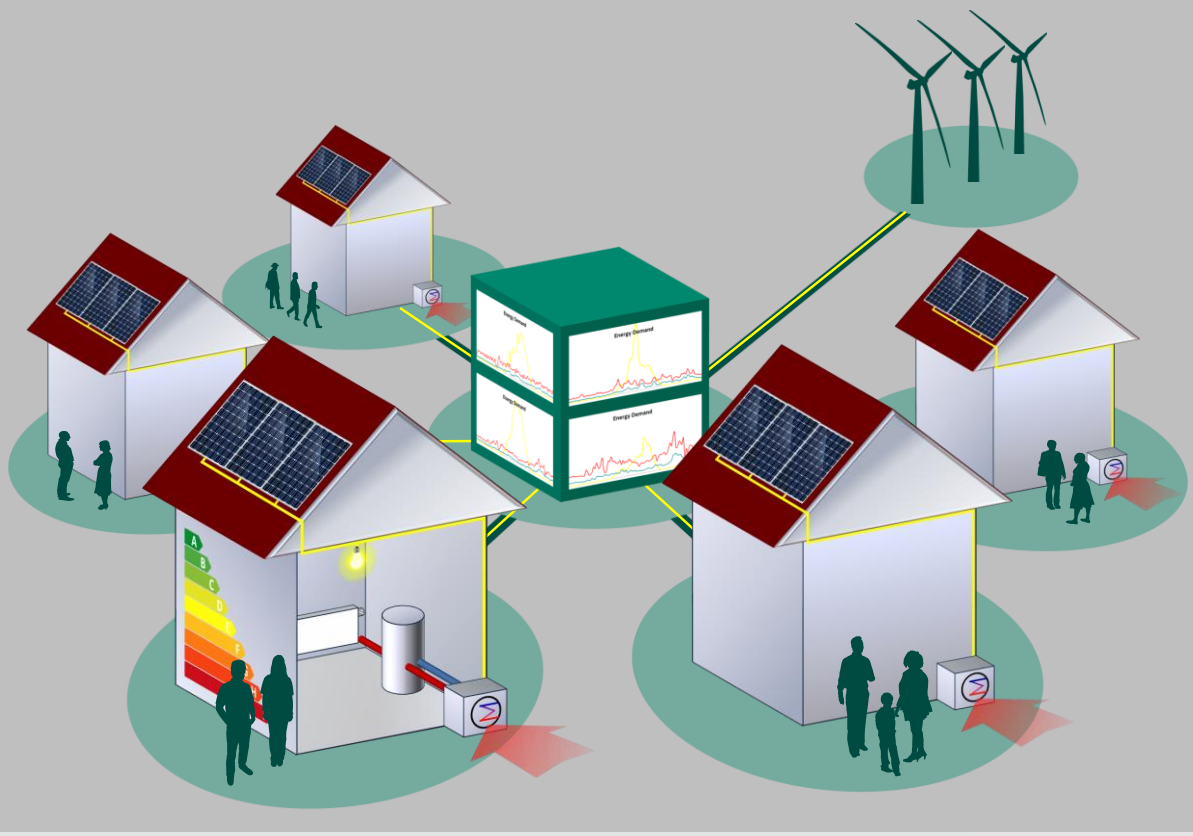
Karlsruhe Institute of Technology

Using automated algorithm configuration to improve the optimization of decentralized energy systems modeled as large-scale, two-stage stochastic programs

By Hannes Schwarz, Lars Kotthoff, Holger Hoos, Wolf Fichtner and Valentin Bertsch

No. 24 | July 2017

WORKING PAPER SERIES IN PRODUCTION AND ENERGY



Using automated algorithm configuration to improve the optimization of decentralized energy systems modeled as large-scale, two-stage stochastic programs

Hannes Schwarz[a,*], Lars Kotthoff[b], Holger Hoos[b,c], Wolf Fichtner[a], Valentin Bertsch[d]

[a] Chair of Energy Economics, Institute for Industrial Production (IIP), Karlsruhe Institute of Technology (KIT), Hertzstraße 16, 76187 Karlsruhe, Germany
[b] Department of Computer Science (CS), University of British Columbia (UBC), Vancouver, BC, Canada
[c] Leiden Institute of Advanced Computer Science (LIACS), Leiden University, Leiden, the Netherlands
[d] Economic and Social Research Institute (ESRI), Dublin, Ireland
[*] Corresponding author. *E-mail address:* hannes.schwarz@kit.edu. *Tel.:*+49 721 60844 694 (H. Schwarz).

Abstract

The optimization of decentralized energy systems is an important practical problem that can be modeled using stochastic programs and solved via their large-scale, deterministic equivalent formulations. Unfortunately, using this approach, even when leveraging a high degree of parallelism on large high-performance computing (HPC) systems, finding close-to-optimal solutions still requires long computation. In this work, we present a procedure to reduce this computational effort substantially, using a state-of-the-art automated algorithm configuration method. We apply this procedure to a well-known example of a residential quarter with photovoltaic systems and storages, modeled as a two-stage stochastic mixed-integer linear program (MILP). We demonstrate substantially reduced computing time and costs of up to 50% achieved by our procedure. Our methodology can be applied to other, similarly-modeled energy systems.

Keywords: OR in energy, large-scale optimization, stochastic programming, uncertainty modeling, automated algorithm configuration, sequential model-based algorithm configuration

Using automated algorithm configuration to improve the optimization of decentralized energy systems modeled as large-scale, two-stage stochastic programs

Hannes Schwarz[a,*], Lars Kotthoff[b], Holger Hoos[b,c], Wolf Fichtner[a], Valentin Bertsch[d]

[a] Chair of Energy Economics, Institute for Industrial Production (IIP), Karlsruhe Institute of Technology (KIT), Hertzstraße 16, 76187 Karlsruhe, Germany

[b] Department of Computer Science (CS), University of British Columbia (UBC), Vancouver, BC, Canada

[c] Leiden Institute of Advanced Computer Science (LIACS), Leiden University, Leiden, the Netherlands

[d] Economic and Social Research Institute (ESRI), Dublin, Ireland

[*] Corresponding author. *E-mail address:* hannes.schwarz@kit.edu. *Tel.:*+49 721 60844 694 (H. Schwarz).

Abstract

The optimization of decentralized energy systems is an important practical problem that can be modeled using stochastic programs and solved via their large-scale, deterministic equivalent formulations. Unfortunately, using this approach, even when leveraging a high degree of parallelism on large high-performance computing (HPC) systems, finding close-to-optimal solutions still requires long computation. In this work, we present a procedure to reduce this computational effort substantially, using a state-of-the-art automated algorithm configuration method. We apply this procedure to a well-known example of a residential quarter with photovoltaic systems and storages, modeled as a two-stage stochastic mixed-integer linear program (MILP). We demonstrate substantially reduced computing time and costs of up to 50% achieved by our procedure. Our methodology can be applied to other, similarly-modeled energy systems.

Keywords: OR in energy, large-scale optimization, stochastic programming, uncertainty modeling, automated algorithm configuration, sequential model-based algorithm configuration

1. Introduction

With the expansion of renewable energy sources (RES) around the world, decentralized energy systems play an increasingly important role (Altmann et al. 2010; Owens 2014; Velik and Nicolay 2016; Yazdanie et al. 2016; Kobayakawa and Kandpal 2016). Their optimal planning and implementation is therefore all the more important. In particular, the fluctuating RES, such as photovoltaic (PV) systems, require a high temporal resolution, resulting in large-scale problems when real energy systems are modeled. Furthermore, the decentralization of the energy system introduces non-negligible uncertainties on the supply side. To meet these challenges, different analytic and computational techniques from Operations Research (OR) can be leveraged (Andriosopoulos et al. 2016). Stochastic Programming is an OR technique that enables an adequate consideration of various uncertainties (see, e.g., Dantzig 1955; Prékopa et al. 1980; Wallace and Fleten 2003; Beraldi et al. 2008; Kuznia et al. 2013). In order to solve a stochastic program computationally, the problem is described by its deterministic equivalent formulation, where a set of scenarios represent the uncertain conditions. This typically results in programs that are much larger than the original and very expensive to optimize (Fragnière et al. 2000).

In this work, we consider a specific real-world decentralized energy system as a case study: the optimization of a residential quarter with a PV system, heat pumps and heat storages (Schwarz et al. 2015). As an optimal implementation of an energy system depends predominantly on the investment at the first stage and on their operation at the second stage, the problem is formulated as a two-stage stochastic program with recourse. To keep the program with more than 100 million variables computationally feasible, the problem is decoupled by fixing the first-stage variables of the program and optimizing them iteratively with a derivative-free optimization (DFO) approach. The sub-problems at the second stage are solved in parallel on a high-performance computing (HPC) system using the commercial MILP solver, CPLEX. At each step of the DFO process, thousands of sub-problems are solved by CPLEX. The default parameter configuration of the solver is unlikely to provide the best performance for all of these problems (see also Hutter et al. 2010). We therefore automatically determine sub-problem-specific parameter configurations using the state-of-the-art algorithm configuration tool SMAC. To the best of our knowledge, this is the first time, automatic configuration of a MIP solver has been demonstrated to be effective on a real-world problem of the magnitude considered here.

The main contribution of our work can be summarized as follows: we describe a general procedure to improve the computational requirements for solving decentralized energy systems, modeled as stochastic program by using automated algorithm configuration.

The remainder of this article is structured as follows. The real-world stochastic optimization problem considered in our work is presented in Section 2. Section 3 gives some background on automated algorithm configuration. Our automated performance optimization approach is described in Section 4 and the results achieved by it are presented in Section 5, followed by further discussion in Section 6. We conclude with a high-level summary of our findings and point out several avenues for future work.

2. Problem description

A real-world case study of a decentralized energy system and its modeling as a two-stage stochastic MILP is presented in Subsections 2.1 and 2.2. As it is not feasible to solve the program on one computer, the problem is decoupled into sub-problems and solved in parallel on the HPC system described in Subsection 2.3.

2.1. Residential quarter as a decentralized energy system

Energy systems are considered as decentralized, when a portion of the energy required to satisfy demand is produced on-site, within the boundaries of, or located nearby and directly connected to, a building, community or development (Wolfe 2008). The residential quarter in our case study pools multi-family and row houses with 70 residential units into a living and energy community on 7708m^2 for up to 180 residents. The quarter has photovoltaic (PV) generators and can handle flexible load through heat pumps and thermal storages. The planning task is to determine the optimal capacities of the storages and, subsequently, their operation under weather-related uncertainties of the electrical and thermal demand as well as energy supply. Fig.1 depicts the energy setup of the quarter.

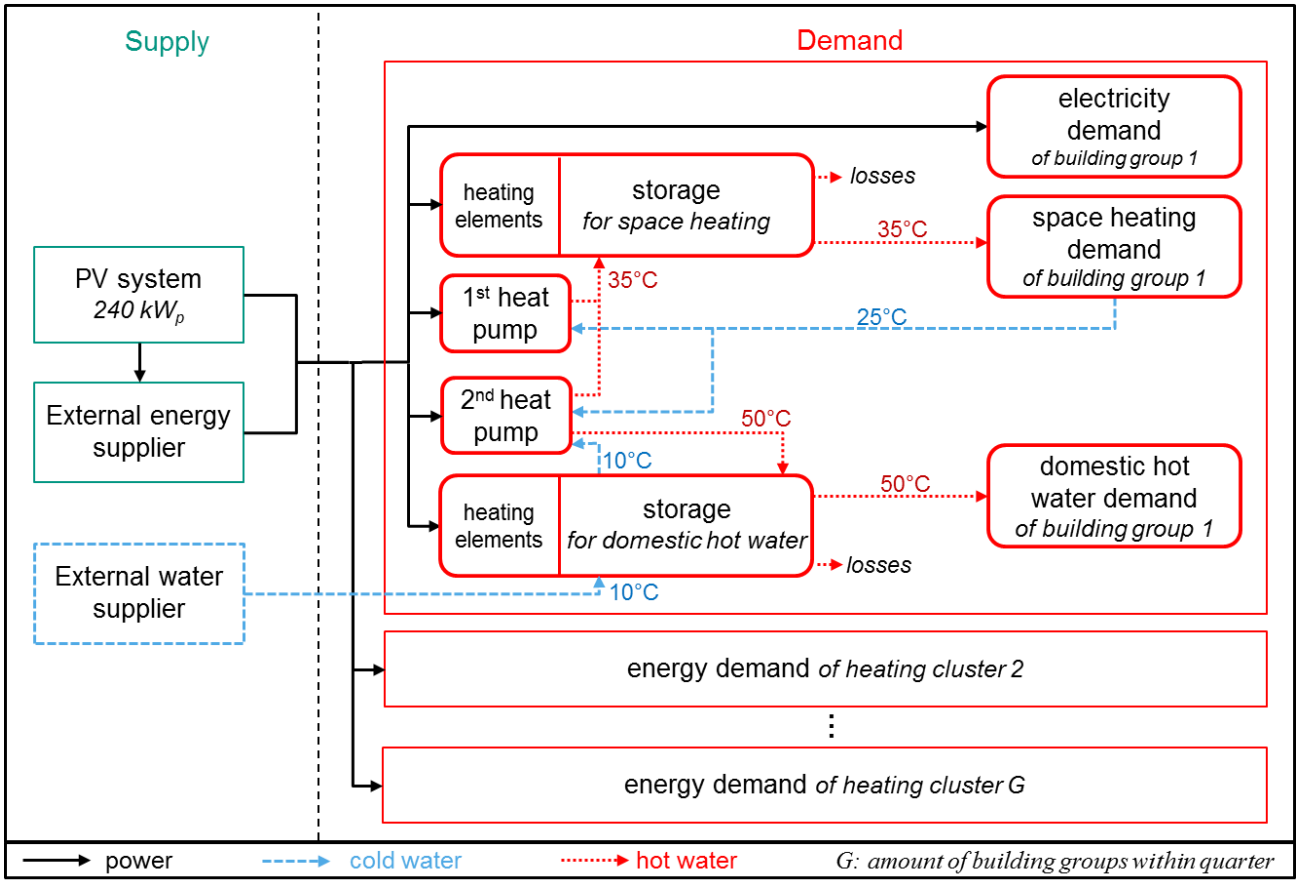


Fig.1: Energy setup of the residential quarter.

On the supply side, there is a PV system of 240kW$_p$ installed, providing power between 0 and up to approximately 200kW$_{el}$. If the PV supply is insufficient, electricity can be purchased from an external energy supplier at a given tariff. On the demand side, 70 households are clustered into $G = 4$ building groups, each with a fluctuating, uncertain demand of electricity, domestic hot water and space heating. To cover the heat demand, each building group is equipped with heat storages in combination with two air-water heat pumps that can provide heat at half or full load up to 120kW$_{th,}$ depending on ambient air temperature. Additional heating elements in the storages secure the thermal covering in times of peak demand and provide a disinfection function. The heating system is separated into two cycles: the closed cycle for space heating runs at lower temperatures than the one for domestic hot water, resulting in a higher coefficient of performance (COP) of the first heat pump and lower heat losses of the storage. The target temperature is assumed to be 35°C and can

drop by approximately 10 Kelvin below this target. For the domestic hot water requirements, fresh water is obtained from an external water supplier and heated by a second heat pump in an open loop from about 10°C to 50°C.[1]

2.2. Modeling of decentralized energy systems as two-stage stochastic MILP

In order to determine an optimal storage size that leads to minimal energy system costs under uncertain conditions, the residential quarter is modeled as a two-stage stochastic program (for a compact introduction to stochastic programming, see Prékopa 1995; Shapiro et al. 2009). The objective function of the program is defined as:

$$costs^* = \min_{\boldsymbol{c_{g,i}}, \boldsymbol{e^{grid}_{\omega,t}}, \boldsymbol{e^{fi}_{\omega,t}}} ANF \cdot \sum_{g=1}^{G=4} \sum_{i=1}^{k_1} cost_i \cdot \boldsymbol{c_{g,i}} + \frac{1}{N} \cdot \sum_{\omega=1}^{N=100} \sum_{t=1}^{T=35040} p^{grid} \cdot \boldsymbol{e^{grid}_{\omega,t}} - p^{fi} \cdot \boldsymbol{e^{fi}_{\omega,t}}. \quad (1)$$

At the first stage, the capital costs of each investment i of building group g, such as the storage for space heating and for domestic hot water, is converted into an equivalent series of uniform amounts per period. The lifetime of the investment and an alternative investment possibility at a certain interest rate of the fixed capital is taken into account by the annuity factor ANF. In this case study, a technical lifetime of 20 years is assumed, with an interest rate of 10%. At the second stage, energy costs of each scenario $\omega = \{1, \dots, \Omega\}$ are affected at each time step t by the energy obtained from the external grid $\boldsymbol{e^{grid}_{\omega,t}}$ at price p^{grid}, minus the energy fed into the grid $\boldsymbol{e^{fi}_{\omega,t}}$ at feed-in tariff p^{fi}. The period $t = \{1, \dots, 35040\}$ includes one year, with a temporal resolution of 15 minute time steps. In total, the optimization is carried out for 100 scenarios generated on the basis of a Markov process.[2] An essential constraint of the system is that the electrical and thermal demand and supply are balanced at any time. The thermal supply in the system is limited by heat pumps plus heating elements and heat storages. The heat pumps can only run stepwise at idle, half or full load, while the heating elements can modulate their heat output on a continuous scale. The storage levels connect the states of time step t to step $t + 1$ and result in a complex stochastic MILP. The entire program is listed in Appendix A. For further information about the program and the scenario generation, see Schwarz et al. (2015).

2.3. Initial optimization approach with default configuration

The sub-problems of the two-stage stochastic MILP are solved in parallel by explicitly setting first-stage variables. The sub-problems are solved by CPLEX for given first-stage variables that are iteratively optimized by an outer DFO. The entire optimization procedure is depicted in Fig.2.

[1] Note that the higher temperature difference results in a larger energy content at the same volume compared to storages for space heating.

[2] Markov processes have proven suitable to generate PV generation and energy demand of the decentralized energy system that depend essentially on fluctuating and uncertain meteorological parameters (see Schwarz et al. 2015 for details).

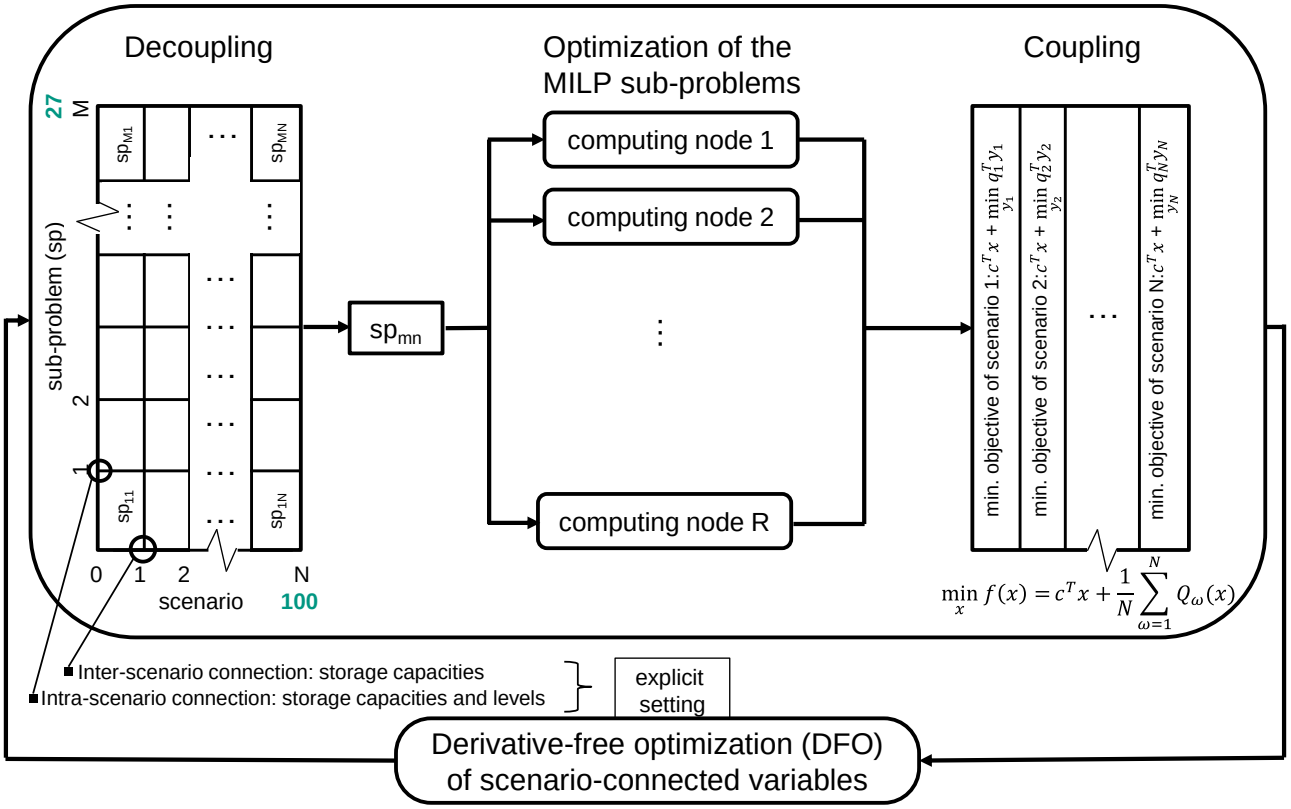


Fig.2: Optimization approach of the two-stage stochastic MILP (used with permission from (Schwarz et al. 2015)).

The first-stage variables, i.e., the storage capacities as inter-scenario connections, are optimized using the steepest-ascent hill-climbing DFO algorithm (Taborda and Zdravkovic 2012). In order to minimize the costs, the storage capacities for space heating and domestic hot water of each building group are sequentially altered by a positive and negative step size s_i. The costs are minimized for each altered storage capacity, and the step with minimal costs is accepted. When there is no improvement in terms of the objective function, the step size is divided by two. This process is repeated until the relative change of the objective function value is smaller than 0.1%.

As a result of the DFO in the first stage, the problem can be decoupled into $N = 100$ sub-problems (one for each scenario). By fixing the storage capacities that connect the time steps of the period $t = \{1, \dots, T\}$ within a scenario, each scenario ω can be decoupled over time t: the one-year period of a scenario is decoupled into periods of two weeks, resulting in $M = 27$ sub-problems per scenario.[3] Hence, $4 \cdot N \cdot M = 10\,800$ sub-problems for each building group need to be solved for a storage capacity given by the outer DFO; the factor 4 stems from the number of positive and negative steps for the storage of space heating and domestic hot water. The sub-problems are solved using CPLEX (version 12.6.2) with a MILP gap of 0.6% and a cutoff-time of 1 800s, so that the hill-climbing approach efficiently progresses to the optimum.[4] The sub-problems are solved in parallel on a HPC cluster. Even when using the HPC system, the initial optimization process requires more than a week of wall-clock time for one building group. Computations for the entire quarter and additional analyses of the energy system are very expensive.

[3] The scenarios are further decoupled, because one scenario cannot be solved within 48 hours for a MILP gap of 0.6% on single computer. The intra-scenario connecting storage levels are not optimized by the DFO, but set to reasonable levels resulting in a negligible error to the optimum of less than 1%.

[4] The MILP gap is set to 0.6%, because we observed practically no improvement of the sub-problem solution after several days of additional computing time; this sub-problem solution is almost always found by CPLEX within 1 800s.

3. Automated algorithm configuration

Most modern software systems, such as the CPLEX solver we use here, expose a multitude of parameters to the user. The default values of these parameters do not provide optimal performance in every case (Atamtürk and Savelsbergh 2005). Configuring them on a case-by-case basis is imperative to be able to solve problems quickly (Hutter et al. 2010). Unfortunately, the space of possible parameter settings is vast, and there is no theoretical knowledge on how parameters should be set. Therefore, in most applications of CPLEX and similar highly parametric solvers, default parameter settings are used or parameter settings are determined on tedious yet limited, ad-hoc manual experimentation.

Automatic algorithm configuration provides an effective solution to this problem (see, e.g., Hutter et al.; Hutter et al. 2011). Instead of manually experimenting with different parameter settings, the user only has to specify the target algorithm (i.e., the algorithm whose performance is to be optimized), the parameter space (defined through the names of the parameters, their corresponding domains and default values), a set of representative problem instances and a performance metric. Then the machine does the rest: the configuration procedure repeatedly selects and evaluates candidate parameter settings, with the goal of optimizing the given performance metric. The general approach is depicted in Fig.3.

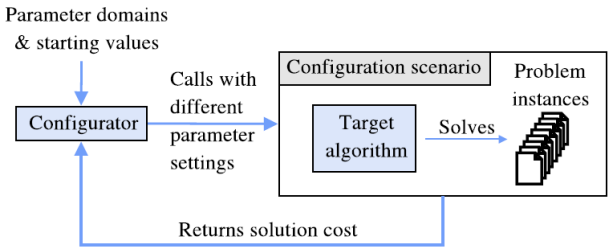


Fig.3: Automated algorithm configuration (used with permission from Hutter et al. 2010).

Automatic algorithm configuration approaches treat the target algorithm as a black box, in that they do not require knowledge of its inner workings; instead, they evaluate its performance by observing it empirically. Configurations that maximize the performance are returned by the procedure.

A simple approach to algorithm configuration is to evaluate a very large number of different configurations, either chosen systematically to cover the given configuration space or randomly sampled. For expensive target algorithms, such as CPLEX in our application, this is usually infeasible because of resource constraints. Instead, state-of-the-art model-based algorithm configuration methods model the parameter-performance response surface with a so-called surrogate model. The surrogate model is cheap to evaluate and provides predictions of how the target algorithm will perform on configurations on which it has not been evaluated. An acquisition function uses these predictions, along with the uncertainty associated with them, to propose the next configuration on which to evaluate the target algorithm.

The surrogate models used by such sequential model-based optimization (SMBO) procedures are induced using machine learning methods, based on the performance of the target algorithm evaluated on a relatively small number of configurations. The acquisition function balances exploitation of regions of the parameter space known to contain good configurations and the exploration of regions where the surrogate model makes predictions with high uncertainty. Together, surrogate model and acquisition function provide a principled way to intelligently explore large configuration spaces.

After each evaluation, the new information on the actual performance of the target algorithm is incorporated into the surrogate model, and the acquisition function predicts the next configuration to

be evaluated; this informs the sequential behavior. The process stops when a user-specified configuration budget is exhausted. The approach provides anytime behavior: the incumbent configuration can be retrieved at each step of the process.

A large body of existing research has been devoted to this problem; a comprehensive survey is beyond the scope of this article. Further information can be found, for instance, in (Jones et al. 1998; Hutter et al. 2009; Ansótegui et al. 2009; Mascia et al. 2014).

SMAC is a state-of-the-art automatic algorithm configurator based on the SMBO approach (see Hutter et al. 2011 for details). We decided to use it in this work, because it is known to perform well on a broad range of algorithm configuration tasks, is readily available and relatively easy to use. Alternative SMBO-based configuration procedures include the Tree-based Parzen Estimator (TPE) (Bergstra et al. 2011) and Spearmint (Snoek et al. 2012), but there is no reason to believe that either of those would reach or surpass the performance of SMAC when configuring CPLEX.

4. Description of the methodology

The application of SMAC to the integrated MILP solver is described in Subsection 4.1. A more detailed analysis of the differences between the optimized configuration (obtained from SMAC) and the default configuration of the MILP solver is provided in Subsection 4.2.

4.1. MILP solver configuration using SMAC

To improve performance, we partition the sub-problems, select instances for each partition and apply SMAC to each of these groups of instances to improve performance. Partitioning the sub-problems is necessary, because they are structurally different and we expect different configurations to be most suitable. By applying SMAC to each partition individually, we allow for partition-specific configurations that result in higher overall performance. Equally, an appropriate, representative selection of training instances is important to enable finding a well-performing solver configuration for the entire partition.

As the running time of each hill-climbing iteration is lower-bounded by the sub-problem with the longest runtime, difficult instances are selected for each group. Improvements on these instances will achieve the highest performance improvements for the overall method. The first choice are sub-problems that are not solved by the default CPLEX parameter configuration within the given cutoff-time of 1 800s, the second choice are sub-problems that are solved successfully, sorted by running time in descending order. In practice, we found it beneficial to apply an iterative approach, where we run SMAC on a small number of instances first and use the optimized configuration from this step as a starting point for running SMAC on the full set of instances. The complete procedure is described below. Details specific to the optimization of the residential quarter are given in *italics*:

1. Partition the sub-problems sp_{mn} with respect to the problem structure (e.g. by winter, summer, transition season, by month or by week).

   The sub-problems are partitioned in 27 parts, as each scenario is decoupled into 2 weeks, where partition 1 represents week 1 and 2 of period $t = \{1, \dots, T\}$, partition 2 represents week 3 and 4, and so on. Partition 27 presents the last part of the one-year period which remains as only 1 day.

2. Select a number of instances per partition that can be computed within a reasonable period of time. The selection criterion depends on the problem structure.

   Overall, 9 instances per partition are selected, where 8 of 9 difficult instances cannot be solved by the default CPLEX parameter configuration with a given MILP gap of 0.6% and a cutoff-time of 1 800s. One instance is chosen randomly to avoid finding configurations that are too specific to difficult instances.

3. Train SMAC on a small set of instances to get an improved initial CPLEX configuration per partition. Chose an appropriate objective (minimization of mean wallclock time, mean central processing unit (CPU) time, etc.).

SMAC is initially applied to 3 difficult instances for each partition (total configuration time of 24h per partition on 1 separate *CPU core). The mean wallclock time on these instances is minimized. Unsuccessful runs that could not be solved within the cutoff-time of 1 800s are counted as having taken 18 000s (i.e., 10 times the cutoff).*

4. Train SMAC with the same objective again on all selected instances, starting with the optimized configuration from the previous step, to find the final CPLEX configuration for each partition.

SMAC is run on the 9 difficult instances from Step 2 per partition on 1 separate *CPU core (total configuration time of 72h per partition on 1* separate *CPU core).* [5]

5. Use the partition-specific optimized configuration for the optimization of the sub-problems sp_{mn}. The training phase of SMAC and the MILP solving process must be executed on the same computer system.

The SMAC training and the CPLEX solving process are executed on a Linux-based HPC cluster using 512 CPU cores with two threads at 2.6GHz and 16GB RAM per core.

If the resulting performance gain is not sufficient, repeat the procedure with another partitioning. Usually, finer partitions lead to better improvements, but make the training phase more expensive, due to a higher number of partition-specific parameter configurations that need to be found.

4.2. Ablation analysis

As our results (presented in Section 5) demonstrate, by means of automated algorithm configuration using SMAC, we are able to effectively find optimized parameter settings for CPLEX. To further analyze these optimized configurations, we use ablation analysis. Ablation analysis assesses the effect of each parameter that differs between two configurations on target algorithm performance in order to identify the most impactful parameter changes. Given a default and an optimized configuration, this is done by changing the value of one parameter at a time to determine what part of the performance difference between the two configurations it accounts for. This process incrementally moves from one configuration to the other and thus takes into consideration interactions between parameters; it proceeds in a greedy fashion, determining in each iteration the largest possible performance improvement and the parameter responsible for it. Ablation analysis determines which parameters are important to achieve improved performance and which have little or no effect in a particular configuration scenario (see Fawcett and Hoos 2016 for details).

5. Results

We present computational results for our automatic configuration approach in Subsection 5.1, ablation analysis results in Subsection 5.2 and the effective gain of using automated algorithm configuration in Subsection 5.3.

5.1. Reduction in running time

We use 27 different partition-specific CPLEX configurations determined by SMAC for the inner optimization of the previously described MILP sub-problems. In total, 6 outer hill-climbing iterations

[5] The final optimized configuration of the most partitions is already found after total configuration time of 24h, only 5 transitions partitions requires the full time of 72h on 1 separate CPU core.

are needed to find optimal storage capacities for one building group. The mean wallclock time of all 54 000 computations is reduced substantially, from 947s to 493s, by using the optimized configurations.[6] The Tukey boxplot shown in Fig.4a illustrates this difference.[7] Even more drastically then the mean, the median running time drops from 1 631s to 168s, while the upper quartile are reduced from 1 800s to 509s. The deviation between the mean and the median of the default case is due to the skew of the wallclock time distribution: the sub-problems tend to be solved either quickly or towards the end of the given cutoff-time of 1 800s. Approximately one in six sub-problems are not solved, but have achieved a MILP gap of 1% or less, which still provides a reasonably accurate basis for the hill-climbing approach. For the optimized configurations, the deviation between the mean and the median is mainly caused by outliers corresponding to unsuccessful runs. In comparison to the default, 30% more sub-problems are solved.

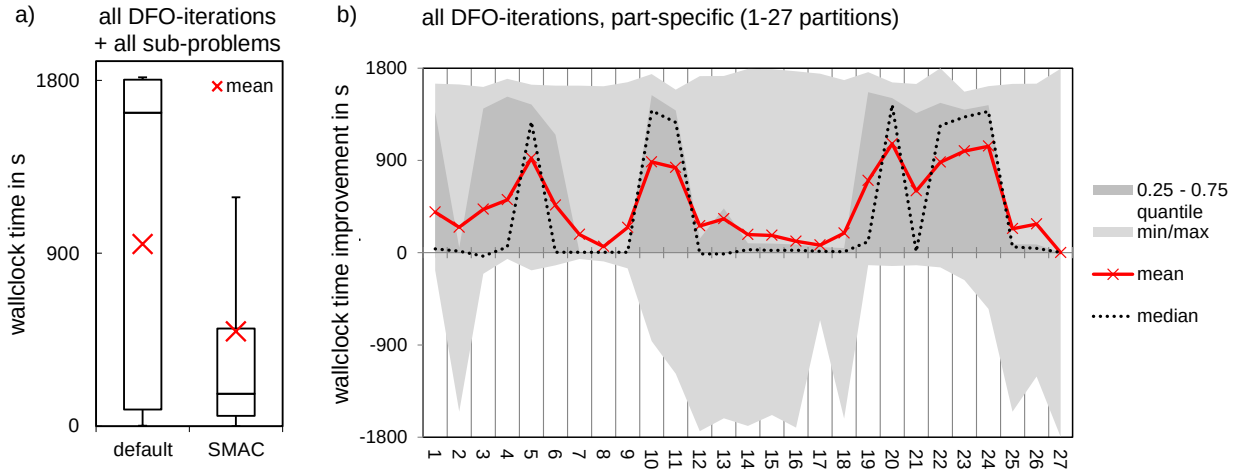


Fig.4: Wallclock time for the sub-problems (MILP gap=0.6%, cutoff-time=1 800s) for a) the default configuration and b) the optimized partition-specific configurations obtained from SMAC.

Fig.4b provides more detailed results for the 27 partitions and shows that the biggest improvements are achieved for the transition seasons (partition 5–11 and 20–23). These sub-problems are more complex than those for the winter (partition 1–4 and 24–27) and summer season (partition 12–19), for which little or no reduction of running time could be achieved. In summer, the space heating demand is zero, and accordingly, the heat pump is turned off. The integer variables of this heat pump can be set to zero, which simplifies the optimization task. In winter, the PV supply minus the electricity usage of the households is low or even zero. Consequently, the potential for profitable load shifting is lower than in transition seasons, which also simplifies the optimization task. The wallclock times for solving the winter and summer sub-problems are much lower than that of the transition sub-problems for the default configuration. Thus, the potential wallclock time reduction is higher for more complex sub-problems in the transition seasons: There is wallclock time improvement of 1 403s within the 0.75 quantile. The negative lower quartile shows that the optimized configuration, which is based on 9 of 100 instances per partition, can lead to worse performance on some sub-problems. More training instances or a finer partitioning could remedy this issue, but would increase the resource requirements for the automatic algorithm configuration process.

[6] There are 4 storage capacities that do not require optimization. Therefore, only 54 000 sub-problems are solved, instead of 64 800 (= 4 storage capacities per iteration times 27 parts per scenario x 100 scenarios times 6 iterations).
[7] The ends of the whiskers represent the lowest wallclock time within the 1.5 interquartile range (IQR) of the lower quartile, and the highest wallclock time within the 1.5 IQR of the upper quartile. Outliers are not shown.

5.2. Parameter importance results

The path of improvements (ablation path) shows the reduction of the mean wallclock time for the 9 selected instances per partition. It is important to note that this path is not the one followed by SMAC, but the optimal improvement path determined by ablation analysis post-hoc. For each partition, 20–50 CPLEX parameters differ between the default and the final optimized configuration obtained from SMAC. The mean wallclock time for solving the selected instances across all partitions with the default configuration (P-0_Default) is 1 556s and is reduced to 481s by the optimized configuration (P-All_SMAC). Note that these values differs from the wallclock time improvement in Subsection 5.1, since not all computations of the sub-problems, but only the 9 selected instances per partition are considered. Fig.5 shows the path of improvements for all 27 partitions: beginning from P-0_Default, over the first three most-effective parameter adjustments P-1, P-2 and P-3 and ending with P-All_SMAC. For each partition, the improvement paths are shown uniformly for the seasons of transition (dotted red line), winter (solid black line) and summer (dashed orange line).

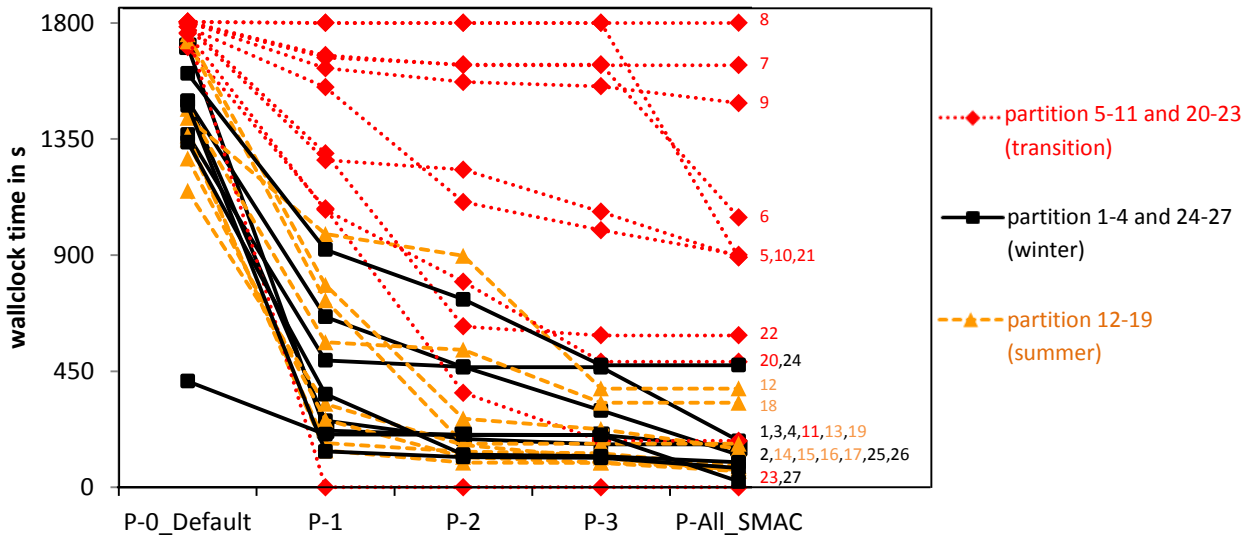


Fig.5: Improvement paths for all 27 partitions (mean wallclock time for the 9 selected instances) for the optimized configuration for the 3 parameters with the highest effect and the final configuration obtained from SMAC.[8]

Fig.5 shows that similar problems, i.e. instances of the same season, tend to show similar ablation paths. However, within the winter, summer and transition partitions, there is still a notable difference of improvements, indicating that a finer segmentation into 27 partitions may achieve further improvements. In particular, the wallclock time of partitions in the transition seasons is reduced to a few seconds in some cases and remains close to the default in others. There is no wallclock time reduction for partition 8. For these 9 difficult instances, SMAC was not able to find a CPLEX configuration that leads to a solution within the cutoff-time of 1 800s in multiple runs and very high configuration budgets of 3 days in total on a processor with 1TB RAM. We ended up solving the sub-problems in partition 8 using the configuration optimized for the similar partition 7 and obtained an improvement in this way (see Fig.4b). Table 1 lists the three parameters that have the largest effect on performance on average, a brief description and their mean wallclock time reduction and their total relative wallclock time reduction over all partitions. Table B.1 in Appendix B presents analogous results for all 27 partitions in parentheses.

[8] The default configuration (P-0_Default) has a lower mean wallclock time of 477s on partition 27 because of the lower time horizon of 1 day compared to 2 weeks for the other partitions.

Table 1: The three parameters with the highest impact on wallclock time leading to a total relative reduction of 40% (descriptions of the parameters are obtained from the CPLEX User's manual IBM 2016).

parameter name	Description	avg. reduction (total rel. reduction)
1. MIP strategy rinsheur	sets the frequency to apply the relaxation induced neighborhood search (RINS) heuristic	618s (27%)
2. MIP strategy nodeselect	rules the selection of the next node to process when backtracking	384s (9%)
3. MIP limits aggforcut	limits the number of constraints that can be aggregated for generating flow cover and mixed integer rounding (MIR) cuts	293s (4%)

5.3. Advantages of the applied methodology

The user cares most about the performance improvement that was achieved for the overall wallclock time as a function of the number of CPU cores utilized. Therefore, the computing time required for each step of our approach, including solving all MILP sub-problems, were logged. Based on these times, we determined overall wallclock time. Computing cost is based on Amazon EC2 (https://aws.amazon.com/ec2/pricing): 0.239US$ per full hour and node with 2 CPU cores and 16GB RAM that is required for the CPLEX optimization of the sub-problems. Fig. 6 compares the default and SMAC-optimized configurations with respect to the total wallclock time and cost as a function of the number of CPU cores. The SMAC configuration effort is additionally illustrated. Note that the overhead of finding the optimized configuration is only incurred once, but the found configuration can be used to solve MILP instances from different scenarios more efficiently. In case of parameter changes and further analyses, the deployed optimized configuration would quickly amortize the configuration cost and over time achieve increasingly larger relative improvements.

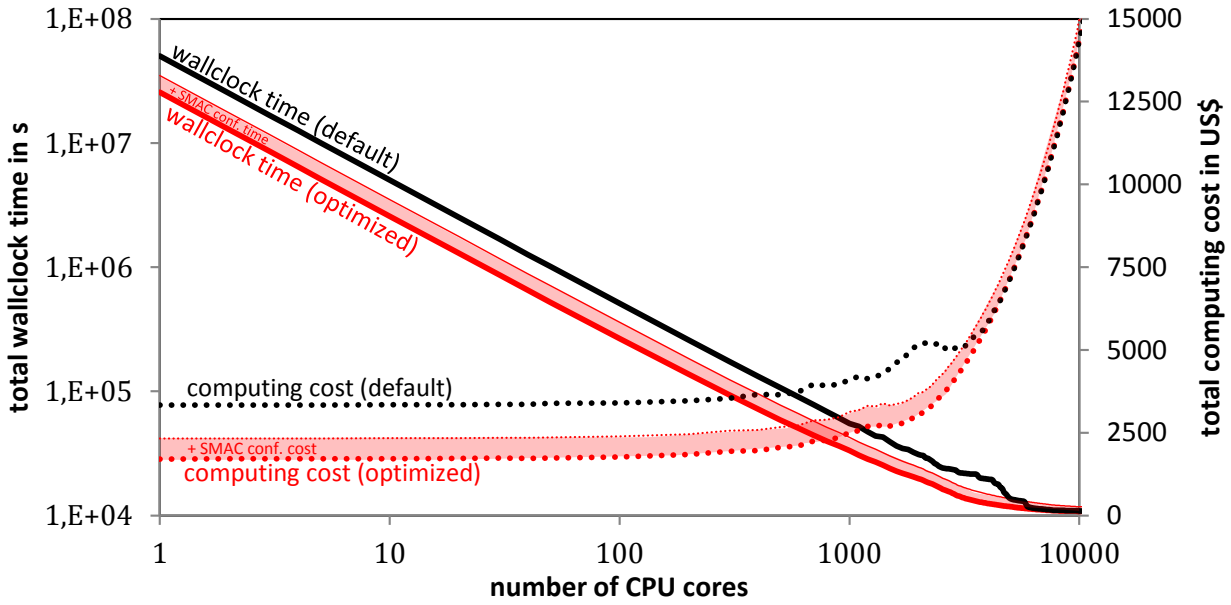


Fig.6: Total wallclock time (solid lines, left log-scaled vertical axis) and total computation cost (dotted lines, right vertical axis) vs. utilized CPU cores (log-scaled horizontal axis) using the default CPLEX configuration (black) and the optimized CPLEX configuration obtained from SMAC plus the configuration effort (red).[9]

[9] We employed SMAC only up to 27 CPU cores and assume the same scaling effect on total wallclock time and cost, when SMAC is employed in parallel mode on more CPU cores.

For one CPU core, the total wallclock time is theoretically reduced from 592 days (51 149 983s) for the default configuration to 308 days (26 629 672s) for the optimized configuration. As a result, the total computing cost amounts to about 50% of the initial optimization with the default configuration (about 30% when SMAC configuration is taken into account). Up to about 100 CPU cores, this ratio in wallclock time and computing cost can be maintained, because there is a sufficient number of sub-problems that can be solved simultaneously at any given time. Beyond 100 CPU cores, the cost increases, because some CPU cores are idle while other CPU cores are still computing difficult sub-problems that are process-decisive for the DFO approach. At about 6 000 CPU cores, this effect compensates the individual wallclock time reduction of the sub-problems achieved by the optimized CPLEX configurations. Consequently, there is no further time and cost reduction in this case.

For this paper, we used up to 512 CPU cores of one HPC cluster. This corresponds to a wallclock time of about 28h and computing cost of 3 716US$, which are reduced by approximately 45% when using the optimized configurations.[10] Even when SMAC training time is fully accounted for, about 30% of time and costs are saved.

6. Discussion

The results presented in Section 5 demonstrate that careful partitioning and selection of problem instances is crucial to achieve good results with automated algorithm configuration methods for a real-world application problem of the magnitude considered here. The fewer partitions are used, the smaller the potential performance improvement becomes. Too few instances can lead to configurations that are too specific and do not perform well on other instances of the partition. On the other hand, too many partitions and instances increase the training cost and counteract the subsequent performance improvements. Partitioning based on winter, summer and transition seasons enabled an overall performance improvement of about 33%. A finer partitioning into 27 partitions based on two-week intervals achieves a performance improvement of 50%, while maintaining acceptable training times.

CPLEX ships with an internal tuning tool that we could have used instead of SMAC. It tries up to 30 different pre-determined configurations and chooses the best of these. While this process takes much less time, the achieved performance improvements are much less. For example, a randomly chosen instance that is not solved within 1 800s by the default configuration could be solved within 420s with the best configuration found by CPLEX tuning tool, but the same instance is solved 4 times faster with the best configuration found by SMAC. This is consistent with results indicating that state-of-the-art automatic algorithm configuration achieves substantially better results than the CPLEX tuning tool (Hutter et al. 2010).

Our ablation analysis shows that only few parameter adjustments are necessary to achieve major improvements: changing just three parameters (MIP strategy rinsheur, MIP strategy nodeselect, and MIP limits aggforcut) already achieves 40% of the total improvement on average. While these findings as well as the actual configurations we determined are specific to our case study, our overall methodology is applicable more broadly, and we expect that similar performance improvements and ablation results can be obtained on challenging, large-scale problems similar to the one considered here.

Further performance improvements can be achieved by changing the MILP gap. In this specific optimization problem, the outer hill-climbing DFO requires a MILP gap of about 0.6% for the sub-problems to work efficiently, but only when the outer DFO is close to the optimum. One approach to

[10] In Practice, due to time restrictions per job of the HPC queuing system, the computation takes about a week using CPLEX in its default configuration, and less than half a week when using the optimized configurations.

reduce the computational requirements would be to start with a wider gap and reduce it dynamically as the outer DFO converges on optimal values. A similar approach could be used for the cutoff-time.

A different way to reduce the required computational resources would be to optimize the amount of memory required as well as the running time. Some sub-problems require up to 16 GB RAM with the current best configuration we have found. Reduced memory requirements would allow more runs on a single node or permit the use of cheaper compute nodes with less RAM.

It might seem tempting to apply algorithm configuration to the full problem, without decoupling, but unfortunately, at the present, considering the computational complexity involved, this appears infeasible.

7. Conclusion and future work

Our energy system structure is changing from centralized to decentralized energy systems that are subject to manifold sources of uncertainty such as the electrical and thermal demand and the energy supply. Stochastic Programming helps to avoid insufficient investment decisions but typically results in very large-scale optimization problems. Here, we have modeled a real-world residential quarter as a two-stage stochastic mixed-integer linear program (MILP), which was subsequently solved on a high-performance computing (HPC) system. With the default configuration of CPLEX, we require about 28h of computation time on 512 CPU cores. By applying the automatic algorithm configuration tool SMAC, we were able to determine a set of performance-optimized configurations and achieve performance improvements of up to 50% overall, and up to 30% when fully accounting for the effort of finding the optimized configurations. This enables not only a faster solution of the given problem, but also facilitates additional analyses, and ultimately makes it possible to tackle more complex energy systems.

Further computing time and cost reductions could be achieved by adapting the MILP gap and/or cutoff-time parameters. Moreover, the exact CPLEX optimization could be substituted by a heuristic method (e.g., using machine learning for quick approximation of solutions to sub-problems) subject to the condition of a sufficient solution quality for the outer DFO approach. Another promising direction would be to adapt the DFO method to be more efficient. Further research is needed into automated algorithm configuration for large-scale problems with infeasible/unreasonable configuration times.

Acknowledgments

The authors acknowledge support by the state of Baden-Württemberg through bwHPC and the Germany Research Foundation (DFG) through grant no INST 35/1134-1 FUGG to HS, as well as through an NSERC Discovery Grant to HH.

Appendix A

The entire two-stage stochastic MILP of the residential quarter is shown in the following (nomenclature is listed in Table A.1):

Objective function:

$$costs^* = \min_{\boldsymbol{c}_{g,i}, \boldsymbol{e}^{\boldsymbol{grid}}_{\omega,t}, \boldsymbol{e}^{\boldsymbol{fi}}_{\omega,t}} ANF \cdot \sum_{g=1}^{4} \sum_{i=1}^{k_1} cost_i \cdot \boldsymbol{c}_{\boldsymbol{g,i}} + \frac{1}{N} \cdot \sum_{\omega=1}^{N} \sum_{t=1}^{T} \left(p^{grid} \cdot \boldsymbol{e}^{\boldsymbol{grid}}_{\boldsymbol{\omega,t}} - p^{fi} \cdot \boldsymbol{e}^{\boldsymbol{fi}}_{\boldsymbol{\omega,t}} + f \cdot \sum_{g=1}^{4} \sum_{u=1}^{2} \boldsymbol{q}_{\boldsymbol{\omega,g,u,t}} \right), \quad \text{(A.1)}$$

- the installed PV capacity of the quarter: $\sum_{g=1}^{4} \boldsymbol{c_{g,i=PV}} = 240$,
- the number of heat pumps for SH within a building group: $\boldsymbol{c_{g,i=HP_{SH}}} = 1$,
- the number of heat pumps for DHW within a building group: $\boldsymbol{c_{g,i=HP_{DHW}}} = 1$,
- the number of heating elements for the SH storage: $\boldsymbol{c_{g,i=HE_{SH}}} = 4$,
- the number of heating elements for the DHW storage: $\boldsymbol{c_{g,i=HE_{DHW}}} = 4$,
- with $f = 100\,000€/\mathrm{kWh_{el}}$.

Additionally, electrical supply and demand have to be balanced:

$$e_{\omega,t}^{pv} + \boldsymbol{e_{\omega,t}^{grid}} = d_{\omega,t}^{ee} + \sum_{g=1}^{4}\sum_{u=1}^{2}\left(\boldsymbol{d_{\omega,g,u,t}^{hp}} + \boldsymbol{d_{\omega,g,u,t}^{he}}\right) + \boldsymbol{e_{\omega,t}^{fi}} \qquad \forall\omega\ \forall t, \quad \text{(A.2)}$$

with supplied PV energy $e_{\omega,t}^{pv} = \sum_{g=1}^{4} e_{\omega,t}^{pv,kwp} \cdot \boldsymbol{c_{g,i=PV}}$ and balanced thermal supply and demand:

$$\begin{aligned} COP_{\omega,u,t} \cdot \boldsymbol{d_{\omega,g,u,t}^{hp}} + \eta \cdot \boldsymbol{d_{\omega,g,u,t}^{he}} + (1 - l_u) \cdot \boldsymbol{s_{\omega,g,u,t}} + \boldsymbol{q_{\omega,g,u,t}} \\ = d_{\omega,g,u,t}^{th} + \boldsymbol{L_{\omega,g,u,t}} + \boldsymbol{s_{\omega,g,u,t+1}} + \boldsymbol{pos_{\omega,g,u,t}} \cdot r_u \ \forall\omega, \forall g, \forall u,\ \forall t, \end{aligned} \quad \text{(A.3)}$$

with the storage heat losses $l_{u=SH} = 0.003$ and $l_{u=DHW} = 0.006$ and ramp up losses $r_u = 0.05$.
The storage possibility is restricted by:

$$\begin{aligned} s_{g,u}^{min} \leq \boldsymbol{s_{\omega,g,u,t}} \\ \leq \boldsymbol{c_{g,i=S_u}} \end{aligned} \qquad \forall\omega, \forall g, \forall u,\ \forall t, \quad \text{(A.4)}$$

where $s_{g,u}^{min} = 0$. Load changes are taken into account by:

$$\boldsymbol{z_{\omega,g,u,t+1}} - \boldsymbol{z_{\omega,g,u,t}} = \boldsymbol{pos_{\omega,g,u,t}} - \boldsymbol{neg_{\omega,g,u,t}} \qquad \forall\omega, \forall g, \forall u,\ \forall t. \quad \text{(A.5)}$$

The heating element supply for each building group is given by:

$$\eta \cdot \boldsymbol{d_{\omega,g,u,t}^{he}} \leq \boldsymbol{c_{g,i=HE_u}} \cdot d^{he,max} \qquad \forall\omega, \forall g, \forall u,\ \forall t, \quad \text{(A.6)}$$

and the heat pump supply by:

$$COP_{\omega,u,t} \cdot \boldsymbol{d_{\omega,g,u,t}^{hp}} = \frac{1}{m} \cdot d_{\omega,t}^{hp,max} \cdot \boldsymbol{z_{\omega,g,u,t}} \qquad \forall\omega, \forall g, \forall u, \forall t, \quad \text{(A.7)}$$

$$\boldsymbol{z_{\omega,g,u=DHW,t}} \leq m \cdot \boldsymbol{c_{g,i=HP_{DHW}}} \qquad \forall\omega\ \forall g\ \forall t, \quad \text{(A.8)}$$

$$\sum_{u=1}^{2} \boldsymbol{z_{\omega,g,u,t}} \leq m \cdot \sum_{u=1}^{2} \boldsymbol{c_{g,i=HP_u}} \qquad \forall\omega, \forall g, \forall t. \quad \text{(A.9)}$$

If heat pumps run only at idle, half or full load, then $m = 2$ with $z_{\omega,g,u=SH,t} \in \{0,1,2,3,4\}$ and $z_{\omega,g,u=DHW,t} \in \{0,1,2\}$, otherwise $z_{\omega,g,u=SH,t}, z_{\omega,g,u=DHW,t} \in \mathbb{R}_+$. The following constraint equals the element of the first and last time step t:

$$\boldsymbol{s_{\omega,g,u,t=T}} = \boldsymbol{s_{\omega,g,u,t=1}} \qquad \forall\omega, \forall g, \forall u, \quad \text{(A.10)}$$

$$\boldsymbol{z_{\omega,g,u,t=T}} = \boldsymbol{z_{\omega,g,u,t=1}} \qquad \forall\omega, \forall g, \forall u. \quad \text{(A.11)}$$

All presented variables need to be positive:

$$\begin{aligned} \boldsymbol{c_{g,i}}, \boldsymbol{e_{\omega,t}^{grid}}, \boldsymbol{e_{\omega,t}^{fi}}, \boldsymbol{q_{\omega,g,u,t}}, \boldsymbol{d_{\omega,g,u,t}^{hp}}, \boldsymbol{d_{\omega,g,u,t}^{he}}, \boldsymbol{s_{\omega,g,u,t}}, \boldsymbol{L_{\omega,g,u,t}}, \boldsymbol{pos_{\omega,g,u,t}}, \\ \boldsymbol{neg_{\omega,g,u,t}}, \boldsymbol{z_{\omega,g,u,t}} \geq 0 \end{aligned} \qquad \forall\omega, \forall g, \forall t. \quad \text{(A.12)}$$

Table A.1: Nomenclature of the residential quarter modeled as a two-stage stochastic program.

parameters	
ANF	annuity factor
$cost_i$	variable capacity costs of component i plus a fix amount
$COP_{\omega,u,t}$	COP of the heat pump in scenario ω of building group g for use u at time t
$d_{\omega,t}^{hp,max}$	maximal heating power of the heat pump at time t
$d^{he,max}$	maximal heating power of the heating element
$d_{\omega,t}^{ee}$	electricity demand for electrical usage in scenario ω of building group g at time t
$d_{\omega,g,u,t}$	thermal demand in scenario ω of building group g for use u at time t
$e_{\omega,t}^{pv,kwp}$	supplied electrical energy per kilowatt-peak of the PV system in scenario ω at time t
$e_{\omega,t}^{pv}$	supplied electrical energy from the PV system in scenario ω at time t
f	compensation factor for not-covered heat demand
l_u	loss factor of heat storage for use u
m	possible power modes of the heat pump
r_u	ramp-up loss factor of heat pump for use u
p^{grid}	price of electricity from grid
p^{fi}	price of feed-in compensation
η	efficiency of the heating element
variables	
$\boldsymbol{c_{g,i}}$	capacity of building group g of component i
$\boldsymbol{c_{g,i=PV}}$	installed PV capacity of building group g
$\boldsymbol{c_{g,i=HP_{SH}}}$	number of heat pumps of building group g for SH
$\boldsymbol{c_{g,i=HP_{DHW}}}$	number of heat pumps of building group g for DHW
$\boldsymbol{c_{g,i=HE_{SH}}}$	number of heating elements of building group g for SH storage
$\boldsymbol{c_{g,i=HE_{DHW}}}$	number of heating elements of building group g for DHW storage
$\boldsymbol{c_{g,i=S_{SH}}}$	maximal capacity of heat storage of building group g for SH
$\boldsymbol{c_{g,i=S_{DHW}}}$	maximal capacity of heat storage of building group g for DHW
$\boldsymbol{d_{\omega,g,u,t}^{hp}}$	used electricity of heat pump in scenario ω of building group g for use u at time t
$\boldsymbol{d_{\omega,g,u,t}^{he}}$	used electricity of heating element in scenario ω of building group g for use u at time t
$\boldsymbol{e_{\omega,t}^{grid}}$	used electricity from the grid in scenario ω at time t
$\boldsymbol{e_{\omega,t}^{fi}}$	fed-in energy of the PV system in scenario ω at time t
$\boldsymbol{pos_{\omega,g,u,t}}$	pos. variable for positive shift of heat pump in scenario ω of building group g for use u at time t
$\boldsymbol{neg_{\omega,g,u,t}}$	pos. variable for negative shift of heat pump in scenario ω of building group g for use u at time t
$\boldsymbol{q_{\omega,g,u,t}}$	not-covered heat demand in scenario ω of building group g for use u at time t
$\boldsymbol{s_{\omega,g,u,t}}$	stored heat in scenario ω of building group g for use u at time t
$\boldsymbol{s_{g,u}^{min}}$	minimal heat storage level of building group g for use u
$\boldsymbol{z_{\omega,g,u,t}}$	integer/continuous heating power level in scenario ω of building group g for use u at time t
indices	
g	building group $1,..,G$ of the quarter with $G = 4$
i	component $i \in \{PV, HP_{SH}, HP_{DHW}, HE_{SH}, HE_{DHW}, S_{SH}, S_{DHW}\}$ of the energy system with $\lvert i\rvert = k_1 = 7$
u	use $u \in \{SH, DHW\}$ for space heating or domestic hot water with $\lvert u\rvert = 2$
t	time index $1,..,T$ indicating the time step of the year
ω	scenario index $1,..,N$

Appendix B

Table B.1 lists the three parameters that have the largest effect on performance and their mean wallclock time reduction, itemized in detail for all 27 partitions.

Table B.1: Results of the ablation analysis listing the three parameters that have the largest effect on performance altered from the default to the partition-specific optimized configuration of CPLEX. The parameters are determined by SMAC on 9 instances for the given 27 partitions. The values in the brackets show the mean wallclock time of the 9 instances to achieve a MILP gap of at most 0.6%, whereby unsuccessful runs that could not be solved within the cutoff-time of 1 800s are counted as having taken 18 000s.

	P-0_Default →	P-1 →	P-2 →	P-3 →	**P-ALL_SMAC**
Partition 1	**(1498s)**	Mip cuts flowcovers (661s)	Read scale (466s)	Mip strategy bbinterval (466s)	**(180s)**
Partition 2	**(1367s)**	Preprocessing reduce (492s)	Preprocessing repeatpresolve (467s)	Preprocessing fill (298s)	**(125s)**
Partition 3	**(1498s)**	Mip strategy rinsheur (258s)	Mip limits submipnodelim (188s)	Mip strategy heuristicfreq (167s)	**(167s)**
Partition 4	**(1709s)**	Mip strategy rinsheur (221s)	Feasopt mode (203s)	Barrier limits growth (203s)	**(154s)**
Partition 5	**(1784s)**	Mip strategy rinsheur (1551s)	Mip limits aggforcut (1106s)	Mip cuts covers (996s)	**(901s)**
Partition 6	**(1800s)**	Mip strategy rinsheur (1665s)	Preprocessing linear (1639s)	no improvment (1639s)	**(1045s)**
Partition 7	**(1800s)**	Mip strategy rinsheur (1675s)	Mip limits submipnodelim (1637s)	no improvment (1637s)	**(1637s)**
Partition 8	**(1800s)**	no improvements for P-1, P-2, P-3 and P-ALL			
Partition 9	**(1800s)**	Simplex pgradient (1625s)	Barrier crossover (1571s)	Mp limits aggforcut (1555s)	**(1489s)**
Partition 10	**(1785s)**	Mip strategy rinsheur (1268s)	Mip limits gomorycand (1231s)	Preprocessing repeatpresolve (1068s)	**(893s)**
Partition 11	**(1706s)**	Mip strategy rinsheur (1081s)	Mip limits aggforcut (366s)	Mp strategy heuristicfreq (180s)	**(180s)**
Partition 12	**(1432s)**	Mip strategy nodeselect (982s)	Mip strategy rinsheur (898s)	Mp limits aggforcut (383s)	**(383s)**
Partition 13	**(1729s)**	Mip strategy rinsheur (725s)	Mip limits gomorycand (169s)	Preprocessing linear (169s)	**(169s)**
Partition 14	**(1348s)**	Preprocessing reduce (171s)	Mip strategy nodeselect (138s)	Mip strategy variableselect (131s)	**(82s)**
Partition 15	**(1149s)**	Mip strategy nodeselect (324s)	Mip strategy rinsheur (164s)	Mip limits submipnodelim (115s)	**(66s)**
Partition 16	**(1363s)**	Mip strategy heuristicfreq (143s)	Mip limits aggforcut (95s)	Mip cuts gubcovers (95s)	**(63s)**
Partition 17	**(1274s)**	Mip strategy nodeselect (262s)	Mip limits aggforcut (117s)	Simplex limits perturbation (96s)	**(74s)**
Partition 18	**(1466s)**	Mip strategy rinsheur (562s)	Mip strategy startalgorithm (533s)	Network pricing (327s)	**(327s)**
Partition 19	**(1755s)**	Mip strategy rinsheur (784s)	Preprocessing repeatpresolve (266s)	Mip strategy nodeselect (225s)	**(149s)**
Partition 20	**(1759s)**	Mip strategy rinsheur (1075s)	Mip strategy nodeselect (797s)	Preprocessing reduce (486s)	**(486s)**
Partition 21	**(1800s)**	no improvements for P-1, P-2 and P-3			**(891s)**
Partition 22	**(1796s)**	Mip strategy rinsheur (1294s)	Mip strategy nodeselect (624s)	Preprocessing fill (589s)	**(589s)**
Partition 23	**(1762s)**	Mip strategy rinsheur (1043s)	Mip strategy nodeselect (773s)	Mp limits aggforcut (472s)	**(472s)**
Partition 24	**(1604s)**	Mip strategy rinsheur (922s)	Preprocessing reduce (729s)	Mip strategy nodeselect (473s)	**(473s)**
Partition 25	**(1480s)**	Emphasis numerical (139s)	Mip cuts gomory (117s)	Perturbation constant (114s)	**(75s)**
Partition 26	**(1338s)**	Mip strategy rinsheur (361s)	Mip limits submipnodelim (126s)	Mip strategy nodeselect (121s)	**(97s)**
Partition 27	**(411s)**	Mip strategy heuristicfreq (205s)	Emphasis mip (204s)	Read scale (203s)	**(22s)**

References

Altmann, M., Brenninkmeijer A, Lanoix J-C, Ellison D, Crisan A, Hugyecz A, et al. (2010). Decentralized energy systems. Tech. rep. European Parliament's Committee (ITRE). http://www.europarl.europa.eu/document/activities/cont/201106/20110629ATT22897/20110629ATT22897EN.pdf. Accessed 29 September 2016.

Andriosopoulos, K., Zopounidis, C., & Doumpos, M. (2016). Editorial to the Special Issue "OR in energy modeling and management". *Computers & Operations Research, 66*, 225–227 (2016). doi:10.1016/j.cor.2015.11.005

Ansótegui, C., Sellmann, M., & Tierney, K. (2009). A gender-based genetic algorithm for the automatic configuration of algorithms. In D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, et al. (Eds.), *Principles and Practice of Constraint Programming - CP 2009* (Vol. 5732, pp. 142–157, Lecture Notes in Computer Science). Berlin, Heidelberg: Springer.

Atamtürk, A., & Savelsbergh, M. W. P. (2005). Integer-Programming Software Systems. *Annals of Operations Research, 140*, 67–124 (2005). doi:10.1007/s10479-005-3968-2

Beraldi, P., Conforti, D., & Violi, A. (2008). A two-stage stochastic programming model for electric energy producers. *Computers & Operations Research, 35*, 3360–3370 (2008). doi:10.1016/j.cor.2007.03.008

Bergstra, J. S., Bardenet, R., Bengio, Y., & Kégl, B. (2011). Algorithms for Hyper-Parameter Optimization. In J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira, & K. Q. Weinberger (Eds.), *Advances in Neural Information Processing Systems 24* (pp. 2546–2554): Curran Associates, Inc.

Dantzig, G. B. (1955). Linear programming under uncertainty. *Management Science, 1*, 197–206.

Fawcett, C., & Hoos, H. H. (2016). Analysing differences between algorithm configurations through ablation. *Journal of Heuristics, 22*, 431–458 (2016). doi:10.1007/s10732-014-9275-9

Fragnière, E., Gondzio, J., & Vial, J.-P. (2000). Building and Solving Large-Scale Stochastic Programs on an Affordable Distributed Computing System. *Annals of Operations Research, 99*, 167–187 (2000). doi:10.1023/A:1019245101545

Hutter, F., Babic, D., Hoos, H. H., & Hu, A. J. Boosting verification by automatic tuning of decision procedures. In *Austin, TX, USA* (pp. 27–34). doi:10.1109/FAMCAD.2007.9

Hutter, F., Hoos, H. H., & Leyton-Brown, K. (2010). Automated configuration of mixed integer programming solvers. In D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, et al. (Eds.), *Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems* (Vol. 6140, pp. 186–202, Lecture Notes in Computer Science). Berlin, Heidelberg: Springer Berlin Heidelberg.

Hutter, F., Hoos, H. H., & Leyton-Brown, K. (2011). Sequential model-based optimization for general algorithm configuration. In D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, et al. (Eds.), *Learning and Intelligent Optimization* (Vol. 6683, pp. 507–523, Lecture Notes in Computer Science). Berlin, Heidelberg: Springer.

Hutter, F., Hoos, H. H., Leyton-Brown, K., & Stützle, T. (2009). ParamILS: an automatic algorithm configuration framework. *Journal of Artificial Intelligence Research, 36*, 267–306 (2009). doi:10.1613/jair.2808

IBM. (2016). ILOG CPLEX Optimization Studio: CPLEX user's manual, version 12 release 6. http://www.ibm.com/support/knowledgecenter/en/SSSA5P_12.6.1/ilog.odms.studio.help/pdf/usrcplex.pdf. Accessed 3 June 2016.

Jones, D. R., Schonlau, M., & Welch, W. J. (1998). Efficient global optimization of expensive black-box functions. *Journal of Global Optimization, 13*, 455–492 (1998). doi:10.1023/A:1008306431147

Kobayakawa, T., & Kandpal, T. C. (2016). Optimal resource integration in a decentralized renewable energy system: Assessment of the existing system and simulation for its expansion. *Energy for Sustainable Development, 34*, 20–29 (2016). doi:10.1016/j.esd.2016.06.006

Kuznia, L., Zeng, B., Centeno, G., & Miao, Z. (2013). Stochastic optimization for power system configuration with renewable energy in remote areas. *Annals of Operations Research, 210*, 411–432 (2013). doi:10.1007/s10479-012-1110-9

Mascia, F., López-Ibáñez, M., Dubois-Lacoste, J., & Stützle, T. (2014). Grammar-based generation of stochastic local search heuristics through automatic algorithm configuration tools. *Computers & Operations Research, 51*, 190–199 (2014). doi:10.1016/j.cor.2014.05.020

Owens, B. (2014). The rise of distributed power. General Electric (ecomagination). https://www.ge.com/sites/default/files/2014%2002%20Rise%20of%20Distributed%20Power.pdf. Accessed 30 September 2016.

Prékopa, A. (1995). *Stochastic Programming*. Dordrecht: Springer Netherlands.

Prékopa, A., Ganzer, S., Deák, I., & Patyi, K. (1980). The STABIL stochastic programming model and its experimental application to the electrical energy Sector of the Hungarian economy. In M. A. H. Dempster (Ed.), *Stochastic programming*. London: Academic Press.

Schwarz, H., Bertsch, V., & Fichtner, W. (2015). *Two-stage stochastic, large-scale optimization of a decentralized energy system - a residential quarter as case study* (Working Paper Series in Production and Energy, Vol. 10). Karlsruhe: KIT.

Shapiro, A., Dentcheva, D., & Ruszczynski, A. P. (2009). *Lectures on stochastic programming: modeling and theory* (MPS-SIAM series on optimization, Vol. 9). Philadelphia Pa.: SIAM.

Snoek, J., Larochelle, H., & Adams, R. P. (2012). Practical Bayesian Optimization of Machine Learning Algorithms. In F. Pereira, C. J. C. Burges, L. Bottou, & K. Q. Weinberger (Eds.), *Advances in Neural Information Processing Systems 25* (pp. 2951–2959): Curran Associates, Inc.

Taborda, D., & Zdravkovic, L. (2012). Application of a hill-climbing technique to the formulation of a new cyclic nonlinear elastic constitutive model. *Computers and Geotechnics, 43*, 80–91 (2012). doi:10.1016/j.compgeo.2012.02.001

Velik, R., & Nicolay, P. (2016). Energy management in storage-augmented, grid-connected prosumer buildings and neighborhoods using a modified simulated annealing optimization. *Computers & Operations Research, 66*, 248–257 (2016). doi:10.1016/j.cor.2015.03.002

Wallace, S. W., & Fleten, S.-E. (2003). Stochastic programming models in energy. In *Stochastic Programming* (Vol. 10, pp. 637–677, Handbooks in Operations Research and Management Science): Elsevier.

Wolfe, P. (2008). The implications of an increasingly decentralised energy system. *Energy Policy, 36*, 4509–4513 (2008). doi:10.1016/j.enpol.2008.09.021

Yazdanie, M., Densing, M., & Wokaun, A. (2016). The role of decentralized generation and storage technologies in future energy systems planning for a rural agglomeration in Switzerland. *Energy Policy, 96*, 432–445 (2016). doi:10.1016/j.enpol.2016.06.010

Working Paper Series in Production and Energy

recent issues

No. 1 Alexandra-Gwyn Paetz, Lisa Landzettel, Patrick Jochem, Wolf Fichtner:
Eine netnografische Analyse der Nutzererfahrungen mit E-Rollern

No. 2 Felix Teufel, Michael Miller, Massimo Genoese, Wolf Fichtner:
Review of System Dynamics models for electricity market simulations

No. 3 Patrick Jochem, Thomas Kaschub, Wolf Fichtner:
How to integrate electric vehicles in the future energy system?

No. 4 Sven Killinger, Kai Mainzer, Russell McKenna, Niklas Kreifels, Wolf Fichtner:
A regional simulation and optimization of renewable energy supply from wind and photovoltaics with respect to three key energy-political objectives

No. 5 Kathrin Dudenhöffer, Rahul Arora, Alizée Diverrez, Axel Ensslen, Patrick Jochem, Jasmin Tücking:
Potentials for Electric Vehicles in France, Germany, and India

No. 6 Russell McKenna, Carsten Herbes, Wolf Fichtner:
Energieautarkie: Definitionen, Für- bzw. Gegenargumente, und entstehende Forschungsbedarfe

No. 7 Tobias Jäger, Russell McKenna, Wolf Fichtner:
Onshore wind energy in Baden-Württemberg: a bottom-up economic assessment of the socio-technical potential

No. 8 Axel Ensslen, Alexandra-Gwyn Paetz, Sonja Babrowski, Patrick Jochem, Wolf Fichtner:
On the road to an electric mobility mass market - How can early adopters be characterized?

No. 9 Kai Mainzer, Russell McKenna, Wolf Fichtner:
Charakterisierung der verwendeten Modellansätze im Wettbewerb Energieeffiziente Stadt

No. 10 Hannes Schwarz, Valentin Bertsch, Wolf Fichtner:
Two-stage stochastic, large-scale optimization of a decentralized energy system – a residential quarter as case study

No. 11 Leon Hofmann, Russell McKenna, Wolf Fichtner:
Development of a multi-energy residential service demand model for evaluation of prosumers‘ effects on current and future residential load profiles for heat and electricity

No. 12 Russell McKenna, Erik Merkel, Wolf Fichtner:
Energy autonomy in residential buildings: a techno-economic model-based analysis of the scale effects

No. 13 Johannes Schäuble, Silvia Balaban, Peter Krasselt, Patrick Jochem, Mahmut Özkan, Friederike Schnellhas-Mende, Wolf Fichtner, Thomas Leibfried, Oliver Raabe:
Vergleichsstudie von Systemansätzen für das Schnelladen von Elektrofahrzeugen

Working Paper Series in Production and Energy

recent issues

No. 14 Marian Hayn, Valentin Bertsch, Anne Zander, Stefan Nickel, Wolf Fichtner:
The impact of electricity tariffs on residential demand side flexibility

No. 15 Erik Merkel, Robert Kunze, Russel McKenna, Wolf Fichtner:
Modellgestützte Bewertung des Kraft-Wärme-Kopplungsgesetzes 2016 anhand ausgewählter Anwendungsfälle in Wohngebäuden

No. 16 Russell McKenna, Valentin Bertsch, Kai Mainzer, Wolf Fichtner:
Combining local preferences with multi-criteria decision analysis and linear optimisation to develop feasible energy concepts in small communities

No. 17 Tilman Apitzsch, Christian Klöffer, Patrick Jochem, Martin Doppelbauer, Wolf Fichtner:
Metaheuristics for online drive train efficiency optimization in electric vehicles

No. 18 Felix Hübner, Georg von Grone, Frank Schultmann:
Technologien zur Zerlegung und zur Dekontamination von kerntechnischen Anlagen

No. 19 Felix Hübner, Jennifer Jana Jung, Frank Schultmann:
Gefahren ionisierender Strahlung für Mensch und Umwelt in Bezug auf kerntechnische Anlagen

No. 20 Juri Lüth, Tobias Jäger, Russell McKenna, Wolf Fichtner:
Photovoltaik auf Gebäuden: eine GIS-gestützte Ermittlung des Potenzials in Baden-Württemberg

No. 21 Felix Hübner, Jennifer Jana Jung, Frank Schultmann:
Auswirkungen nuklearer Unfälle auf den Menschen und die Umwelt

No. 22 Felix Hübner, Ulli Schellenbaum, Christian Stürck; Patrick Gerhards, Frank Schultmann:
Evaluation von Schedulingproblemen für die Projektplanung von Großprojekten am Beispiel des kerntechnischen Rückbaus

No. 23 Martin Hain, Hans Schermeyer, Marliese Uhrig-Homburg, Wolf Fichtner:
An Electricity Price Modeling Framework for Renewable-Dominant Markets

No. 24 Hannes Schwarz, Lars Kotthoff, Holger Hoos, Wolf Fichtner and Valentin Bertsch:
Using automated algorithm configuration to improve the optimization of decentralized energy systems modeled as large-scale, two-stage stochastic programs

Impressum

Karlsruher Institut für Technologie

Institut für Industriebetriebslehre und Industrielle Produktion (IIP)
Deutsch-Französisches Institut für Umweltforschung (DFIU)

Hertzstr. 16
D-76187 Karlsruhe

KIT – Universität des Landes Baden-Württemberg und
nationales Forschungszentrum in der Helmholtz-Gemeinschaft

Working Paper Series in Production and Energy
No. 24, July 2017

ISSN 2196-7296

www.iip.kit.edu