A Robust Decentralised Agent Based Approach for Microgrid Energy Management

Garcia-Rodriguez, Sandra
CEA, LIST
Gif Sur Yvette, France
sandra.garciarodriguez@cea.fr

Gomez-Sanz, Jorge J.
Universidad Complutense de Madrid
Madrid, Spain
jjgomez@ucm.es

ABSTRACT

This paper introduces a decentralized agent based implementation of an off-line optimization technique applied to the configuration of electrical microgrids with distributed generation. This implementation conforms the "Agent Based Energy Management System" (ABEMS). Software agent principles are used to decentralize the microgrid control without having an implicit or explicit control hierarchy. Unlike most approaches, they work as peers. Orders to be sent to the energy micro-generation units are produced off-line by an evolutionary computation method. The orders are encoded into the agents once the micro-grid is started and agents decide when to apply them. The paper includes experiments that show the resulting behavior in several scenarios executed in real time. Results indicate the technique is stable enough and outperforms more simple approaches, a necessary initial step prior to a more comprehensive comparison with other solutions.

KEYWORDS

Multi-Agent System; Smart Grid; Optimisation; Decentralization

ACM Reference Format:
Garcia-Rodriguez, Sandra and Gomez-Sanz, Jorge J.. 2019. A Robust Decentralised Agent Based Approach for Microgrid Energy Management. In *Proc. of the 18th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2019), Montreal, Canada, May 13–17, 2019,* IFAAMAS, 9 pages.

1 INTRODUCTION

In the last few years, power grids have gone through several changes to make them work as "Smart Grids". For instance, several elements have been added such as sensors and meters, network nodes with computation capabilities, switches or actuators, to cite some. Together, they allow the grid setup to be highly configurable [6]. To increase manageability, smaller grids can be distinguished within a larger grid. Such sub-systems are called "microgrids" and consist of energy consumers and producers at a small scale that are able to manage themselves [30]. Those will be the ones considered in this paper.

Smart grids are expected to make decisions while controlling the grid. They must achieve optimal results according to one (e.g. reduce the bill using your own generators) or many, sometimes conflicting, goals (e.g. reduce maintenance costs by not always using your own generators). Optimization methods for grids are known and can be applied when the microgrid constituents do not change. For instance, Georgilakis and Hatziarguriou [7] introduce a comparative of many optimization methods to choose the best distribution (and sizes) of generators in order to optimize electrical distribution network operation minimizing as possible system losses. While it is possible to attain an optimal configuration, maintaining optimality is another open problem. Weather and/or consumers' demand patterns can change. Elements in the grid can fail. New generation sources could be added or removed (e.g. someone installing an unauthorized solar panel). Therefore, different ways for adjusting the decisions to the new situations are needed.

On the other hand, a smart-grid, beyond augmented sensing capabilities, is also characterized by the allocation of computing nodes (e.g. system on chip boards). An important limitation is the computational power of those nodes. CPUs are usually ARM processors with a few hundred Mhz and kilobytes of memory available. Thus, a research question is how to use this limited computing capability to help a smart grid to adjust its operation parameters to the new operation conditions.

Agent technology has been identified as a key solution to address this problem [25]. However, existing agent based approaches have brought the distribution vs. decentralization confrontation [9, 29]. If the centralized approach concentrates the different grid control functions in a single node, the distributed approach simply moves those functions out to other different computational nodes. Despite the change, they still behave as a centralized node with one central element distributing instructions to the others, possibly following a hierarchy. Decentralization, however, distributes the control itself, in a way that computational nodes become peers. There is no central node issuing orders and all nodes act in a coordinated way. Certainly, a decentralized solution (with several control nodes sharing the same authority and collaborating in a peer to peer fashion) is harder to design, but it can be more flexible in handling modifications to the grid and less prompt to failures.

The contribution of this paper is a proof of concept that deciding in a decentralized way which configuration is better can be done with a decentralized multi-agent system. Therefore, the paper introduces an agent based design, the "Agent Based Energy Management System" (ABEMS), that performs a decentralized execution of optimal configurations obtained in a off-line fashion. ABEMS also borrows some concepts of "any time learning" approach presented by [12], since it performs a continuous learning when environment changes.

Off-line solutions are opted here because the available hardware that was expected to host the agents was just not powerful enough. The original project considered only ARM processors with a few hundred MHz and few kilobytes of RAM. Besides, the extra off-line computation time is affordable.

Proc. of the 18th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2019), N. Agmon, M. E. Taylor, E. Elkind, M. Veloso (eds.), May 13–17, 2019, Montreal, Canada.

From the optimization state of art, multi-objective evolutionary techniques are the most suited for this task due to its capacity of optimizing several objectives at the same time and their flexibility to deal with soft and hard constraints. However, the optimization algorithm itself is not essential here. It could be replaced by any other available approach as long as it is off-line. It proves that it is possible to decentralize the decision making while avoiding oscillations in the micro-grid's energy demand and preserving the goals of the original optimization strategy.

This paper is organized as follows. The section 2 studies the current state of the art in optimization methods using evolutionary computation and the approaches to decentralization using software agents technology. Then section 3 describes the main characteristics of ABEMS system and deeply studies the optimization approach designed in this work. Section 4 shows the experiments for training and testing that have been carried out. It also includes comments on the obtained results. Finally, there are conclusions and future lines in section 5.

2 RELATED WORK

The contribution of this paper relates to work done in the optimization of microgrids control, but also to the decentralized and adaptive control solutions for microgrids.

In what refers to optimization techniques, the problem may be addressed as a multi-objective optimization using evolutionary computing techniques. It means that the improvement of a single objective worsen the others; so the goal is to obtain the best trade-off among objectives.

Decentralization is another concern in this work, since the number of computing nodes in a smart grid is higher. However, most authors deliver distributed solutions with hierarchies of control components, rather than decentralized peer-based control alternatives. A key clue to recognize them is the existence of some distinguished agent types of which only one instance is allowed.

2.1 Optimization and Microgrids

In the optimization part, microgrids usually have several objectives to satisfy that are often in conflict. This means that improving any of the objectives makes the others worse. Since there is not any single best solution that satisfies all the function objectives at the same time, multiobjective procedures provide a set of solutions with the best trade-off among objectives. However, literature on microgrid control has several works on single objective optimization too, like [3], [23] or [27] who use the Quadratic Programming approach to optimize. Unfortunately, these works do not focus on the operation of such a grid. They provide optimal configuration of the production considering some demand without taking into account other strong constraints derived from microgeneration. This is not a coincidence, since a Quadratic Programming problem that takes into account the necessary variables to operate a microgrid may not be feasible. The work Schafer and Moser [27] apply decomposition to overcome this complexity in part, but it remains to be proven the adequacy of this kind of solution for real time operation.

Among soft-computing solutions, NSGA-II [4] has been repeatedly used in the literature related to optimization of microgrid configuration and multi-objective optimization. It represents a new version of the NSGA algorithm with a complexity of $O(M \cdot N^2)$; with M the number of objectives and N the population size. It is a generational genetic algorithm based on obtaining an auxiliary population from the original one by applying the typical genetic operators (selection, crossover and mutation). Then, the two populations are merged and the individuals are sorted according to their rank. Inside each of these ranks, the crowding distance is used to sort the individuals from less to more crowded. A solution with a smaller value of this distance measure is, in some sense, more crowded by other solutions. Finally, the best solutions are selected to compose the new population that will be used to create the new offspring population.

Most authors deal with a stable configuration of the grid without considering that actually conditions change. Literature in this area is wide. Ramaswamy and Deconinck [24] use Simple Genetic Algorithm (GA) and the above mentioned NSGA-II [4] to solve the smart grid reconfiguration problem. Some goals are conflicting ones, in particular, providing minimum sum of voltage deviation conflicts the minimum power loss. For those cases, authors acknowledge GA solution with weighted sum of objective evaluation functions works less well than NSGA-II, which was devised precisely to deal with multi-objective computations. NSGA-II has also been used to solve the problem of reconfiguration by minimizing active power losses and system average interruption frequency index [31]. Authors generate the initial population using a branch-exchange heuristic algorithm. Another work is the one presented by Lopez et al. [20] who use this algorithm to solve the optimal distributed generation placement problem. In this case the authors optimize the minimum installation costs and expected energy not supplied.

Replanning the microgrid setup is less frequent. San Severino et al. [26] utilize the NSGA-II as well to develop a replanning module of distributed generation units to minimize the system energy losses, fuel consumption cost and CO_2 emissions. They improved the initial plan based on forecasts to consider new information of the meters and sending it to the microgrid central controller. Nevertheless, they use a microgrid central controller.

NSGA-II has been also used to solve the problem of reconfiguration by minimizing active power losses and system average interruption frequency index [31]. Authors generate the initial population using a branch-exchange heuristic algorithm. Another work is the one presented by Lopez et al. [20] who use this algorithm to solve the optimal distributed generation placement problem. In this case the authors optimize the minimum installation costs and expected energy not supplied.

In [22] and [21], the authors use another Multiobjective Optimization (MO) algorithm to determine the optimal operating strategy. The microgrid modelled consists of windturbine, micro turbine, diesel generator, photo-voltaic array, fuel cell and batteries (added in the second paper). They minimize operation costs and the emissions but always ensuring the load demand.

As a conclusion, the reviewed works use optimization through metaheuristics. They mostly fall into one of these two categories: solutions that transform the multiobjective problem into a mono-objective form, and those who deal with it using a mutiobjective approach. In the first case, it is assumed that there is a chance of combination of these objectives in a single objective function, such

problems can be solved using conventional single-objective optimization methods. In the second case, there is no easy, or proper, way to combine the problem's objectives; this happens whenever objectives are conflicting, that is, improving some of the objectives means worsening others. In these situations a Multiobjective Optimization problem needs to be solved. Such multiobjective approach is frequent in the literature where the solution to this kind of problems is known as the pareto-optimal front [8]. Within this field, Multiobjective Evolutionary Algorithms (MOEAs) have been proved to be specially suited for this kind of optimization task, as it is the NSGA-II algorithm [4], which will be chosen as example for this paper.

2.2 Decentralized Control

Literature on agents applied to microgrids tend to suggest distributed/centralized solutions, with central-node-agents upon which a hierarchy of other slave-agents is arranged. A key clue to recognize them is the existence of some distinguished agent types of which only one instance is allowed. Though the solution works and can be easily engineered, it adds little with respect to more conservative technologies. Following the classical agenthood test [17], where the behavior of the system is improved after adding another agent of the same type, the question is whether the system performs better if there are two, or more, central nodes or if there are two, or more, hierarchies within the same system. Examples of a distributed rather than decentralized approaches are Katirei et al. [18], Evora et al. [5], and Logenthiran et al. [19].

Katirei et al. [18] explain a decentralized control that intends to give DER units and loads as much autonomy as possible. Both DER and loads exchange information to coordinate their actions and achieve some common goals. Agent technology is suggested as a candidate to achieve this kind of behaviors. The authors propose a hierarchy of agents distinguishing those in charge of the grid, those with management functions, and those at the field level. Anyway, authors do not specify how to achieve the kind of intelligence or how to achieve the suggested coordination capabilities.

Evora et al. [5] use multi-objective particle swarm optimisation to achieve decentralised production and control. Particle Swarm Optimization (PSO) is another kind of evolutionary metaheuristic that produces results similar to NSGA-II. Authors assume a hierarchical structure where a commander issues orders to intermediaries so that the orders can be propagated downwards. They propose PSO to be applied continuously to determine what to do next. Conflicting goals are to make the production meets the demand; to focus control in those appliances that have been controlled a few times; and keep controlling current appliances before starting to control new ones. The approach is not decentralized, but hierarchical with a central node, and depends on a relatively high computational capabilities of the commander.

Logenthiran et al. [19] present a distributed multiagent system (MAS) for real-time operation of a microgrid. Agents in this system belong to one of the three major groups: a deliberative layer, a coordination layer, and a reactive layer. Despite some decentralization, there are still critical centralized parts, like the single agent responsible of controlling and managing the microgrid, or the single agent which uses genetic algorithms for demand side management. Also, authors are not discussing how this system is supposed to evolve or to incorporate new elements.

Decentralized approaches are not frequently found in the literature. Examples of decentralized solutions are works like [14] or [13]. They get the intended agent-like behavior by using the appropriate hardware control circuits. As working hypothesis, an implicit coordination pointed out by Katirei et al. [18] may arise just by reading shared media information with specialized hardware, like current and voltage in Guerrero et al. [14]. However, approaches like this are harder to evolve without replacing the circuits. Furthermore, hardwired control circuits may not be able to be changed or learn. This work is further developed in Guerrero et al. [15], but load planning and scheduling is not considered. Authors focus on the grid stability, which is not the issue of this contribution.

Similarly, Gu et al. [13] propose that each DER has its own controller. They also propose that all DER controllers can be in either utility dominating mode, storage dominating mode, or generation dominating mode. The idea is to make independent control decisions based on the local information. It uses a variation of the droop method, which sets the voltage threshold at which each source/storage interface converter becomes active [28]. This method is harder to apply for dealing with failures in nodes or complex scenarios where variables different from voltage or current are taken into account.

Decentralized planning of the operations to perform in smartgrids is an open problem [25]. Therefore, the challenge is obtaining a decentralized control that has agent-like capabilities and is still flexible enough to consider different factors to achieve decisions. This paper's contribution pursues this goal using software and computation nodes able to process it, rather than hardware control boards. The trade-off is the reaction time, but software response can be good enough [14].

3 THE AGENT BASED ENERGY MANAGEMENT SYSTEM

This section introduces the *Agent Based Energy Management System* (ABEMS) (see figure 1). It is composed by two main components, the *Optimization Training Framework* or **OpTF** and the *Energy Management Framework* or **EMaF**. The first is responsible of determining, off-line, how agents should behave in the different scenarios. The second enables this behavior during the microgrid operation.

This paper distinguishes between two concepts: **Scenario** as the specification of the microgrid topology and its elements (consumers, generators, lines, connections, etc.) with their properties (including operational status) and constraints; and **situation** (or **profile**), that represents a set of particular values that describe the conditions in which the microgrid has to work as well as the status of its elements, including the transport grid; i.e. weather conditions that set the maximum theoretical power in photo-voltaic systems or wind-turbines at each instant. The combination of all possible values of these factors would generate a huge number of situations, however just few of them would be feasible real-world ones.

ABEMS can pursue many, possibly conflicting, optimization goals in a microgrid for several possible *scenarios* and *situations*. For that purpose, ABEMS proposes a procedure that identifies: (i) what actions each agent must perform to fulfill those system goals; and

(ii) when such actions ought to be initiated. The sets of actions and conditions that indicate when to trigger them conform a *catalog*, which is key to the approach. The procedure that creates this *catalog* is based on evolutionary computation and it is executed off-line. It takes into account a variety of scenarios: unexpected changes in the topology such as the disconnection of any element (DER or load); defective/non-existing/delayed measurements of the grid status; variable weather conditions that affect the energy produced by some DERs; or operation constraints/preferences. The *catalog* is distributed among the agents in the infrastructure according to their responsibilities. Once distributed, each agent follows its instructions and triggers actions depending on the trained indicators. Coordination among agents emerges as a result of the trained actions and will be as good as extensive the training procedure is. The more trained scenarios the more versatile the agents will be.

ABEMS bases on the GRIDLAB-D [2] for discovering the effects of the instructions sent to the different DERs of the microgrid. GRIDLAB-D is a fast discrete event simulator very convenient for evaluating lots of scenarios. For performance evaluation, however, a *real time* simulation is needed, where agents can interact with the simulation not bounded by the simulation cycle and with the uncertainty of the outcome of the results. This service can be emulated using the same GRIDLAB-D [10, 11]. It is possible to conveniently invoke GRIDLAB-D to obtain order execution data over long periods of simulated time each time an agent performs an action. This data can be reproduced in real time later on.

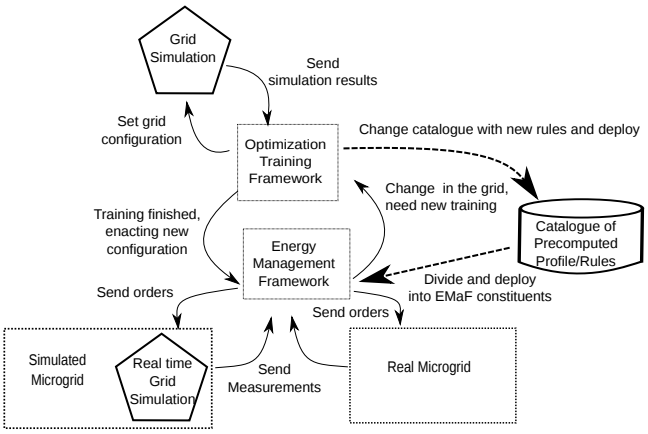

Figure 1: ABEMS main components

3.1 Optimization Training Framework (OpTF)

As commented in previous section, EMaF agents must be capable to react properly and perform the correct actions to face any incoming situation. In other words, they have to find the best *configuration* for the self-controlled elements in a microgrid. Sometimes, in case of simplistic microgrids, best decisions are easy to find by just carrying straightforward calculations. Unfortunately the complexity of such problem increases when it tackles a higher number of elements and constraints. For this reason, the mentioned problem demands more complex optimization processes. OpTF framework takes care of this by means of metaheuristics where NSGA-II is chosen to create the catalogs. Note that the goal of this work is not finding the best optimizer but using it as an optimization tool so, if convenient, it could be easily replaced by any other algorithm.

A *catalog* of an agent is the set of $\{p, (i_1..., i_n)\}$ where $\{i_1..., i_n\}$ are the instructions that it must carry out when the activation conditions of profile p are fulfilled. Those instructions represents the orders to be executed by the agent, where each i_i represents a command of configuration for a specific device assigned to the agent. They contain relevant grid configurations (elements to be turned on/off or PQ values to be set) selected after running thousands of simulations over different scenarios, including faulty ones (power line disconnections for various reasons, mainly). Each simulation represents hours of microgrid operation, and it is evaluated to determine how well they satisfy different objectives and operation preferences. The most promising configurations are stored in the *catalog* and then divided and deployed in each **EMaF** agent.

NSGA-II works with populations of individuals that represent particular microgrid configurations. Each one will contain a list of numbers which are processed to get the values that will be set in the controlled elements. It is coded as $x = c_1, ..., c_m, ..., c_n$ where $c_m \in [0, 1]$ is a command for DER element m. For on/off controlled DERs: OFF mode is $c_m = 0$ and ON is $c_m = 1$. In full control DERs, c_m codifies the active power (W). This way, it is possible to conceive the DER control that is either set to ON or OFF, or to determine its power throughput. Note that non self-controlled elements will not be represented in the individual. The fitness function is defined in terms of the results of discrete event simulations made with the grid configuration and GRIDLAB-D. However, not all individuals are valid ones. It means that, as a consequence of working with real microgrids, they must meet the following constraints:

(1) Each controlled element must respect its own operational constraints.
(2) In full control elements, the power value must be between the lower and upper limits: $c_i \in [min_i, max_i]$.
(3) Lines, elements or transformers cannot be overloaded.
(4) Lines voltage must respect the voltage limits interval.
(5) Do not switch off any generator if it is not producing ($c_i = 0$ W) or if it is not strictly necessary.

The optimization process deals with these constraints by two different ways. Constraints 1 to 4 are already managed within reproduction operators in order create only solutions that meet them. However, since constraint 5 is more complex, a specific reparation mechanism is applied within the optimization process.

The result of the training is a set of pareto fronts of solutions (i.e. instructions for each controllable DER) confronting the different conflicting goals. Since just one solution is needed, we implemented three strategies for selecting an individual from the set: (i) select the solution with the best value of a specific objective; (ii) select the solution with less losses if only if the demand is below a specified threshold; and (iii) use an instruction selection by giving preferences to each objective. This last method, presented in algorithm 1, is the most complex one. Once the solution is selected, it needs to be decoded. **OpTF** will extract the values from the individual converting them into actions to be applied by **EMaF** agents to its corresponding DER assignments. of the corresponding agent that

Algorithm 1 Preferences Selection Method

INPUTs: $\overline{I} = I_0, \ldots, I_n$ the priority level (in %) for each objective
INPUTs: $\overline{O} = o_0, \ldots, o_n$ the objectives under study
Select priority objective po
Get min_i and max_i values for each objective o_i
Apply Bubble Sort algorithm to sort the Pareto front from min to max
$selected = 0$, $proximity = 0$, $previous = \infty$
for every $s \in [1..n_f]$ being n_f the number of solutions in the pareto front **do**
  $assesment = 0$
  for every $i \in [1..n]$ being n the cardinality of $\overline{O}$, $n \geq n_f$ **do**
    $proximity = oi^s / min_{po}$
    $assesment = proximity * I_i$
    if $assesment < previous$ **then**
      $selected = s$
    end if
    $previous = assesment$
  end for
end for
OUTPUT $selected$ solution

will execute them. Batteries are not included as a DER, because they increase too much the search space.

This procedure comes with a time complexity of $O(n * m)$, where n is the number of different profiles and m the number of controlled DERs that are being considered. This cost can be assumed, since it is done only once after the training. During runtime, choosing the right *profile* is at most polynomial time $O(n)$, where n is the number of different profiles.

3.2 Energy Management Framework (EMaF)

The **EMaF** is a Multi-Agent System that enacts the catalog of actions obtained from the **OpTF**. Agents are not cognitive or BDI. Their intelligence is computed offline by the OpTF. Since different control strategies can be followed, every agent has a *catalog* per each of them. The strategy is specified as a parameter so that agents know which *catalog* activate.

There is no prior organizational structure in the **EMaF**, just an aggregation of agents responsible of sets of DER devices. A logical assignment is to identify one software agent per transformation center (TC), as in figure 2, because it is a natural division between lower voltage section (downstream the TC) and medium/high voltage section (upstream the TC). Grid metering is usually transmitted every a specific amount of time (5 minutes in our experiments) through Data Concentrators (DC). Since DCs establish links with Smart Meters (SM) located downstream them[16], real time information is not assumed and that eventual noise or micro-variations would be ignored.

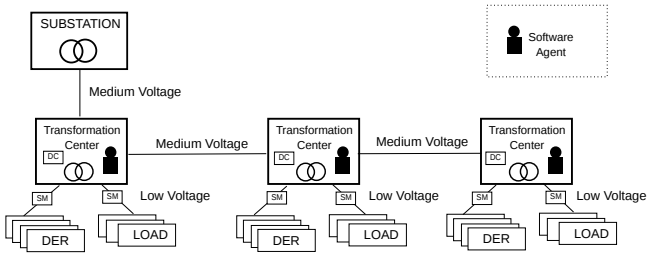

Figure 2: EMaF deployment

Given this arrangement, it makes sense to allocate DCs in the transformation centers since SMs are downstream those centers. As DCs have better hardware than SMs, agents are expected to perform better if hosted by them.

Agents situated in DCs, then, select an adequate course of actions according to their catalogs and enact the right configuration. When a set of actions is selected, each agent still has to consider timeouts and assume orders that may have not the exact effect as trained.

All agents have the same level of responsibility, i.e., there are no master-slave relationships or hierarchies. Besides, an action is not conditioned by instructions or calls made by other agents. This is a pure decentralized system where coordination emerges thanks to the catalog produced by the **OpTF** during the training and the fact that all agents share the same grid. This way, effects of orders can be almost immediately seen and used for coordination, as in [14].

Agents are coded with plain Java in this paper, but they can be easily coded into C. They do not rely on any agent platform. However their construction and behavior are based on agent principles, in particular, the agent lifecycle of perception-decision-action. Perception is implicit since it is related to collecting data from the grid with some periodicity, as explained before. The following two components address the rest:

(1) The *Order selector* analyzes the information received from the grid, determines if any action is needed, and identifies a matching profile from the training catalog. Each agent has its own piece of the original **Catalog** (see figure 1) with actions that only affects to its controlled elements. The cost of choosing an entry from the catalog is $O(n)$, where n is the number of different profiles. If the profile is not found, the agent will trigger a default action, and so will do the rest of agents.
(2) Orders are transmitted directly to the *Communication Service* component of its DC to be submitted by TCP/IP or Modbus to the correspondent DER.

In order to reduce the training complexity, batteries are handled separately. Batteries either increase or reduce the load capacity of the microgrid, and the training does account for different configurations depending on the load. Hence, reaction of the agents according to the catalog is consistent with a non-catalog based operation of the batteries. The batteries handling strategies follow:

- If the tariff period is the lowest cost one, then the priority is charging batteries. Batteries can behave either as a load or as a generator, but they have limitations and the most important one is the charge cycle. First of all, batteries must be discharged. If batteries are charging, then their consumption is considered as an additional load. The resulting increased demand is then used as query for the instructions catalog.
- If the energy demand is lower than the production capability of the microgrid, then battery charging may be considered.
- If demand is greater than generation, and batteries are charged, then batteries start discharging. Batteries are recharged later on according to the previous instructions.

4 EXPERIMENTAL FRAMEWORK

In this section, the ABEMS performance with a use case is presented. ABEMS interacts with a real time smart grid Simulator that will reproduce specific weather and load *situations*.

4.1 The Tested Power Grid

The power grid used in the experimentation is based on a real one located at the CEDER-CIEMAT research center (Soria, Spain) [1]. For experimental purposes, we selected a group of available elements in CEDER to compose the microgrid showed in figure 3. There are 7 TCs where each of them has a transformer represented by two partially overlapped circles. These transformers are used to convert the electricity from 15000 V to 400 V and vice versa. The power grid also counts with 17 consumers represented with green boxes and 8 suppliers (4 solar panels and 4 windturbines). Each segment is labeled as LV (lower voltage), MV (medium voltage), or HV (high voltage). The electric company supplies electricity at 45 kV and, by a transformer of 1000 kVA placed at the entry of the center, it is transformed in to 15 kV. Internal transformation centers convert it again to lower voltage parameters (400V). Each transformation center is assumed to contain a data concentrator (DC), which hosts a software agent. Figure 3 shows which DERs can be controlled and which ones cannot. All solar panels in the grid can be turned on or off. Some wind generators cannot be controlled, one can be turned on/off, and the power of the last one provides can be full controlled (PQ control). Buildings are modeled as loads with a maximum expected consumption power.

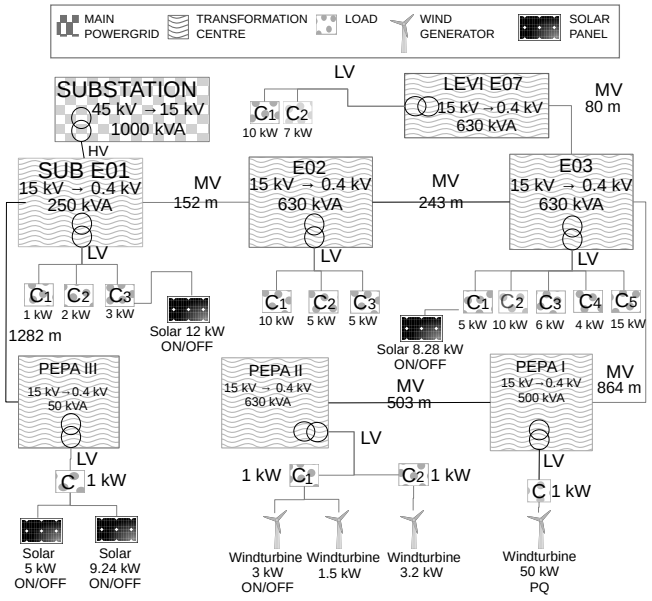

Figure 3: CEDER power prid

The external electricity is provided to the power grid by the spanish electrical company "IBERDROLA" under the tariff 3.1*A*. This tariff considers two penalties: for exceeding contracted power and for exporting energy to the main power grid (producing more energy than it is consumed).

4.2 Experimental Setup

The ABEMS will work with scenarios without forecasting capabilities. It must be able to react to different situations when the mentioned conditions vary. As it is previously commented, **OpTF** identifies these possible situations in advance and computes optimized solutions for them. Then, these optimizations are transformed into instructions that are stored in **EMaF** agent *catalogs*. Such *catalogs* are used by **EMaF** agents in the energy management framework.

In the experimental part of this work, **OpTF** faces two objectives to minimize at the same time. (i) *Minimize the positive demand or power excess at the substation*: according to the contracted rate with the electrical company, the more imported energy there is, the higher the bill is. On the other hand, the exportation of energy is also penalized. And (ii) *Minimize the overall transformation and distribution losses*: it is desirable not to waste energy that can be used. This favors activating DERs that are closer in distance to target consumers. This way, it reduces power line transport and transformation center loses.

In addition to this, CEDER test grid topology imposes the following constraints that must be satisfied: (1) avoid, if possible, any over-voltage situation (over-voltage situations are signalized by GRIDLAB-D); (2) if a supplier is not producing any energy, do not switch it off; (3) maximum power generation/consumption per element cannot be exceeded.

OpTF uses the NSGA-II[4] as optimizer. This algorithm is set up with a population and archive size of 50 individuals. It performs 5000 evaluations and utilizes the Binary Tournament selection method. It also uses a SBX crossover and a Polynomial mutation with probabilities of 0.9 and 0.1 respectively. Constraints are satisfied by repairing the individuals after the operators. Furthermore, the optimization method has been adapted to run several optimizations in parallel. Remind that the full training framework is executed off-line and calls the optimization for any single pre-computed profile.

For testing purposes, another pool of 100 feasible scenarios was generated and used by EMaF and situations of 24h duration has been evaluated (each of them with its own load and weather variations). Remind that a "feasible scenario" is a set of parameters which represents generation/consumption power values that devices of the microgrid may present in the real world. Feasibility means, for instance, considering that solar suppliers do not generate power at night, or making values to always respect the power limits.

The agent deployment corresponds to an agent for each transformation centre having controllable elements. In the figure 3, agents are assigned to each transformation centre, but only five of them will have controllable DER. These will be agents allocated into PEPA III, PEPA II, PEPA I, SUB E01, and E03. As a result, agents in the remaining transformation centres will not have assigned orders.

4.3 Experimental Results

The ABEMS setup and running is performed into three stages: **train**, **solution selection** and **test**. The last part is where the decentralized execution of the most appropriate configuration is performed. It evaluates on the one hand the stability of the substation demand (no oscillations due to the distribution of the order deliver), and the effectiveness of the trained strategy (the overall behavior of

the decentralized order dispatch matches the expected behavior from the evolutionary computing algorithm). The first evaluation is achieved by inspecting scenarios and checking the substation demand is stable. The second evaluation is attained by running a pool of scenarios and comparing the resulting imported and exported energy, and losses, among the different trained strategies. The result should match the intuitive result of the original strategy despite being enacted in a decentralized way.

Simulations of the micro-grid are performed using a Java wrapper for the GRIDLAB-D [2] tool for discovering the effects of the instructions sent to the different DERs of the microgrid. This explains why the variations within the scenarios are so stable and with little noise. This is also consistent with the approach in the project where data is not collected in real time, but with some predetermined periodicity, by data concentrators hosted in the transformation centres.

Within the **training** part, we first identify the possible consumption/generation power that each supplier/consumer of the grid can have, and then we establish ranges for those elements with many possible values. The result is a set of 29750 possible profiles from which about 6000 were simple enough to be easily solved by performing little calculations instead of calling NSGA-II. Then, to deal with the rest, we run NSGA-II that searched for the solutions with the best compromise between objectives. It starts with a random population of individuals which evolves along the generations to form the final solution set and its correspondent Pareto front. Such training was run in parallel by using 8 independent processes. Therefore we can define the computational complexity of the training as $N_s * O(M \cdot N^2)$; where $N_s = 23750/8 = 2969$ is the number of trained profiles for the experimental microgrid, $M = 2$ the number of the objectives (energy demand -positive or negative- and losses), and $N = 10$ the population size. The training is repeated several times, one per profile. Under these conditions, generating *catalogs* for CEDER scenarios took one day to compute.

Once the final fronts are generated, the **solution selection** stage is activated to extract the most suitable individual from each of them. Selected solutions are then decomposed into orders which are then distributed into the agent catalogs. The best configuration can be chosen according to three selection preferences (as described in sec. 3.1): giving preference to reduce the energy demanded from the substation, named *MinDemand* (one extreme of the front); the one that aims to minimize transport losses, named *MinLosses* (opposite extreme); or the one that moves across the pareto curve searching a point with good trade-off and that fits user desires, named *Pondered* and for which selection alg. 1 is used.

Finally, the **testing** stage runs the ABEMS within the simulator so that **OpTF** is tested under different scenarios. Scenarios are run using dedicated code, but a graphical tool is used to monitor their execution and interact with them if needed (e.g. switching off some parts of the grid). The tool also shows which orders are being given to each DER element and what generation capability is being used.

In order to study that the decentralized configuration enactment still follows the trained strategies, the three mentioned strategies (*Pondered*, *MinDemand* and *MinLosses*) were tested and their performance compared with two standard grid configurations (not optimized): all DERs are ON and producing at maximum capacity (*ON*); and DERs disconnected but consumers are working (*NoGen*). Results are shown in table 1. Besides these scenarios, the experimentation includes a sample non-trained situation in figure 4 (above). These evidences suggest that the decentralized execution performs well in this domain, which is the goal of this paper and a necessary previous step prior to a comparisson with other methods.

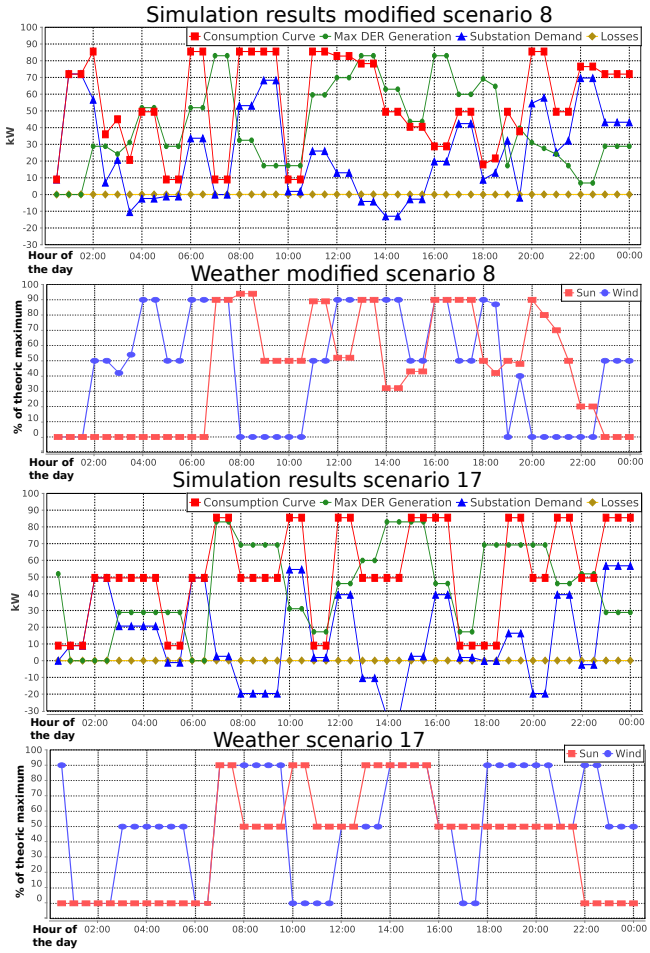

Figure 4: Above, results for a modified scenario 8 and Pondered strategy. Below, for scenario 17 and Pondered strategy.

Figure 4 shows two executions of the system attending to the simulation results and the weather conditions along 24 hours. The weather is depicted as a percentage of the maximum amount of sun or wind along the day. The grid behavior is presented as four variables: the demanded energy from consumers (consumption curve), the theoretical maximum grid generation capability given the current weather conditions (max DER generation), the energy dissipated at this moment (losses), and the energy that is finally demanded to the substation (substation demand). Figure 4 (above) introduces a randomly modified version of one of the scenarios from Table 1 to show that it works with an untrained situation. Figure 4 (below) introduces an unaltered scenario and situation of the same grid. The latter shows a stable behavior in curves with almost no variations in the demand and weather. This scenario is

useful to detect if orders given to one agent add alterations that trigger a cyclic behavior, such as an agent shutting down a DER and then turning it on repeteadly.

General execution is partially satisfactory because the excess of produced energy (negative substation demand) is greatly reduced with agents. They adjust and enable/disable specific DER when the maximum DER generation capability variable exceeds the demand. The line losses seem close to zero because losses are in the range of watts. Scenario 8 in Figure 4 (above) showed us that, despite the small variations made to the original scenario, agents are not issuing endless instructions in the logs. Not even in scenarios with stable conditions, such as weather in scenario 17.

Figure 5 shows a particular scenario and situation to compare the optimization strategy following algorithm 1 (line with circle markers) against the non-optimized configuration with all suppliers at maximum capacity *ON* (line with triangle markers) and with only consumers and no DER *NoGen* (line with diamond markers). The graphic clearly shows how the smart simulation decreases, when possible, the excess of imported (positive y axis) and exported energy (negative y axis). Note that a zero demand in the substation means that the microgrid is self-supplying (best situation). The energy lost along the process is minimal compared with the energy demanded, but it may not be the case if the grid configuration was less efficient. Again, the behavior of the decentralized enactment of the pondered strategy seems to preserve the intended behavior, limiting the amount of exported energy (substation demand is always positive) while reducing the substation demand (it is always below the "only consumption" curve).

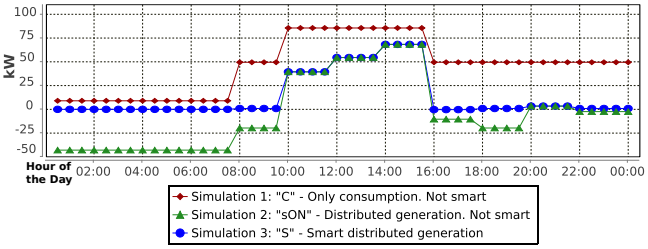

Figure 5: Substation demand in a scenario (*Pondered* strategy)

Table 1 presents the substation demand split into imported and exported energy. Obviously, the *NoGen* will not have any exported energy since this configuration do not use any DERs. However, *ON* does have some imported energy due to the fact that in several occasions the maximum feasible power generation was less than all the power demand. Also accumulated energy losses dissipated in lines and transformation across all scenarios are displayed in Table 1. *ON catalogs* are the ones that provoke more losses to the grid, but the energy dissipated by *NoGen* is similar to the optimisations. We also observe that the grid (due to its topology) looses more energy supplying than consuming. However, the energy dissipated in the grid is much smaller than the power demand and so the differences are less remarkable. Since both objectives are contradictory we want to find the configurations with the minimum and best compromise between losses and power demand, and so the solutions must be analysed regarding both objectives at the same time, and not separately. Having few losses is not useful if it costs a big excess of demanded energy. For the previous reason, and since the other goal is also to decrease as much as possible the positive or negative demand to the external substation, we compared the addition of both excess of energy on non-optimized profiles (*ON* and *NoGen*) with the optimizations evaluated in the 100 scenarios. IE+EE results from Table 1 show too that the optimized versions notably decrease the demand at the same time they minimize the losses. Again, the intuitive expected results from the original strategies is preserved despite the decentralization, and the results are objectively better than the *ON*/*NoGen* strategies.

Table 1: Aggregating results from 100 possible scenarios. Imported Energy (IE), Exported Energy (EE) and Losses

(kWh)	IE	EE	Losses	IE + EE
No Gen	116756.63	0.00	103.23	116756.63
ON	49345.18	33107.594	139.08	82452.77
Pondered	50711.17	355.99	105.10	51067.15
Min Demand	50525.70	355.99	108.17	**50881.69**
Min Losses	62838.96	0.00	**101.93**	62838.96

If the engineers were more interested on having an efficient grid with lower demand, they would select the *MinDemand* catalogs to manage the grid. But if they preferred to decrease the energy demand with reasonable losses, they would choose *MinLosses*. However, *Pondered catalogs* represents a good compromise between objectives: both energy losses and demand are slightly higher than the extremes of the front but differences are not remarkable. Due to the opposite nature between objectives, the smaller the losses are, the higher the substation demand is, and vice versa. Besides, the gain obtained by taking into account transport losses is not high. For instance, *Pondered* strategy increases the imported energy by 186kWh compared with *MinDemand*, whereas less than 3kWh were saved in loses. Note also that conditions in the tested scenarios are more stable than what real devices would find, e.g. there is no noise or micro-variations.

In terms of energy saving, tables show the outperformance of ABEMS strategies when facing all kind of scenarios (including failure ones). This matches the expectations we had of these strategies and proves their decentralized execution is not altering this.

5 CONCLUSIONS

The paper introduced an agent based framework (ABEMS) able to control, by pursuing conflicting goals, Distributed Energy Resources in a simulated microgrid. It works in a decentralized way and at a low computational cost. The off-line method would not need to be run on every change if the training in the OpTF were thorough enough. It means that a good training will check many possible grid variations and this will make deployed agents more capable.

The behavior of the Multi-Agent System is satisfactory in system control terms since it does not involve apparent oscillations in the microgrid. This is a proof of concept. Now more extensive tests can be made to check the performance over other control alternatives.

Acknowledgments. To the project MIRED-CON IPT-2012-0611-120000 and TIN2017-88327-R funded by the Ministry of Economy and Competitiveness from Spain.

REFERENCES

[1] CEDER. 2012. Center for the Development of Renewable Energy Sources. http://www.ciemat.es/CEDERportal/portal.do. (2012). (accessed on 17 November 2014).

[2] D.P. Chassin, K. Schneider, and C. Gerkensmeyer. 2008. GridLAB-D: An open-source power systems modeling and simulation environment. In *Transmission and Distribution Conference and Exposition, 2008. IEEE/PES*. 1–5. https://doi.org/10.1109/TDC.2008.4517260

[3] Hanane Dagdougui, Riccardo Minciardi, Ahmed Ouammi, Michela Robba, and Roberto Sacile. 2012. Modeling and optimization of a hybrid system for the energy supply of a Green building. *Energy Conversion and Management* 64, 0 (2012), 351 – 363. https://doi.org/10.1016/j.enconman.2012.05.017 {IREC} 2011, The International Renewable Energy Congress.

[4] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. 2002. A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation* 6, 2 (2002), 182–197.

[5] Jose Evora, Jose Juan Hernandez, and Mario Hernandez. 2015. A MOPSO method for direct load control in smart grid. *Expert Systems with Applications* 42, 21 (2015), 7456–7465.

[6] H. Farhangi. 2010. The path of the smart grid. *Power and Energy Magazine, IEEE* 8, 1 (2010), 18–28. https://doi.org/10.1109/MPE.2009.934876

[7] P.S. Georgilakis and N.D. Hatziargyriou. 2013. Optimal Distributed Generation Placement in Power Distribution Networks: Models, Methods, and Future Research. *Power Systems, IEEE Transactions on* 28, 3 (Aug 2013), 3420–3428. https://doi.org/10.1109/TPWRS.2012.2237043

[8] D. Goldberg. 1989. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Longman Publishing Co., Inc. Boston, MA, USA.

[9] Jorge Gomez-Sanz, Sandra Garcia-Rodriguez, Nuria Cuartero-Soler, and Luis Hernandez-Callejo. 2014. Reviewing microgrids from a multi-agent systems perspective. *Energies* 7, 5 (2014), 3355–3382.

[10] Jorge J. Gómez-Sanz, Nuria Cuartero-Soler, and Sandra Garcia-Rodriguez. 2015. A Testbed for Agent Oriented Smart Grid Implementation. In *Engineering Multi-Agent Systems - Third International Workshop, EMAS 2015, Istanbul, Turkey, May 5, 2015, Revised, Selected, and Invited Papers*. 92–108.

[11] Jorge J Gomez-Sanz, Sandra Garcia-Rodriguez, and Nuria Cuartero-Soler. 2018. Modelling agent oriented solutions for the smart grid. *International Journal of Agent-Oriented Software Engineering* 6, 2 (2018), 133–155.

[12] Grefenstette, Ramsey, and Connie Loggia Ramsey. 1992. An Approach to Anytime Learning. In *Proceedings of the Ninth International Conference on Machine Learning*. Morgan Kaufmann, 189–195.

[13] Yunjie Gu, Xin Xiang, Wuhua Li, and Xiangning He. 2014. Mode-Adaptive Decentralized Control for Renewable DC Microgrid With Enhanced Reliability and Flexibility. *Power Electronics, IEEE Transactions on* 29, 9 (Sept 2014), 5072–5080. https://doi.org/10.1109/TPEL.2013.2294204

[14] J.M. Guerrero, J. Matas, Luis Garcia de Vicuna, M. Castilla, and J. Miret. 2007. Decentralized Control for Parallel Operation of Distributed Generation Inverters Using Resistive Output Impedance. *Industrial Electronics, IEEE Transactions on* 54, 2 (April 2007), 994–1004. https://doi.org/10.1109/TIE.2007.892621

[15] Josep M Guerrero, Poh Chiang Loh, Mukul Chandorkar, and Tzung-Lin Lee. 2013. Advanced Control Architectures for Intelligent MicroGrids, Part I: Decentralized and Hierarchical ControlAdvanced Control Architectures for Intelligent Microgrids. *IEEE Transactions on Industrial Electronics* 60, 4 (2013), 1254–1262.

[16] Vehbi C Gungor, Dilan Sahin, Taskin Kocak, Salih Ergut, Concettina Buccella, Carlo Cecati, and Gerhard P Hancke. 2011. Smart grid technologies: communication technologies and standards. *Industrial informatics, IEEE transactions on* 7, 4 (2011), 529–539.

[17] Michael N Huhns and Munindar P Singh. 1999. Multiagent treatment of agenthood. *Applied Artificial Intelligence* 13, 1-2 (1999), 3–10.

[18] F. Katiraei, R. Iravani, N. Hatziargyriou, and A. Dimeas. 2008. Microgrids management. *Power and Energy Magazine, IEEE* 6, 3 (May 2008), 54–65. https://doi.org/10.1109/MPE.2008.918702

[19] Thillainathan Logenthiran, Dipti Srinivasan, Ashwin M. Khambadkone, and Htay Nwe Aung. 2012. Multiagent System for Real-Time Operation of a Microgrid in Real-Time Digital Simulator. *IEEE Trans. Smart Grid* 3, 2 (2012), 925–933.

[20] Libardo López, Ricardo Alberto Hincapié, and Ramón Alfonso Gallego. 2011. PLANEAMIENTO MULTIOBJETIVO DE SISTEMAS DE DISTRIBUCIÓN USANDO UN ALGORITMO EVOLUTIVO NSGA-II. *Revista EIA* 15 (2011).

[21] F.A. Mohamed and H.N. Koivo. 2007. Online Management of MicroGrid with Battery Storage Using Multiobjective Optimization. In *Power Engineering, Energy and Electrical Drives, 2007. POWERENG 2007. International Conference on*. 231–236. https://doi.org/10.1109/POWERENG.2007.4380118

[22] F.A. Mohamed and H.N. Koivo. 2007. System Modelling and Online Optimal Management of MicroGrid Using Multiobjective Optimization. In *Clean Electrical Power, 2007. ICCEP '07. International Conference on*. 148–153. https://doi.org/10.1109/ICCEP.2007.384202

[23] M. Petersen, J. Bendtsen, and J. Stoustrup. 2012. Optimal dispatch strategy for the Agile Virtual Power Plant. In *American Control Conference (ACC), 2012*. 288–294.

[24] P.C. Ramaswamy and G. Deconinck. 2012. Smart grid reconfiguration using simple genetic algorithm and NSGA-II. In *Innovative Smart Grid Technologies (ISGT Europe), 2012 3rd IEEE PES International Conference and Exhibition on*. 1–8. https://doi.org/10.1109/ISGTEurope.2012.6465615

[25] Sarvapali D Ramchurn, Perukrishnen Vytelingum, Alex Rogers, and Nicholas R Jennings. 2012. Putting the'smarts' into the smart grid: a grand challenge for artificial intelligence. *Commun. ACM* 55, 4 (2012), 86–97.

[26] Eleonora Riva Sanseverino, Maria Luisa Di Silvestre, Mariano Giuseppe Ippolito, Alessandra De Paola, and Giuseppe Lo Re. 2011. An execution, monitoring and replanning approach for optimal energy management in microgrids. *Energy* 36, 5 (2011), 3429 – 3436. https://doi.org/10.1016/j.energy.2011.03.047

[27] A. Schafer and A. Moser. 2012. Dispatch optimization and economic evaluation of distributed generation in a virtual power plant. In *Energytech, 2012 IEEE*. 1–6. https://doi.org/10.1109/EnergyTech.2012.6304655

[28] J. Schonbergerschonberger, R. Duke, and S. D. Round. 2006. DC-Bus Signaling: A Distributed Control Strategy for a Hybrid Renewable Nanogrid. *IEEE Transactions on Industrial Electronics* 53, 5 (Oct 2006), 1453–1460. https://doi.org/10.1109/TIE.2006.882012

[29] David Schuff and Robert St Louis. 2001. Centralization vs. decentralization of application software. *Commun. ACM* 44, 6 (2001), 88–94.

[30] Anita Sobe and Wilfried Elmenreich. 2013. Smart Microgrids: Overview and Outlook. *CoRR* abs/1304.3944 (2013).

[31] Bogdan Tomoiaga, Mircea Chindris, Andreas Sumper, Antoni Sudria-Andreu, and Roberto Villafafila-Robles. 2013. Pareto Optimal Reconfiguration of Power Distribution Systems Using a Genetic Algorithm Based on NSGA-II. *Energies* 6, 3 (2013), 1439–1455. https://doi.org/10.3390/en6031439