a state σ_z with $z = y$, but σ_z has a lower value λ_z, such that $\exists \sigma_z = \langle \gamma_z, \lambda_z \rangle \in \Sigma_i \,:\, z = y \,\wedge\, \lambda_z < \lambda_y$. This means, σ_z is outdated (see Definition 2), and hence σ_y replaces σ_z in Σ_k.

Using this information spreading strategy, agents build a complete representation Σ_{global} of the whole system over time, and take this information into account in their decision making as well. However, due to possibly rather long communication paths between any two agents, these global views on the system are likely to be outdated as soon as they are built and represent *beliefs* about the systems rather than facts. Nevertheless, they provide a valuable guide in the search for optimal local decisions.

In order to ensure convergence and termination, a third information flow is established on top of that. In addition to the perceived configuration Σ_i (which reflects the currently known system configuration including the agent's own current state σ_i), each agent keeps track of the *best known configuration* Σ_i^* it has seen during the whole process so far.

Definition 8. *A configuration $\Sigma_i^* = \{\sigma_i^*, \sigma_k^*, \dots\}$ is an arbitrary snapshot of the system taken by an agent a_i.*

Definition 9. *The best Σ_i^* over all agents in the population is denoted by Σ^*.*

Whenever an agent updates its Σ_i by means of received information, it compares this new configuration Σ_i to Σ_i^*. If Σ_i yields a better solution quality than Σ_i^* according to DO-MC-COP (3), Σ_i is stored as new best known configuration Σ_i^*. In addition to σ_i and Σ_i, an agent a_i also exchanges its Σ_i^* with its neighbors, everytime it changes. Thus, when an agent a_k receives a Σ_i^* from a neighbor a_i, the agent replaces its currently stored Σ_k^* by Σ_i^*, if the latter yields a better solution quality than the former.

The whole process can be summarized in the following three steps:

1. (`update`) An agent a_i receives information from one of its neighbors and imports it into its own knowledge base. That is, its beliefs Σ_i about the current configuration of the system is updated, as well as the best known configuration Σ_i^*.

2. (`choose`) The agent now adapts its own decision variables x_{ij} according to the newly received information, while taking its own local objectives into account as well. If it is not able to improve the believed current system configuration Σ_i, the state σ_i^* stored in the currently best known configuration Σ_i^* will be taken. The latter causes a_i to revert its current state σ_i to a previous state σ_i^*, that once yielded a better believed global solution.
3. (`publish`) Finally, the agent publishes its belief about the current system configuration Σ_i (including its own new state $\acute{\sigma}_i$), as well as the best known configuration Σ_i^* to its neighbors. Local objectives are not published to other agents, thus maintaining privacy.

Accordingly, an agent a_i has two behavioral options after receiving data from a neighbor. First, a_i will try to improve the currently believed system configuration Σ_i by choosing an appropriate w_{ij}, and subsequently adding its new state $\acute{\sigma}_i$ to Σ_i. Yet, this only happens if the resulting Σ_i would yield a better solution quality than Σ_i^*. In that case, Σ_i replaces Σ_i^*, so that they are identical afterwards. If the agent cannot improve Σ_i over Σ_i^*, however, the agent reverts its state to the one stored in Σ_i^*. This state, σ_i^*, is then added to Σ_i afterwards. Thus, Σ_i always reflects the current view of a_i on the system, while Σ_i^* always represents the currently pursued goal of a_i, since it is the best configuration the agent knows. In either case, Σ_i and Σ_i^* both contain a_i's current state after Step 2.

As can be seen from the above description, the COHDA heuristic is inherently *adaptive*: the agents permanently adapt to changes in their environment; for more details on this see [6]. Also, since an overall best configuration Σ^* (Definition 9) can be identified at any point in time, which is replaced only when an even better configuration is found, the heuristic exhibits the *anytime behavior* [2]. In order to reveal the properties *scalability* and *robustness*, we performed a simulation-based evaluation. This evaluation will be discussed in the following sections.

2.4 Implementation

We implemented the proposed heuristic COHDA in a multi-agent system (MAS). In our simulation environment, agents communicate asynchronously, using a network layer as communication backend. This backend may be a physical one, so as to be able to distribute the MAS over arbitrary machines. In our evaluation however, we used a simulated network layer, in order to have full control over message travelling times, and to permit deterministic repetitions of simulation runs. For this, we used predefined seeds for the random number generators. This allows us to simulate unsteady communication layers with varying message delays. Technically, our simulation is event-driven, i. e., agents react to events (messages from other agents) in the continuous time domain, which is induced by the above mentioned varying message delays. For the ease of evaluation, however, the simulation status is reported to the experimenter exactly every integer-valued time step. Following, from the user perspective, a discrete-time simulation is performed. Our implementation ensured that we were able to monitor (and count) all exchanged messages.

In the conducted experiments, each agent represents a simulated combined heat and power (CHP) device with an $800\,l$ thermal buffer store. We used the simulation model of an EcoPower CHP as described in [3]. For each of those devices, the thermal demand for a four-family house during winter was simulated according to [11]. The devices were operated in heat driven operation and thus primarily had to compensate the simulated thermal demand. Additionally, after shutting down, a device would have to stay off for at least two hours. However, due to their thermal buffer store and the ability to modulate the electrical power output within the range of $[1.3\,kW,\ 4.7\,kW]$, the devices had still some degrees of freedom left.

For each conducted experiment, and for each agent, the simulation model has been instantiated with a random initial temperature level of the thermal buffer store and a randomly generated thermal demand. Subsequently, a number of feasible power profiles were generated from each of these simulation models. The resulting sets of power profiles are then used as local search spaces by the agents. The global goal c of the optimization problem was generated as a random electrical power profile, which was scaled to be feasible for the given population of CHP devices. However, we cannot guarantee that an optimal solution actually lies within in the set of randomly enumerated search spaces. The task of the agents now was to select one element out of their given sets of power profiles each, so that the sum of all selected power profiles approximates the target profile c as exactly as possible.

3 Results

As a first step, we examined the general behavior of the heuristic without penalties. In Fig. 1, the results of a single simulation run ($m = 30$ devices with $n = 2000$ possible power profiles each) are visualized. The planning horizon was set to four hours in 15-minute intervals. The upper chart shows the target profile (dashed line) and the resulting aggregated power output (solid line). The individual power output profiles of the devices are depicted in the lower chart. The latter is quite chaotic, which is due to the limited sets of available power output profiles per device. Nevertheless, the heuristic was able to select 30 profiles (one for each device), whose sum approximates the target profile with a remaining imbalance of less than $2.5\,kW$ per time step in the planning horizon.

In Fig. 2, the process of the heuristic for this simulation run is shown in detail. This data is visible to the simulation observer only, the individual agents still act upon local knowledge. The solid line depicts the global fitness value of the heuristic over time. This fitness represents the rating of the best configuration Σ^* existing in the population (see Definition 9), at each point in time, respectively. These values are determined according to (3), but have been normalized to the interval [0.0, 1.0], with 0.0 being the optimum. The normalization was done by taking an approximation for the worst combination of power profiles as upper

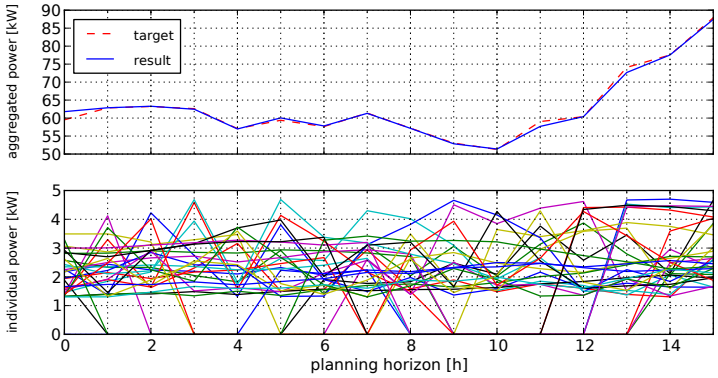


Fig. 1. Optimization result of a single simulation run with 30 CHP (and local search spaces comprising 2000 feasible power profiles each), for a planning horizon of four hours in 15-minute intervals.

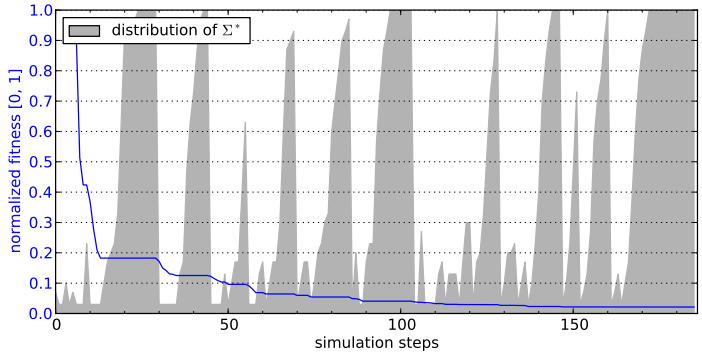


Fig. 2. Detailed illustration of the COHDA heuristic during a simulation.

bound:

$$d_{worst} = \max\left(d\left(c, \sum_{i=1}^{m} w_{i,min}\right),\ d\left(c, \sum_{i=1}^{m} w_{i,max}\right)\right)$$

(with $w_{i,min}$ and $w_{i,max}$ being elements of an agent a_i with minimal and maximal absolute cumulative value, respectively), and assuming the existence of an optimal solution (no remaining imbalance) as lower bound. In order to examine convergence, the agent population was parametrized with the upper bound as initial solution.

In general, the fitness value decreases over time (lower is better, so this means an improvement of the fitness) until it converges to a near-optimal solution. However, it is not strictly decreasing, since there are non-decreasing intervals. This is due to the information spreading strategy in COHDA, and can be explained with the distribution ratio of Σ^*. The latter is visualized by the shaded area

(the higher, the more agents are aware of the current Σ^*). Recall that an agent a_i inherits a received Σ_k^* from a neighbor a_k if Σ_k^* yields a better rating than the currently stored Σ_i^* of the agent a_i. Thus, a configuration with very good rating prevails and spreads in the network, until a better rated configuration is found somewhere. As an example, consider the situation at simulation step 12. Some agent, say a_i, has found a configuration Σ_i^* with a normalized fitness rating of ≈ 0.18. This configuration is, at that time, the best configuration found in the whole population, therefore $\Sigma^* = \Sigma_i^*$. The agent publishes Σ_i^* to its neighbors, who accept it as a new best configuration, and re-publish it again to their respective neighbors. Hence, the distribution of Σ^* (shaded area) rises in the following time steps, but the fitness value (solid line) remains constant. In simulation step 30, however, some agent a_k finds an even better configuration Σ_k^*. Thus the fitness value improves and, from this point in time, $\Sigma^* = \Sigma_k^*$. At the same time, the distribution of the (new) Σ^* drops dramatically, since this configuration is known to a_k only and has yet to be spread in the network. The heuristic terminates after 185 simulation steps, where a certain Σ^* has been distributed to all agents, and no better configuration can be found. The final fitness value is 0.02, which amounts to a total remaining imbalance of $7.09\,kW$ ($0.007\,\%$ of the targeted $1004.13\,kW$ in total over the planning horizon).

Figure 3 shows the aggregated behavior of the COHDA heuristic for 100 simulation runs. For each simulation run, the same CHP devices and thus the same

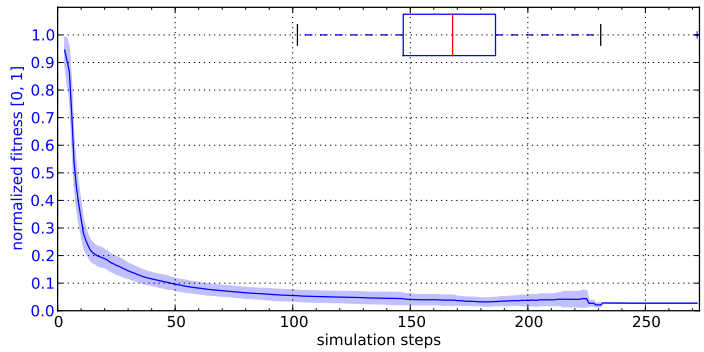


Fig. 3. Aggregated behavior of COHDA for 100 simulation runs with 30 CHP (and local search spaces comprising 2000 feasible power profiles each), for a planning horizon of four hours in 15-minute intervals.

local search spaces were used, but the communication network was initialized with different seeds for the random number generator. This yielded a different communication graph in each run, as well as different generated message delays. The solid line represents the mean fitness over time, while the shaded area around this line depicts the standard deviation. Obviously, in this example the COHDA heuristic is able to converge to near optimal solutions independently from the

underlying communication backend. On average over all 100 simulation runs, each agent sent 1.5 ± 0.04 messages per simulation step. The boxplot visualizes simulation lengths, with 169.69 ± 28.38 simulation steps being the mean.

3.1 Performance Criteria

Besides the inspection of the general behavior, simulation performance can be measured in terms of (a) the resulting fitness after termination, (b) the simulation length, or (c) the average number of exchanged messages per agent per simulation step during the process. In our experiments, the influence of different input parameters on each of these numbers (a-c) has been analyzed. From the resulting interactions, properties like robustness and scalability may be derived. If not stated otherwise, the experiments were conducted using a message delay $msg_{max} = 2$ (see Sect. 3.2), a small world network topology with $\phi = 2.0$ (see Sect. 3.3), a target comprising $q = 16$ dimensions (see Sect. 2.3), a population size of $m = 30$ agents, and no penalties (such that $\alpha_i = 1.0$ in (2)). Each examined scenario was simulated 100 times. Figure 4 shows a summary of our results. We will discuss each part in the following sections.

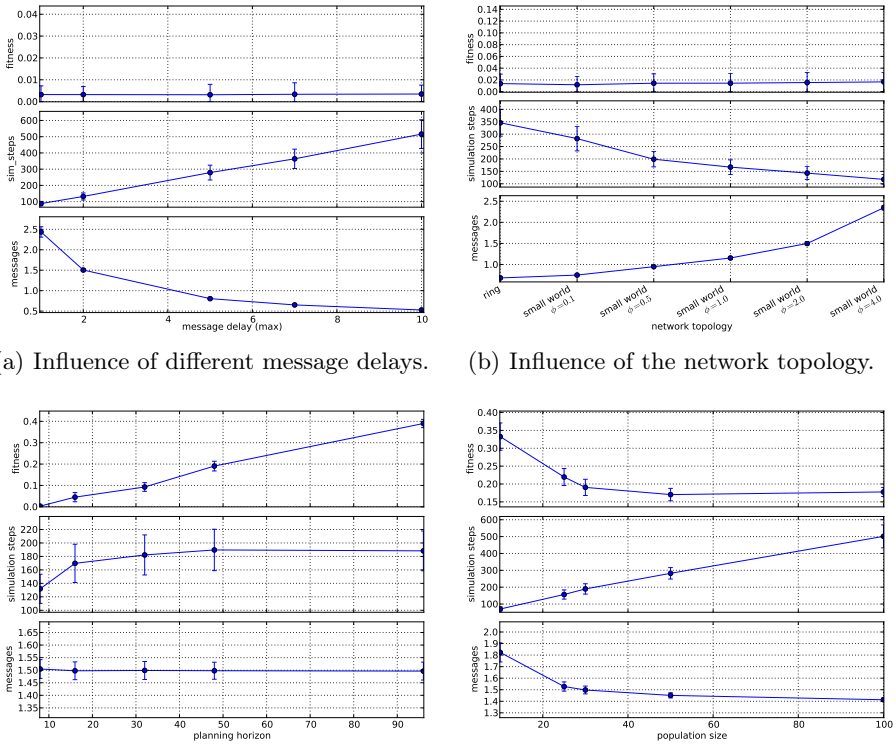
3.2 Message Delay

An important property of the simulated communication backend is its ability for delayed messages. In order to evaluate the robustness of the heuristic against a non-deterministic communication layer, we tested the approach with different amounts of message delays. To accomplish that, we defined an interval $[1,\ msg_{max}]$, from which a random number is generated for each sent message. The message is then delayed for the according number of simulation steps. We evaluated $msg_{max} \in \{1, 2, 5, 7, 10\}$.

Figure 4(a) shows the influence of message delays on the simulation performance, as defined in Sect. 3.1 (criteria a-c). Fortunately, message delays have absolutely no influence on the final fitness produced by the heuristic (criterion a, top chart). This means that COHDA is very stable against an unsteady communication network. The time until termination (criterion b, middle chart) consequentially rises linearly with increasing message delay. With regard to the amount of exchanged messages (criterion c, bottom chart), a strongly decreasing trend towards less than one sent message on average per agent per simulation step with increasing delay is visible. When multiplied with the number of simulation steps, the number of messages per agent throughout a whole simulation run can be determined (chart not shown here). We find a minimum of exchanged messages per simulation run with $msg_{max} = 2$. Following, COHDA does not only cope with, but even benefits from a slight variation at agent level introduced by message delays (for details on inter-agent variation see [1]).

3.3 Network Density

The composition of an agents' neighborhood is directly coupled to the underlying communication graph $\mathcal{G} = (\mathcal{E}, \mathcal{V})$. Preliminary experiments showed a beneficial



(a) Influence of different message delays.

(b) Influence of the network topology.

(c) Influence of the planning horizon.

(d) Influence of the population size.

Fig. 4. Performance analysis of COHDA regarding different input parameters.

impact of random graphs with a low diameter. Thus, we evaluated the following topologies:

- *Ring*: The agents are inserted into a ring-shaped list. Each agent is then connected to its predecessor and successor.
- *Small World*: This network comprises an ordered ring with $|\mathcal{V}| \cdot \phi$ additional connections between randomly selected agents, cf. [25]. We examined $\phi \in \{0.1,\ 0.5,\ 1.0,\ 2.0,\ 4.0\}$.

In Fig. 4(b), the results of these experiments are visualized. We ordered the plotted data according to the approximated average neighborhood size, which defines the overall density of the communication graph. Similar to the previous section, there is no influence of the network density on solution quality. Expectedly, the message complexity increases with larger neighborhoods. Similarly, simulation length decreases with more connections. As in the previous section, a trade-off between run-time in terms of simulation steps, and run-time in terms of exchanged messages is visible. A comparison of the number of messages per agent throughout a whole simulation run against network topology shows that, for the

given scenario, a small world topology with $\phi = 0.5$ yields the least messages on average during a whole simulation (chart not shown here).

3.4 Planning Horizon

When the heuristic is applied to scheduling problems (as in the provisioning of active power, which we focus at), the dimensionality q of the target $c \in \mathbb{R}^q$ is interpreted as planning horizon. For real-world applications, it is interesting to know what planning horizon the heuristic is capable of. Figure 4(c) shows the result of planning horizons with a length of $\{2,\ 4,\ 8,\ 12,\ 24\}$ hours in 15-minute intervals (thus $q \in \{8,\ 16,\ 32,\ 48,\ 96\}$). The final fitness in the upper chart deteriorates almost linearly with larger planning horizons. Similarly, the number of simulation steps rises, whereas the number of exchanged messages is not influenced. While we expected the last, we did not expect the influence of the planning horizon on fitness and simulation length, and examined it in more detail. After several experiments with synthetic scenarios (i.e. carefully generated search space values according to [15]), it turned out to be a side effect in our use of the CHP simulation models: Randomly enumerating a rather small number of feasible power profiles does not yield a sufficient coverage of the theoretically feasible action space of the devices. Thus, in the following section, we examine the influence of the size of local search spaces on simulation performance.

3.5 Search Space Complexity

We analyzed scenarios with $\{20,\ 200,\ 2000,\ 20000\}$ pre-generated feasible power profiles per device. This yielded fitness values of 0.12 ± 0.06, 0.02 ± 0.02, 0.003 ± 0.004 and 0.001 ± 0.002, respectively. Since the coverage of the theoretically feasible action space of the simulated devices increases with larger enumerated local search spaces (c.f. Sect. 3.4), simulation fitness improves significantly. The number of simulation steps and the number of exchanged messages per agent per simulation step were constant (132 ± 21 and 1.5 ± 0.04, respectively).

3.6 Population Size

Another interesting property regarding real-world applications is the influence of population size on the heuristic. In Fig. 4(d), a linear increase in simulation steps until termination can be seen. This is consequently due to the increased coordination complexity in larger networks. Yet, since the increase is linear at most, this shows that COHDA is quite robust against the number of participating individuals. Interestingly, the final fitness as well as the number of exchanged messages per time step significantly improve with larger population sizes. The former may be related to the increased diversity, which could already be observed to be beneficial in the analysis of the sizes of local search spaces in the previous section. The latter can be attributed to an increased diameter of the communication graph with larger population sizes. Here, information spreads more slowly, and it takes a longer time for the system to converge.

3.7 Bi-Objective Behavior

As described in Sect. 2.2, we introduced local objective functions at agent level for the COHDA heuristic. As a proof of concept, we conducted an experiment (100 simulation runs) with randomly generated penalty values $p_{ij} \in [0,\ \max(c)]$. The preference adjustment parameter, as defined in (2), was set to $\alpha_i = 0.5$ for all agents, so that the local objectives were considered equally important to the global objective. Figure 5 shows the aggregated results of 100 simulation runs. The heuristic was able to minimize local penalties to a normalized value of

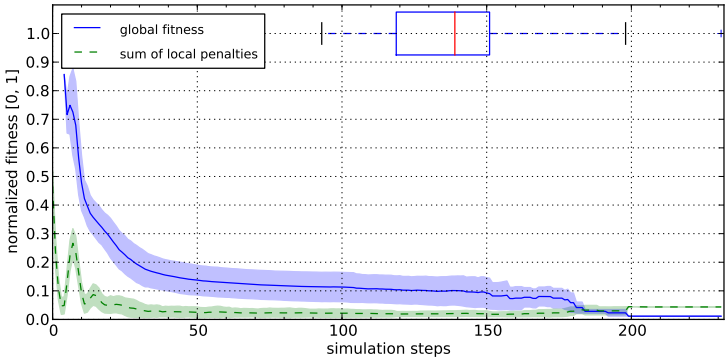


Fig. 5. Aggregated performance of COHDA over 100 simulation runs with distributed local objective functions ($\forall i:\ \alpha_i = 0.5$).

0.02 ± 0.01. At the same time, the global objective fitness could be optimized to a normalized value of 0.15 ± 0.07, which amounts to a remaining imbalance of $33.12\,kW \pm 17.02$ in total over the planning horizon ($0.06\,\% \pm 0.03$ of the targeted $544.26\,kW$).

4 Conclusion

In the contribution at hand, we presented COHDA, which is a self-organizing heuristic for solving distributed combinatorial problems. It was shown that COHDA is inherently adaptive, and exhibits anytime behavior. We applied the heuristic to a problem from the smart grid domain, and performed a thorough evaluation under varying conditions. For this, we implemented an asynchronous multi-agent system with full control over the communication backend. Regarding our example application, it could be shown that the heuristic exhibits convergence and termination, and is robust against unsteady communication networks as well as different network topologies. The run-time of COHDA, in terms of simulation steps, rises linearly with increasing population sizes. Yet it is unaffected by the size of local search spaces, so we conclude that the heuristic is sufficiently

scalable. However, there is a trade-off between the number of simulation steps until termination, and the number of exchanged messages. This trade-off can be adjusted through the density of the communication network (i.e., the average size of the neighborhoods). The evaluation of a bi-objective scenario showed the ability of the heuristic to optimize local penalties as well as a global objective in parallel.

In the present form, COHDA needs a central operator that is able to detect the termination of the process (and thus has a global view on the system). But the actual optimization process is still performed in a truly decentralized manner! A fully decentralized variant of COHDA, however, could be realized by including a distributed termination detection algorithm.

Acknowledgments

Due to the vast amounts of simulations needed, all experiments have been conducted on HERO, a multi-purpose cluster installed at the University of Oldenburg, Germany. We would like to thank the maintenance team from HERO for their valuable service. We also thank Ontje Lünsdorf for providing the asynchronous message passing framework used in our simulation environment, and Jörg Bremer for providing the CHP simulation model.

References

1. Anders, G., Hinrichs, C., Siefert, F., Behrmann, P., Reif, W., Sonnenschein, M.: On the Influence of Inter-Agent Variation on Multi-Agent Algorithms Solving a Dynamic Task Allocation Problem under Uncertainty. In: Sixth IEEE International Conference on Self-Adaptive and Self-Organizing Systems (SASO 2012). pp. 29–38. IEEE Computer Society, Lyon, France (2012)
2. Bernon, C., Chevrier, V., Hilaire, V., Marrow, P.: Applications of Self-Organising Multi-Agent Systems: An Initial Framework for Comparison. Informatica 30(1), 73–82 (2006)
3. Bremer, J., Sonnenschein, M.: A Distributed Greedy Algorithm for Constraint-based Scheduling of Energy Resources. In: SEN-MAS'2012 Workshop, Proc. of the Federated Conference on Computer Science and Information Systems. pp. 1285–1292. Wrocław, Poland (2012)
4. Chapman, A.C., Rogers, A., Jennings, N.R., Leslie, D.S.: A unifying framework for iterative approximate best-response algorithms for distributed constraint optimization problems. The Knowledge Engineering Review 26(04), 411–444 (2011)
5. Gershenson, C.: Design and Control of Self-organizing Systems. Ph.D. thesis, Vrije Universiteit Brussel (2007)
6. Hinrichs, C., Lehnhoff, S., Sonnenschein, M.: Paving the Royal Road for Complex Systems: On the Influence of Memory on Adaptivity. In: Pelster, A., Wunner, G. (eds.) International Symposium Selforganization in Complex Systems: The Past, Present, and Future of Synergetics. Springer (2013), (in press)
7. Hinrichs, C., Lehnhoff, S., Sonnenschein, M.: A Decentralized Heuristic for Multiple-Choice Combinatorial Optimization Problems. In: Operations Research Proceedings 2012. pp. 297–302. Springer (2014)

8. Hinrichs, C., Sonnenschein, M., Lehnhoff, S.: Evaluation of a Self-Organizing Heuristic for Interdependent Distributed Search Spaces. In: Filipe, J., Fred, A.L.N. (eds.) International Conference on Agents and Artificial Intelligence (ICAART 2013). vol. Volume 1 – Agents, pp. 25–34. SciTePress (2013)
9. Hölldobler, B., Wilson, E.O.: The Ants. Belknap Press of Harvard University Press (1990)
10. Jones, J.C., Myerscough, M.R., Graham, S., Oldroyd, B.P.: Honey Bee Nest Thermoregulation: Diversity Promotes Stability. Science (New York, N.Y.) 305(5682), 402–4 (2004)
11. Jordan, U., Vajen, K.: Influence Of The DHW Load Profile On The Fractional Energy Savings: A Case Study Of A Solar Combi-System With TRNSYS Simulations. Solar Energy 69, 197–208 (2001)
12. Kaddoum, E.: Optimization under Constraints of Distributed Complex Problems using Cooperative Self-Organization. Ph.D. thesis, Université de Toulouse (2011)
13. Kroeker, K.L.: Biology-Inspired Networking. Communications of the ACM 54(6), 11 (2011)
14. Li, J., Poulton, G., James, G.: Coordination of Distributed Energy Resource Agents. Applied Artificial Intelligence 24(5), 351–380 (2010)
15. Lust, T., Teghem, J.: The multiobjective multidimensional knapsack problem: a survey and a new approach. International Transactions in Operational Research 19(4), 495–520 (2012)
16. Modi, P., Shen, W., Tambe, M., Yokoo, M.: ADOPT: Asynchronous Distributed Constraint Optimization with Quality Guarantees. Artificial Intelligence 161(1-2), 149–180 (2005)
17. Nieße, A., Lehnhoff, S., Tröschel, M., Uslar, M., Wissing, C., Appelrath, H.J., Sonnenschein, M.: Market-Based Self-organized Provision of Active Power and Ancillary Services: An Agent-Based Approach for Smart Distribution Grids. In: Complexity in Engineering (COMPENG), 2012. pp. 1–5 (2012)
18. Penya, Y.: Optimal Allocation and Scheduling of Demand in Deregulated Energy Markets. Ph.D. thesis, Vienna University of Technology (2006)
19. Pournaras, E.: Multi-level Reconfigurable Self-organization in Overlay Services. Ph.D. thesis, Technische Universiteit Delft (2013)
20. Pournaras, E., Warnier, M., Brazier, F.M.: Local Agent-based Self-stabilisation in Global Resource Utilisation. International Journal of Autonomic Computing 1(4), 350 (2010)
21. Prehofer, C., Bettstetter, C.: Self-Organization in Communication Networks: Principles and Design Paradigms. IEEE Communications Magazine 43(7), 78–85 (Jul 2005), http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1470824
22. Ramchurn, S.D., Vytelingum, P., Rogers, A., Jennings, N.R.: Putting the "Smarts" into the Smart Grid: A Grand Challenge for Artificial Intelligence. Communications of the ACM 55(4), 86 (2012)
23. Reynolds, C.W.: Flocks, Herds and Schools: A Distributed Behavioral Model. SIGGRAPH Comput. Graph. 21(4), 25–34 (1987)
24. Serugendo, G., Gleizes, M.P., Karageorgos, A.: Self-organisation in multi-agent systems. The Knowledge Engineering Review 20(2), 65–189 (2005)
25. Strogatz, S.H.: Exploring Complex Networks. Nature 410(March), 268–276 (2001)
26. Tero, A., Takagi, S., Saigusa, T., Ito, K., Bebber, D.P., Fricker, M.D., Yumiki, K., Kobayashi, R., Nakagaki, T.: Rules for Biologically Inspired Adaptive Network Design. Science (New York, N.Y.) 327(5964), 439–42 (2010)