2013 IEEE 7th International Conference on Self-Adaptive and Self-Organizing Systems

Self-Adapting Coalition Formation among Electric Vehicles in Smart Grids

Gabriel de O. Ramos
PPGC/Instituto de Informática
Univ. Federal do Rio Grande do Sul
15064-Porto Alegre, RS, Brazil
Email: goramos@inf.ufrgs.br

Juan C. Burguillo
Telematics Engineering Department
University of Vigo
36310-Vigo, Spain
Email: J.C.Burguillo@uvigo.es

Ana L. C. Bazzan
PPGC/Instituto de Informática
Univ. Federal do Rio Grande do Sul
15064-Porto Alegre, RS, Brazil
Email: bazzan@inf.ufrgs.br

***Abstract*—In the last years, the need for using multiple energy sources made the concept of smart grids emerge. A smart grid is a fully automated electricity network, which monitors and controls all its elements being able to supply energy in an efficient and reliable way. Within this context, the use of electric vehicles (EVs) and Vehicle-To-Grid (V2G) technologies have been advocated as an efficient way to reduce the intermittent supply associated with renewable energy sources. However, operating on V2G sessions in a cost effective way is not a trivial task for EVs. To address this problem, the formation of coalitions among EVs has been proposed as a mean to improve profitability on V2G sessions. Addressing these scenarios, in this paper we introduce the Self-Adapting Coalition Formation (SACF) method, which is a local and dynamic heuristic-based mechanism for coalition structure generation. In our approach, coalitions are formed observing constraints imposed by the grid to the EVs, which negotiate locally the formation of feasible coalitions among themselves. Based on experiments, we see that SACF is an efficient method, providing good solutions in a simple and low-cost way. SACF is faster than centralized methods and provides solutions with near optimal quality. In dynamic scenarios, SACF also shows very good results, being able to keep the agents gain relatively stable along time, even in quickly changing environments. Its main advantage is that the computational effort is very low, while classical centralized methods are limited to manage no more than a dozen agents in a reasonable amount of time.**

I. Introduction

Despite its importance, electricity grids have evolved very little since they were created. The energy demand, in turn, has grown manifold. In order to meet such a demand, the energy industry has invested mostly on the construction of large power plants. However, such policies resulted in non-redundant grids, which strongly depend on non-renewable, polluting energy sources, whose availability is becoming increasingly scarce. As a consequence, such an infrastructure has become very inefficient and unsafe. Moreover, the demand for reliable, uninterrupted energy supply increases year by year.

In this scenario, the concept of smart grids emerges. According to the U.S. Department of Energy [1], smart grid is a fully automated electricity network, which intensively monitors and controls every element that composes it, being able to supply energy in an efficient and reliable way. One of the main features of a smart grid is the bidirectional flow of energy and communication between its elements. Along these lines, any element can both supply and consume energy so that, e.g., households with own power generation capacity can sell their surplus energy to the grid.

An interesting concept that has emerged in the field of smart grids is the Vehicle-To-Grid (V2G). Through V2G sessions, electric vehicles (EVs) can provide part of the energy available in their batteries to the grid [2]. Such mechanism is important in situations where the grid relies on intermittent renewable energy sources, such as wind and solar. Thus, the energy stored in the EVs' batteries can be used when supply is not able to meet the demand. Additionally, EVs can make a profit by selling their energy on V2G sessions.

Although the V2G mechanism brings several advantages to the grid, participating in V2G sessions in a cost-effective way is not a trivial task for EVs. According to [3], to act in V2G sessions, EVs must commit to provide a given amount of energy. However, when operating alone, EVs are not able to fulfil their commitment due to their insufficient energy capacity and unpredictable availability. In order to address this problem, a particularly interesting approach is the formation of coalitions among EVs, forming virtual power plants (VPPs) [4, 5, 6, 7]. By operating in coalitions, EVs can coordinate in order to provide more accurate predictions about their energy availability. Furthermore, the amount of available energy also increases. Therefore, coalition formation has shown to be an effective way of increasing the profitability of EVs [3, 2].

A coalition can be defined as a group of individuals or agents that decide to cooperate in order to achieve a common goal [8]. According to [9], coalition formation includes three activities: coalition structure generation (CSG), solving the optimization problem of each coalition, and dividing the obtained value among the agents. Among these activities, CSG is the most studied one in the multiagent systems community. Traditional CSG algorithms [9, 10, 11] are concerned not only with grouping agents, but also wish grouping them in order to obtain the greatest possible reward. Such activity, however, has been proven to be NP-complete [9]. Differently, in this paper we focus on dynamic CSG (DCSG hereafter). DCSG is an heuristic approach for CSG, whose focus is not on finding the optimal solution, but on finding a good one (in a reasonable amount of time) in dynamic environments, i.e., environments that change constantly (agents can enter or leave the system at any time). We remark that traditional methods for CSG do not deal with such dynamic scenarios.

The smart grid scenarios described before are characterized by a great flexibility (the environment should adapt quickly

978-0-7695-5129-6/13 $26.00 © 2013 IEEE
DOI 10.1109/SASO.2013.12

to changes in the context), robustness (a failure in one or more EVs should not affect the whole system), decentralized behaviour (due to the inherent characteristics) and self-organization (no external supervision). In this sense, DCSG is a more suitable approach for smart grids than traditional CSG.

In this paper we propose the formation of coalitions among EVs in order to form virtual power plants and supply energy to the grid. As stated in [3, 2], by forming coalitions, EVs can increase their efficiency and, consequently, their profitability. Therefore, we present the SACF (acronym to Self-Adapting Coalition Formation), which is a DCSG approach capable of pruning the space of coalition structures, based on domain information that refers to impossibility of some agents belonging to the same coalition. SACF has been designed to run in every agent and to interact locally in its limited neighbourhood to avoid computation issues faced by classical CSG approaches. Through experiments, SACF has shown to be effective on providing good solutions in a fast way.

The remainder of this paper is organized as follows. In Sect. II we give details about coalition formation. Related work on coalition formation in smart grids is discussed in Sect. III. In Sect. IV we present the modelling of the problem addressed in this work. Section V presents our approach. Then in Sect. VI we present experiments and analysis of the method. Conclusions and future work are presented in Sect. VII.

II. Background on coalition formation

The organization of agents in an efficient way is a major challenge in multiagent systems. According to Horling and Lesser [12], different organizational paradigms can be used to coordinate agents. Among these, the formation of coalitions stands out. A coalition can be defined as a group of agents that decide to cooperate in order to achieve a common goal [8]. Specifically, given a set of agents $A = \{1, 2, ..., a\}$, a subset of it is denominated a coalition C. The partition of the set A of agents into disjoint and exhaustive coalitions is called a coalition structure (CS) [9]. In the literature, coalition formation is commonly studied in the form of characteristic function games (CFGs). In CFGs, the value of a coalition C is given by a characteristic function $v(C)$, and the value of a coalition structure CS is given by $V(CS) = \sum_{C \in CS} v(C)$.

As previously stated, coalition formation includes three activities. Among these, CSG has drawn more attention of the multiagent systems community. In CSG, the aim is to find the optimal coalition structure, i.e., the one with the highest value, which is commonly referred as CS^*. However, there is a scalability issue: the number of possible coalitions is $2^a - 1$ and of coalition structures is asymptotically in the order of $\mathrm{O}(a^a)$ and $\omega(a^{\frac{a}{2}})$ [9]. Moreover, Sandholm et al. [9] has proved that this problem is NP-complete. Many methods have been proposed in order to solve this problem based on heuristics [13], dynamic-programming [11] or in the use of anytime algorithms [10, 14], some of which are discussed next.

Concerning anytime algorithms, Rahwan et al. [10] proposed a branch-and-bound algorithm called IP. The IP algorithm is based on an efficient search space representation, where coalitions are grouped by their sizes (in coalition lists) and coalition structures are grouped by the size of coalitions they contain. This approach has to search, in the worst case, $O(a^a)$ coalition structures. Recently, we have proposed a preprocessing phase for IP [7] (henceforth CPCSG) that uses domain information in order to identify unfeasible coalitions. This improvement prunes the search space before the search is started by IP, but in the worst case the search space remains $O(a^a)$.

With respect to dynamic programming approaches, the main representative is the IDP algorithm [11]. IDP uses basically the same idea as DP [15], which was originally proposed to solve the set partitioning problem. It works by solving each possible coalition, i.e., deciding whether it is better to split it or to keep it as it is. In the worst case, the computational complexity of IDP ($O(3^a)$) is better than IP ($O(a^a)$), but IP, in turn, is able to return near-optimal solutions anytime. However, both IP and IDP do not work on dynamic scenarios.

Other approaches have been proposed to solve the CSG problem from different perspectives. Ueda et al. [16] proposes the solution of the CSG problem using distributed constraint optimization (DCOP) instances. However, the focus of their work is not to find optimal CSG solutions (the best CS), but optimal DCOP solutions (the correct value of the coalitions). Chalkiadakis and Boutilier [17] proposed a Bayesian model-based reinforcement learning framework for repeated coalition formation under uncertainty. Such approach, however, does not address CFGs and is mainly concerned with agents' learning and decision making. Ramchurn et al. [18] have studied coalition formation in task oriented domains. Their approach, however, is suitable for up to ten agents and does not address CFGs.

Thus, traditional CSG methods are not suitable for dynamic scenarios, which are the main motivation of this work.

III. Coalitions in smart grids

The use of coalitions in smart grids has been widely discussed in the multiagent systems community (see [19, 20] for overviews). One of the main interests of the field has been to increase the reliability of renewable energy production.

In [21], Chalkiadakis et al. propose coalition formation among distributed energy resources (DERs) to form VPPs. DERs are renewable energy generators with small-to-medium energy capacity, like wind turbines and solar panels. Taking into account that renewable energy sources are intermittent due to weather conditions, their approach suggests grouping DERs in order to aggregate their production, thus improving their reliability and efficiency. The proposed mechanism incentivizes DERs to provide accurate estimates of their energy production, rewarding good ones. However, this approach has a primary focus on mechanism design rather than on coalition formation, disregarding how far the solution is from the optimal one.

Another approach is the one of Kamboj et al. [22, 5], which proposes the formation of coalitions among EVs in order to operate in the regulation market. The goal of the regulation market is to bring stability to the grid by ensuring that it always meets the demand. The regulation market basically provides power to the grid whenever demand exceeds supply, and stores energy whenever supply exceeds demand. To operate, the market usually depends on large and very expensive batteries and/or slow and polluting generators. Thus, considering that vehicles remain parked 96% of the time [2], the use of EVs' batteries would help to reduce costs and to improve efficiency of the regulation market. However, such approach addresses coalition formation in a naïve fashion, disregarding the solution quality.

In the works of Mihailescu et al. [23, 24], formation of coalitions among producers and consumers is proposed. In their approach, producers who have increased energy availability are probabilistically selected to coordinate coalitions. Such coordinators are responsible for inviting other producers to join their coalitions. Finally, the consumers join the coalitions whose energy profile is more similar to theirs, and also based on their proximity. However, this approach does not address coalition formation as a CFG, nor cares about the solution quality.

The formation of coalitions among producers and consumers is also addressed in [4], specifically, among wind turbines and EVs, also forming VPPs. The goal here is more specific: solve the problem of intermittent power generation of the wind turbines through the use of EVs' batteries, in order to increase the reliability of this kind of energy. However, aspects concerning the optimization of the coalition structure are neither taken into account.

Summarizing, existing works have primarily focused on applications of smart grids than on coalition formation itself. In this way, we can say that our approach lies between traditional CSG and ad hoc methods.

IV. Problem modelling

The scenario presented in our work consists of a smart grid where EV-agents sell their surplus energy in V2G sessions. The aim of the EV-agents is to get the highest profit possible. However, as previously discussed in Sect I, EVs are unable to operate alone in a cost-effective way. Thus, forming coalitions among EVs represents a suitable approach to solve this issue. In this approach, the grid incentivizes the formation of coalitions among the EVs through a monetary value, which is proportional to the coalition size, up to certain limits. We assume that the grid is always willing to buy the energy offered by EVs. Specifically, whenever an EV has energy to be sold, the grid will buy it. Thus, the aim of this work is to increase the agents' profitability through the formation of coalitions among themselves.

A further assumption must be made. As discussed in [7], EVs should supply energy only to consumers who are in the same region as them (or close enough). Such a constraint exists because power lines have a limited energy flow capacity. Considering that multiple power lines may be used in order to supply energy to a single consumer, travelling long distances may impose a huge burden on the distribution network, which may be overloaded. Therefore, the distance among the EVs is a constraint that must be taken into account while the coalitions are being formed. Specifically, EVs must form coalitions only with EVs that are close enough. A coalition that fits into such criteria is said feasible. More formally, if a coalition is feasible, then Eq. (1) must hold, where C is a coalition, i and j are agents of coalition C, d^i_j is the distance between agents i and j, and α is the maximum distance that can exist among the agents of a given coalition. The maximum distance α is characteristic of each grid, defined by the capacity of its power lines.

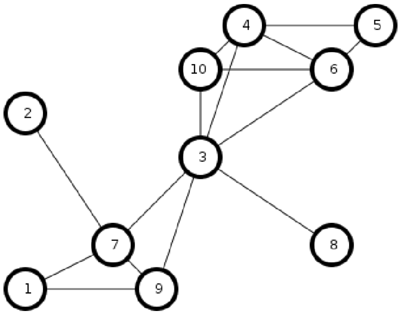


Fig. 1. Graph representation of a randomly generated scenario for ten agents, where nodes represent agents and links represent a neighbourhood relation between two agents

Definition 1 (feasible coalition) A feasible coalition is one for which Eq. (1) holds.

$$\forall i \in C, \forall j \in C \setminus i, d^i_j \leq \alpha \tag{1}$$

Another important definition is the one of neighbours. Basically, two agents are neighbours if the distance between them is lower than α.

Definition 2 (neighbours) The neighbours of a given agent $i \in A$ are all agents $j \in A \setminus i$ for whom d^i_j is smaller than α.

The problem can be represented by a graph where the agents are expressed by nodes and the neighbourhood relation among them is represented by links. An example is presented in Fig. 1. As can be seen, the neighbours of agent 1 are the agents 7 and 9. In this case, $\{1, 7, 9\}$ is a feasible coalition.

Based on the problem proposed in this work, our focus is on EVs that are willing to participate on V2G sessions, only. The proposed approach works as follows. Whenever an EV is plugged into the smart grid, it automatically signs in a peer-to-peer (P2P) network maintained by the grid, whose design and characteristics are out of the scope of this paper. The P2P network is also used for communication purposes, i.e., message exchange. Thus, the P2P network can be seen as a communication layer, through which the agents can share information and even communicate among themselves. Through this P2P network, EVs share personal information with their neighbours (such as location, current coalition and so on). In this way, agents can look for feasible coalitions in their neighbourhood, and ask for permission to join the best one, aiming to increase its profitability. Details about the coalition formation process and its dynamics are described in Sect. V.

Our approach can be formalised as CFGs where the value of a given coalition represents how much incentive the grid is willing to pay for it. The incentive provided by the grid to a given coalition is a function of the coalition's amount of energy (the more, the better) and its size. Specifically, the value of a coalition represents how much the grid is willing to pay beyond the normal price of energy. For example, if the normal price were \$0.50 per energy unit, and for a given coalition

C with five agents the grid would pay \$0.55 per energy unit, then the value $v(C)$ would equal 0.05. In the same way, if the coalition had only one agent and the grid pays \$0.50 per energy unit, then $v(C)$ would be zero.

In this work, we assume that all EVs have the same energy transfer rate, i.e., all EVs that are participating on V2G sessions transfer the same amount of energy per time-step. In this respect, we can simplify our model in a way that the payoff obtained by a given coalition can be equally divided among its members, on every time-step[1]. Although somewhat naïve, this mechanism seems to be a fair scheme. Also important, legal aspects of the energy supply (the contracts) are managed by the grid operators. However, such features were left for future work, since they influence only the coalitions' values, not the CF process itself.

The instantaneous payoff of a coalition, i.e., the payoff received in a single time-step, can be obtained from Eq. (2), where E_C is the amount of energy available in coalition C. By multiplying the amount of energy by the value of the coalition, the total amount that the grid is willing to pay to the coalition is obtained. However, as the duration of a coalition is uncertain, the payoff is calculated for each time-step. As the energy unities are given in kWh, the total amount of energy is divided by 60. The total amount of energy of a given coalition C is given by Eq. (3), where e_i is the amount of energy that agent i has available. Finally, based on Eqs. (2) and (3), the instantaneous payoff of a single agent $i \in C$ is calculated through Eq. (4).

$$P_C = \frac{E_C * v(C)}{60} \tag{2}$$

$$E_C = \sum_{i \in C} e_i \tag{3}$$

$$P_i = \frac{P_C}{|C|} \tag{4}$$

V. Approach

Following the modelling discussed in the previous section, we present SACF (Self-Adapting Coalition Formation), which is a dynamic heuristic-based method for constrained DCSG. SACF was designed to run distributed among the agents, i.e., every agent in the environment runs an instance of SACF.

The SACF method consists of several procedures, which are performed by each agent. In this section we present each of these procedures.

A. Main procedure

The main procedure performed by each agent is presented in Alg. 1. The first line refers to the agents' initialization, where some variables are initialized according to Table I. The rest of the code refers to the negotiation processes for coalition formation.

Negotiation processes are started by agents who are not in coalitions, which are called *requesters*[2]. Requester agents can ask their neighbours (i) for permission to join their existing coalitions or (ii) to form a new coalition (if the neighbour is not in a coalition). In the former case, the requester wants to join a coalition that already exists. In the latter case, both the requester and its neighbour are not in coalitions, so a new coalition is going to be created. The agent who has the lowest ID in its coalition is selected as its *leader*.

TABLE I. Definitions of the procedures' variables

Variable	Description	Initial value
i	id of the agent who is running the procedure	(agent id)
N_i	list of neighbours of agent i	$\{j \in A \setminus i : d^i_j \leq \alpha\}$
C_i	agent i's coalition	$\{i\}$ (singleton)
C_i^{bst}	best coalition found by agent i in phase 1	$\emptyset$
C_i^{ack}	coalition accepted by agent i in phase 2	$\emptyset$
M_i^{rqs}	list of "request messages" received by agent i	$\emptyset$
M_i^{rpl}	list of "reply messages" received by agent i	$\emptyset$
M_i^{cnf}	list of "confirmation messages" received by agent i	$\emptyset$
$wait\_ph4?$	whether the agent is waiting for the coalition consolidation or not	$false$

Algorithm 1: Main procedure performed by the agents

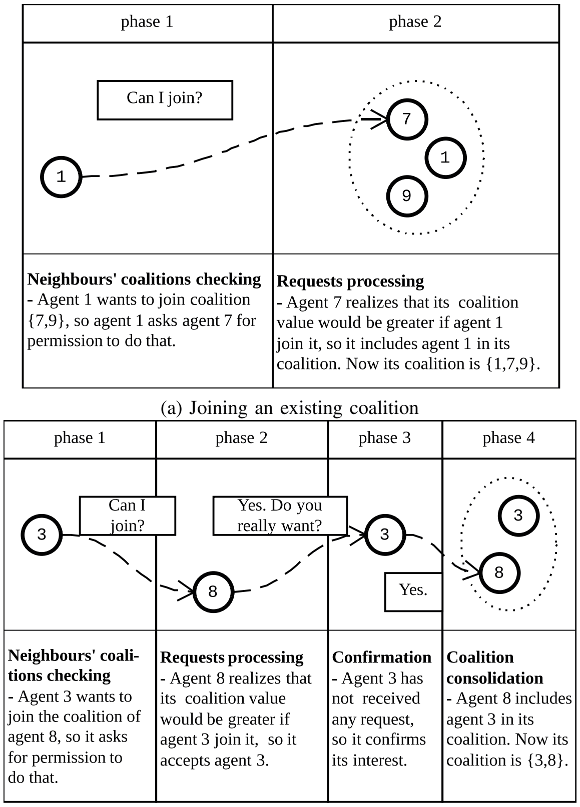
```
1  initialize variables according to Table I;
2  while simulation is running do
       // Neighbours' coalitions checking phase
3      if $C_i = \{i\}$ and $C_i^{bst} = \emptyset$ and $C_i^{ack} = \emptyset$ and
4      not $wait\_ph4?$ then
5          check neighbours' coalitions (Alg. 2);
6      end
       // Requests processing phase
7      if $M_i^{rqs} \neq \emptyset$ and $C_i^{bst} = \emptyset$ and not $wait\_ph4?$ then
8          process requests (Alg. 3);
9      end
       // Confirmation phase
10     if $M_i^{rpl} \neq \emptyset$ and $M_i^{rpl}.answer =$ "Yes" and
       $C_i^{bst} \neq \emptyset$ then
11         confirmation (Alg. 4);
12     end
       // Coalition consolidation phase
13     if $M_i^{cnf} \neq \emptyset$ and $C_i^{ack} \neq \emptyset$ and $wait\_ph4?$ then
14         process confirmations (Alg. 5);
15     end
16 end
```

The working flow of the agents, which characterizes the negotiation process, is basically divided into four phases:

- Neighbours' coalitions checking (lines 3 to 6): singleton agents (requesters) check the existence of coalitions in their neighbourhoods. The agent sends a join request to the leader of the best coalition found, i.e., the one with highest value. At this point, if there is no coalition in its neighbourhood, the agent can send

[1] In this approach, one time-step corresponds to one minute.

[2] It is important to note that, in the beginning, all agents act as requesters.



Neighbours' coalitions checking
- Agent 1 wants to join coalition {7,9}, so agent 1 asks agent 7 for permission to do that.

Requests processing
- Agent 7 realizes that its coalition value would be greater if agent 1 join it, so it includes agent 1 in its coalition. Now its coalition is {1,7,9}.

(a) Joining an existing coalition

Neighbours' coalitions checking
- Agent 3 wants to join the coalition of agent 8, so it asks for permission to do that.

Requests processing
- Agent 8 realizes that its coalition value would be greater if agent 3 join it, so it accepts agent 3.

Confirmation
- Agent 3 has not received any request, so it confirms its interest.

Coalition consolidation
- Agent 8 includes agent 3 in its coalition. Now its coalition is {3,8}.

(b) Creating a new coalition

Fig. 2. The negotiation process for coalition formation, where the requester can ask its neighbours (a) for permission to join their existing coalitions or (b) to form a new coalition (if also the neighbour is not in a coalition)

requests to other singleton agents.

- Requests processing (lines 7 to 9): every agent who has received any request must decide whether the requesters can join its coalition or not. If the agent is a leader then it can directly accept the request. Otherwise, if the agent is singleton, then the negotiation process must advance to the next two phases[3].
- Confirmation (lines 10 to 12): every requester who was accepted (by a singleton neighbour) must confirm whether it still intends to form the new coalition.
- Coalition consolidation (lines 13 to 15): where new coalitions are created with the requesters who have confirmed their intention to join.

The sequence of these four phases characterizes the coalition creation flow, which is illustrated through two examples in Fig. 2. In the case of Fig. 2a, the two neighbours of agent 1 are in the same coalition. When agent 1 requests to join the coalition (phase 1), agent 7 accepts it immediately (phase 2), since $\{1, 7, 9\}$ has a greater value than $\{7, 9\}$. In the case of Fig. 2b, agent 3 decides to ask agent 8 to join its coalition (phase 1). As agent 8 is also singleton, it does not accept agent 3 immediately (phase 2). Instead, it asks if agent 3 still wants to join it. After agent 3 has confirmed its intention (on phase 3), agent 8 creates the new coalition $\{3, 8\}$ (phase 4).

Algorithm 2: Checking neighbours' coalitions

```
$NC_i \leftarrow \{C_j : \forall j \in N_i\}$;
foreach $C \subset NC_i$ do
    if $C \cup i$ $is\ not\ feasible$ then
        $NC_i \leftarrow NC_i \setminus C$;
    end
end
if $NC_i \neq \emptyset$ then
    $C_i^{bst} \leftarrow \arg\max_{C \subset NC_i} v(C)$;
    // ask leader of $C_i^{bst}$ to join it
    update $M^{rpl}_{C_i^{ack}}$;
end
```

The four negotiation phases are described in detail in the following sections.

B. Neighbours' coalitions checking phase

In the neighbours' coalitions checking phase, agents who are not in coalitions (requesters) verify the existence of coalitions in their neighbourhoods. Such a search behaviour is performed by Alg. 2, where all coalitions in agent i's neighbourhood are checked[4]. For each coalition, the procedure verifies whether it remains feasible if agent i enters it. Among the feasible coalitions identified, the procedure selects (and returns) the one with the highest value.

C. Requests processing phase

The requests processing phase is performed through Alg. 3. In this phase, agents who have received requests must decide whether or not the requesters can join their coalitions. As previously stated, this procedure can be performed by leader agents and also by singleton agents. This procedure is divided into: requests evaluation (lines 2 to 11) and decisions notification (lines 12 to 24).

As previously stated in Sect. V-A, two cases are taken into account when evaluating requests: (i) agent i is the leader of the coalition, and (ii) agent i is singleton. These cases are treated according to the amount of requests received. If multiple requests are received (no matter if agent i is singleton or not), then the agent must identify the best feasible coalition among the requesters, and accept only those who are part of it (as in line 10). Otherwise, if just one request is received, and agent i is the leader, then the requester can be directly accepted. Finally, if just one request is received, but agent i is singleton, then it only accepts the requester in the case of a mutual request[5].

[3] The rationale behind the second case (request received by a singleton agent) is that the requester can also have received requests after having made his request. In this sense, the confirmation is needed to ensure that the requester is still interested in joining the coalition.

[4] If there are no coalitions, then singleton neighbours are considered in this process.

[5] A request is mutual when two agents propose the same coalition to each other. In this case, only the agent with the lowest ID will accept the request of the other, thus becoming the leader of that coalition.

Algorithm 3: Processing requests

```
$C_i^{ack} \leftarrow \emptyset$;
if $|M_i^{rqs}| = 1$ then
    if $C_i \neq \{i\}$ and $C_i \cup M_i^{rqs}.requester$ is feasible then
        $C_i^{ack} \leftarrow C_i \cup M_i^{rqs}.requester$;
    else if $C_i = \{i\}$ and $C_i^{bst} = M_i^{rqs}.coalition$ and
    $i < M_i^{rqs}.requester$ then
        $C_i^{ack} \leftarrow \{i, M_i^{rqs}.requester\}$;
    end
else
    $C_i^{ack} \leftarrow C_i \cup best\_feasible(M_i^{rqs}.requesters)$;
end
if $C_i^{ack} \neq \emptyset$ then
    if $C_i \neq \{i\}$ then
        foreach $j \in C_i^{ack}$ do
            $C_j \leftarrow C_i^{ack}$;
        end
    else
        // send an accept to $C_i^{ack}$ agents
        $\forall j \in C_i^{ack}$, update $M_j^{rpl}$;
        $wait\_ph4? \leftarrow true$;
    end
    // send rejection to refused agents
    $\forall j \in M_i^{rqs}.requesters \setminus C_i^{ack}$, update $M_j^{rpl}$;
else
    // send a rejection message to all
    $\forall j \in M_i^{rqs}.requesters$, update $M_j^{rpl}$;
end
```

After all requests were evaluated, the requesters must be notified about the decisions made by agent i. If agent i is the leader of an existing coalition, then no further negotiation is necessary, i.e., the coalition can be formed immediately. Otherwise, agent i must send a notification message to every requester (accepting or rejecting their requests) in order to proceed to the next negotiation phases.

D. Confirmation phase

In the confirmation phase, requesters who have received positive notifications from the previous phase must confirm whether or not they still intend to form the new coalitions. This phase is performed on the basis of Alg. 4, only by requesters who have asked to form a new coalition with other singleton agents. Through this procedure, the agent is able to withdraw its request if a better one was received. Specifically, if agent i has received (in second phase) a request that is better than the one it has proposed (in first phase), then C_i^{ack} will not be empty (as in line 2). In this case, agent i confirms its intention only if the request was mutual[6] (as in lines 3 to 6), otherwise it refuses it. If agent i has not received a better request, then it simply confirms its interest of forming the initially proposed coalition.

Algorithm 4: Confirmation

```
1  $accept? \leftarrow false$;
2  if $C_i^{ack} \neq \emptyset$ then
3      if $C_i^{ack} = M_i^{rpl}.coalition$ and
4      $i > M_i^{rpl}.sender$ then
5          $accept? \leftarrow true$;
6      end
7  else
8      $accept? \leftarrow true$;
9  end
10 if $accept?$ then
       // send a confirmation message
11     update $M^{cnf}_{M_i^{rpl}.sender}$;
12     $wait\_ph4? \leftarrow true$;
13 else
       // send a cancellation message
14     update $M^{cnf}_{M_i^{rpl}.sender}$;
15 end
```

Algorithm 5: Processing confirmations

```
foreach $m \in M_i^{cnf}$ do
    if $m.wait\_ph4?$ then
        $C_{m.requester} \leftarrow C_i^{ack}$;
    end
end
```

E. Coalition consolidation phase

Finally, the coalition consolidation is performed as in Alg. 5. In this phase, the singleton agent (who has received the request in the second phase) creates the new coalition with the requesters who have confirmed their interest in forming the coalition with him (in the previous phase).

VI. Experiments and analysis

In this section we compare SACF against other CSG approaches, and afterwards we evaluate its performance in dynamic environments.

A. Methodology

To define $v(C)$ we use a characteristic function, that is realistic and suitable for smart grids domains, initially proposed in [7]. This characteristic function is formulated as in Eq. (5), where δ is the amount of energy the grid expects that a given coalition provides, ϵ defines the maximum monetary incentive the grid is willing to pay for a given coalition, E_C is the amount of energy that coalition C has available, and p is the normal price per energy unit.

$$v(C) = \begin{cases} 0, \text{ if } \exists i \in C, \exists j \in C \setminus i : d_j^i > \alpha \\ \min\{(\frac{E_C}{\delta})^2 \times \epsilon, \epsilon\} * p, \text{ otherwise} \end{cases} \quad (5)$$

According to the first line of Eq. (5), the values of unfeasible coalitions are set to zero. The second line, in turn,

[6] It is noteworthy that only the agent with the greatest ID operates this phase.

assigns the value of feasible coalitions. This takes into account the amount of energy (E_C) it has. The greater the E_C, the greater the coalition value[7]. The ϵ is used to limit the incentive given by the grid, i.e., in order to allow the grid to control the maximum value it wants to pay for an energy unit. Finally, through δ the grid defines the desired amount of energy[8] each coalition should have.

Regarding the values set to the parameters of the characteristic function, we set parameter ϵ to 0.9, i.e., the grid would pay up to 90% beyond the normal price to a coalition. In order to promote the granularity of the coalitions' amount of energy, we assume that the grid would like to form small VPPs, whose energy availability is around 50 kWh. Thus, δ was set to 50. Finally, the normal price of an energy unit, p, was set to R$0.50, which is approximately the energy price per kWh in Brazil (in Brazilian currency).

In order to evaluate our approach, two kinds of scenarios were designed: closed world and open world. In the closed world settings, no agent can enter or leave. The open world settings, in turn, allow new agents to enter the simulation, and existing agents to leave it. The focus of our approach is on open world scenarios, which are dynamic and more complex. However, we use closed world scenarios in order to compare our approach against other CSG algorithms, which do not work on dynamic scenarios.

The agents were randomly positioned in a grid based. Also, links were created between pairs of agents whose Euclidian distance is lower than α. For all scenarios tested, both in open and closed world settings, the parameter α was set[9] to 7.

SACF was implemented in NetLogo [25] version 5, while IDP, IP and CPCSG were implemented in Java version 1.6. All experiments were performed in an Intel(R) Core(TM) i7-2600 3.40GHz PC, with 16GB RAM and Ubuntu 12.04 64 bits.

B. Closed world settings

In the closed world case, SACF is compared against traditional CSG approaches: IDP [11], IP [10] and CPCSG [7], in terms of runtime and solution quality.

Experiments were run for different number of agents $a = \{10, 11, ..., 20\}$. For each number of agents, 30 different scenarios were generated (as described in Sect. VI-A). In order to accurately compare the algorithms, each of them was tested in exactly the same scenarios. The results are presented in Fig. 3. In the graph, each point shows the average runtime of the 30 scenarios and the error bars represent the standard deviation.

As seen in Fig. 3, SACF outperforms the other algorithms by many orders of magnitude. The average runtime of SACF was lower than 1 second in all sets of experiments, while in other algorithms the average runtime increases exponentially with the number of agents. For instance, for 20 agents the IP algorithm takes, on average, about 6 hours to run. SACF, in turn, takes less than 1 second.

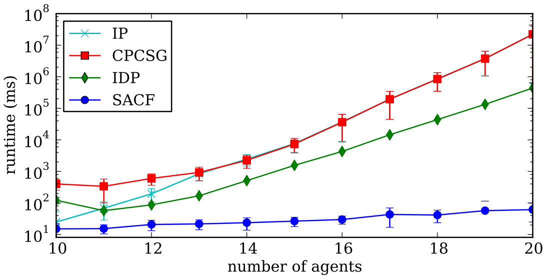


Fig. 3. Comparison of SACF against IDP, IP and CPCSG, showing the runtime (vertical axis, in log scale) for different number of agents (horizontal axis, for 10 to 20 agents)

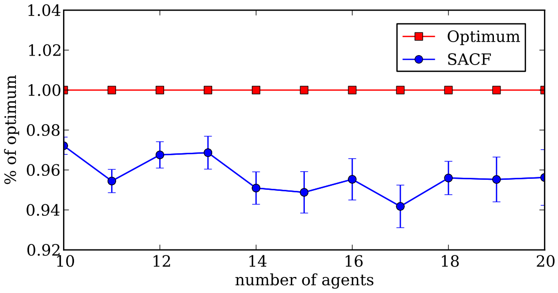


Fig. 4. Quality of solution generated by SACF, showing the percentage of SACF in comparison with the optimal solution (vertical axis), for different number of agents (horizontal axis, for 10 to 20 agents)

Now we analyse how far the solution generated by SACF is from the optimum. The results are plotted in the graph of Fig. 4, where is shown how far the SACF's average solutions are from the optimal ones. Results were normalised in order to show the percentage achieved by SACF in relation to the optimum. Error bars plot the standard deviation of each set of experiments. It is important to note that the non-normalised curves behave in an ascending monotonic fashion (as a function of the number of agents).

As shown in Fig. 4, the results are very good. Even though SACF has taken less than 1 second to run, it was able to find good solutions. In almost all tested cases, the solutions generated by SACF achieved more than 94% of the optimal solution. The average quality achieved was approximately 95.7% (averaged over all experiments). Also important, the standard deviation was at most 1.4% in all experiments. Therefore, SACF's advantage over the other algorithms is that it runs on a small amount of time, in dynamic and distributed environments, achieving good solutions in all tested cases.

C. Open world settings

In this section we empirically evaluate SACF in an open world setting. To this end, we have generated an initial scenario with 40 agents[10] (under the conditions described in Sect.

[7] The value of E_C is obtained from Eq. (3). The energy transfer rate e_i was set to 3.3 kW per hour for all agents. This is the energy transfer rate of some commercial EVs, such as Nissan Leaf.

[8] Despite the fact that the greater the amount of energy a coalition has, the better, δ is used to promote a better energy granularity. This parameter, however, does not prevent the formation of coalition whose amount of energy is greater than δ. Rather, it only guarantee that coalitions with δ or more energy will have the same value.

[9] In this work, the distance unit is not important, and it is measured by cells. However, one can imagine that each cell has 10x10 meters, as adopted in [7].

[10] Although SACF could be scaled for more than 40 agents, we empirically noticed that NetLogo took too long to run SACF for a larger number of agents.

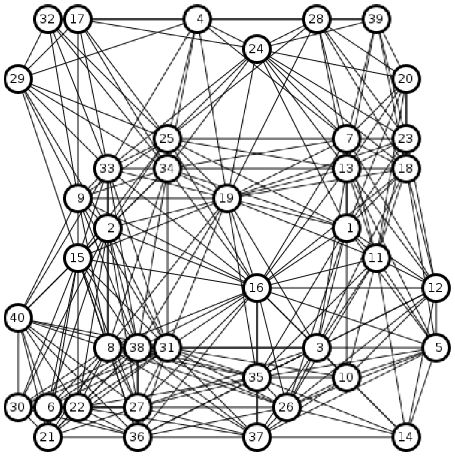


Fig. 5. Initial scenario with 40 agents used in the open world experiments

VI-A), which is used in the experiments along this section. This scenario is shown in Fig. 5.

Considering that the world is open and dynamic, agents can enter and leave the simulation while it is running. The frequency of such events is defined in terms of probabilities. Specifically, the probability of a new agent to enter in the system in a given time-step is defined by $\mathcal{P}_e$. In the same way, the probability of a given agent (selected in a random uniform way) to leave the system in a given time-step is defined by $\mathcal{P}_l$. Importantly, when an agent leaves the system, it notifies the leader of its current coalition (if it is in one) about that fact. In the experiments, we have simulated a 24-hour period, which corresponds to 1440 time-steps.

In order to evaluate how SACF behaves in different conditions, three different settings are tested in this section, all of them using the same 40 agents initial scenario. In all cases, small values are set to parameters $\mathcal{P}_e$ and $\mathcal{P}_l$, to avoid fast changes in the environment. These three settings are:

- Setting 1: $\mathcal{P}_e = \mathcal{P}_l = 0.005$.
- Setting 2: $\mathcal{P}_e = 0.02$ and $\mathcal{P}_l = 0.005$.
- Setting 3: $\mathcal{P}_e = 0.005$ and $\mathcal{P}_l = 0.02$.

In the first setting, both parameters have the same value 0.005, so the number of agents tends to remain stable along time, as it is confirmed in Fig. 6.

Concerning the value of the solutions, Fig. 7 plots the variation in the social welfare (the coalition structure value, $V(CS)$) over time. The social welfare does not experiment large variations. The greatest variations occur when an agent leaves the system, as can be observed by comparing the graphs of Figs. 6 and 7. Such behaviour shows that SACF is effective on organizing the agents into coalitions.

In order to better understand the organisational behaviour we show the variation in the instantaneous payoff along time in Fig. 8. Recall that the instantaneous payoff is obtained through Eq. (4). As shown, it remains stable over time. Here it is important to note that the instantaneous payoff is less than \$0.001, as this is the received value in just 1 minute.

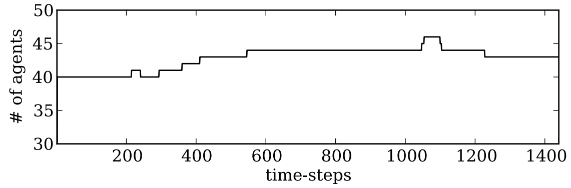


Fig. 6. Number of agents along time, for setting 1

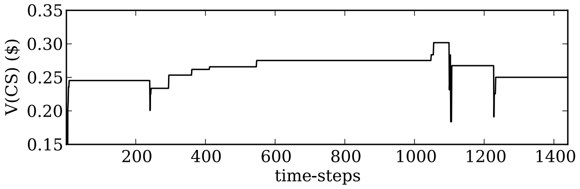


Fig. 7. Variation of V(CS) over time, for setting 1

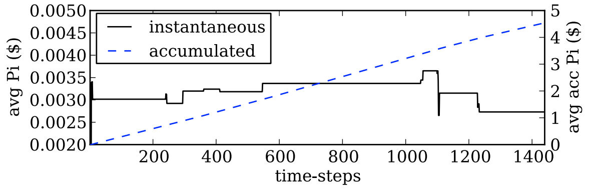


Fig. 8. Average instantaneous (left vertical axis) and accumulated (right vertical axis) payoff of agents over time, for setting 1

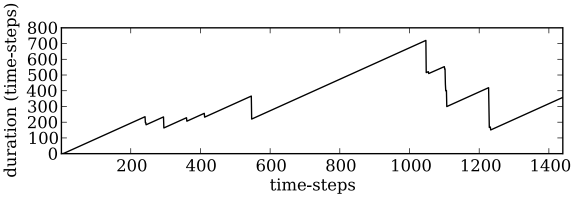


Fig. 9. Average duration of coalitions along time, for setting 1

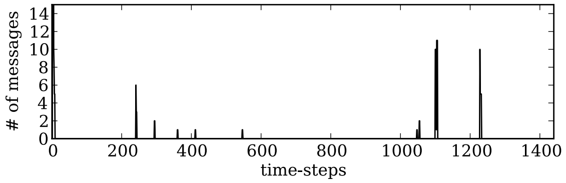


Fig. 10. Amount of messages exchanged among the agents over time, for setting 1

Remember that the payoff considered in this CFG is the value the grid pays beyond the normal price (as described in Sect. VI-A). In this sense, after the entire 24 hours simulation, the average payoff was of approximately \$4.54 per agent. Also, considering the total profit (normal price + coalition price) obtained by the agents, the average was of \$44.14. In this respect, an agent which worked as a singleton throughout all simulation has received only \$39.6. Therefore, agents who worked in coalitions have received, on average, a profit 11.46%

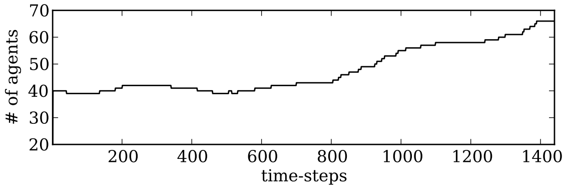


Fig. 11. Number of agents along time, for setting 2

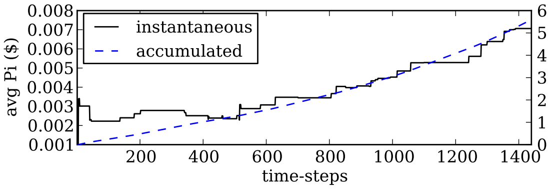


Fig. 12. Average instantaneous (left vertical axis) and accumulated (right vertical axis) payoff of agents over time, for setting 2

greater than singleton agents.

Concerning the system stability, another important point is the amount of time the coalitions last. Fig. 9 shows the instantaneous average duration of the coalitions. It can be observed by comparing Fig. 9 with Fig. 6 that whenever an agent enters or leaves the system, new coalitions are created. Consequently, the curve of Fig. 9 drops significantly. In the end of the simulation, the average duration of the coalitions was approximately 330.94 time-steps.

To conclude the experiments with the first setting, Fig. 10 shows the number of messages[11] exchanged along time. As expected, the amount of exchanged messages increases whenever an agent enters or leaves the system. Such behaviour can be observed by comparing Figs. 6 and 10.

Now we analyse the second setting, where $\mathcal{P}_l$ remains with the same value as in setting 1, but $\mathcal{P}_e$ is increased to 0.02, i.e., 4 times greater than $\mathcal{P}_l$. In this way, as expected, the number of agents increases over time (see Fig. 11).

Concerning the agents' payoff, Fig. 12 presents the average instantaneous payoff of the agents along time. The behaviour of the instantaneous payoff curve is less stable than in the case of setting 1. Obviously, this occurs because, in setting 2, the entering of new agents in the system is much more frequent. However, it can be noted that the payoff increases as well. This means that agents who have joined coalitions obtained greater total profits than singleton agents, on average, as the value of the coalitions increase as these become greater.

The other metrics analysed in setting 1 are very similar for setting 2. Thus, we go straight to setting 3, where $\mathcal{P}_e = 0.005$ and $\mathcal{P}_l = 0.02$. In this way, the number of agents over time is expected to decrease. These results are depicted in Fig. 13. In the case of setting 3, the final number of agents (in the end of simulation) has dropped by half.

In the case of setting 3, the average payoff received by the agents after the whole simulation was approximately 3.1,

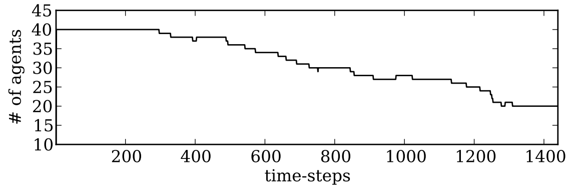


Fig. 13. Number of agents along time, for setting 3

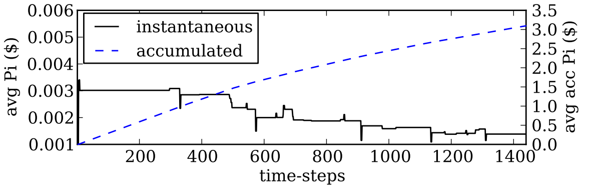


Fig. 14. Average instantaneous (left vertical axis) and accumulated (right vertical axis) payoff of agents over time, for setting 3

which corresponds to an average profit of \$42.7. Considering that the profit obtained by singleton agents was of \$39.6, agents in coalitions have obtained an improvement of 7.8% on average. We remark that, even though this value is lower than in the other two settings, the average value over time is not much affected during the simulation. In other words, the agents obtain a good payoff even with constant changes in the environment.

VII. Conclusions and future work

In this paper we have presented the SACF, a distributed, flexible, robust and adaptive method for DCSG in smart grids, regarding constraints imposed by the infrastructure. The proposed approach works by allowing agents to negotiate the formation of local coalitions among themselves. On this basis, the agents can propose to join their neighbours, forming coalitions.

Based on experiments, we showed that SACF is really effective on providing good solutions in a fast and fully distributed way. We showed that, compared to classical algorithms, our approach outperforms them by several orders of magnitude, providing solutions whose quality was above 94% of the optimum (with a standard deviation of 1.4%) in all tested cases. In dynamic environments, SACF has shown very good results, being able to keep the agents' payoff sufficiently stable over time. Besides, SACF can manage an arbitrary number of agents in much less time than classical approaches.

For future work, we would like to investigate new agreement mechanisms in order to make the approach more realistic and to increase the coalitions' values. Issues such as price changes, penalties (for unreliable EVs), different kinds of DERs (with different preferences and battery features) and a more flexible distance heuristic may be considered. Also important, scenarios involving uncertainties (i.e., scenarios where the agents are not able to accurately predict their availability) should be investigated. Finally, probabilistic models could also be incorporated, allowing the agents to explore new coalitions in order to improve the global utility.

[11] In order to improve the visualisation, the vertical axis range was set between 0 and 15. In the very beginning, around 40 messages were exchanged.

Acknowledgment

The authors would like to thank the anonymous reviewers for their valuable suggestions and comments. Gabriel Ramos and Ana Bazzan are partially supported by CNPq and FAPERGS grants. Juan C. Burguillo is supported by the European Regional Development Fund (ERDF), and the Galician Regional Government under projects CN 2012/260 (Consolidation of Research Units: AtlantTIC) and CN 2011/021

References

[1] U. S. Department of Energy, "Grid 2030: A national vision for electricity's second 100 years," Jul 2003.

[2] W. Kempton and J. Tomić, "Vehicle-to-grid power implementation: From stabilizing the grid to supporting large-scale renewable energy," *Journal of Power Sources*, vol. 144, no. 1, pp. 280–294, 2005.

[3] D. Pudjianto, C. Ramsay, and G. Strbac, "Virtual power plant and system integration of distributed energy resources," *Renewable Power Generation, IET*, vol. 1, no. 1, pp. 10–16, Mar 2007.

[4] M. Vasirani, R. Kota, R. Cavalcante, S. Ossowski, and N. Jennings, "Using coalitions of wind generators and electric vehicles for effective energy market participation," in *Proc. of 10th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 11)*, May 2011, pp. 1099–1100.

[5] S. Kamboj, W. Kempton, and K. S. Decker, "Deploying power grid-integrated electric vehicles as a multi-agent system," in *Proc. of 10th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 11)*, Taipei, Taiwan, May 2011, pp. 13–20.

[6] N. Matta, R. Rahim-Amoud, L. Merghem-Boulahia, and A. Jrad, "A cooperative aggregation-based architecture for vehicle-to-grid communications," in *Global Information Infrastructure Symposium (GIIS)*, 2011, pp. 1–6.

[7] G. O. Ramos and A. L. C. Bazzan, "Reduction of coalition structure's search space based on domain information: an application in smart grids," in *Proc. of 3rd Brazilian Workshop on Social Simulation (BWSS)*, Curitiba, Brasil, Oct 2012, pp. 112–119.

[8] A. L. C. Bazzan and S. R. Dahmen, "Targeted search in coalition structures," in *Proc. of 5th European Workshop on Multi-agent Systems*, Hammamet, Tunisia, Dec 2007, pp. 180–192.

[9] T. Sandholm, K. Larson, M. Andersson, O. Shehory, and F. Tohmé, "Coalition structure generation with worst case guarantees," *Artificial Intelligence*, vol. 111, no. 1–2, pp. 209–238, 1999.

[10] T. Rahwan, S. D. Ramchurn, V. D. Dang, and N. R. Jennings, "Near-optimal anytime coalition structure generation," in *Proc. of 20th Int. Joint Conf. on Artificial Intelligence (IJCAI)*, 2007, pp. 2365–2371.

[11] T. Rahwan and N. R. Jennings, "An improved dynamic programming algorithm for coalition structure generation," in *Proc. of 7th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 08)*, Estoril, Portugal, May 2008, pp. 1417–1420.

[12] B. Horling and V. Lesser, "A survey of multi-agent organizational paradigms," *Knowledge Engineering Review*, vol. 19, no. 4, pp. 281–316, 2004.

[13] O. Shehory and S. Kraus, "Methods for task allocation via agent coalition formation," *Artificial Intelligence*, vol. 101, no. 1–2, pp. 165–200, 1998.

[14] T. Rahwan and N. R. Jennings, "Coalition structure generation: Dynamic programming meets anytime optimisation," in *Proc. of 23rd Conf. on Artificial Intelligence (AAAI 2008)*, Estoril, Portugal, May 2008, pp. 156–161.

[15] D. Y. Yeh, "A dynamic programming approach to the complete set partitioning problem," *BIT Numerical Mathematics*, vol. 26, no. 4, pp. 467–474, 1986.

[16] S. Ueda, A. Iwasaki, M. Yokoo, M. Silaghi, K. Hirayama, and T. Matsui, "Coalition structure generation based on distributed constraint optimization," in *Proc. of 24th AAAI Conference on Artificial Intelligence (AAAI 10)*. AAAI Press, Jul 2010, pp. 197–203.

[17] G. Chalkiadakis and C. Boutilier, "Sequentially optimal repeated coalition formation under uncertainty," *Autonomous Agents and Multi-Agent Systems*, vol. 24, no. 3, pp. 441–484, May 2012.

[18] S. D. Ramchurn, M. Polukarov, A. Farinelli, N. Jennings, and C. Trong, "Coalition formation with spatial and temporal constraints," in *Proc. of 9th Int. Conf. on Autonomous Agents and Multi-Agent Systems (AAMAS 10)*, 2010, pp. 1181–1188.

[19] S. McArthur, E. Davidson, V. Catterson, A. Dimeas, N. Hatziargyriou, F. Ponci, and T. Funabashi, "Multi-agent systems for power engineering applications – part i: Concepts, approaches, and technical challenges," *IEEE Transactions on Power Systems*, vol. 22, no. 4, pp. 1743–1752, Nov 2007.

[20] S. Ramchurn, P. Vytelingum, A. Rogers, and N. Jennings, "Putting the "smarts" into the smart grid: A grand challenge for artificial intelligence," *Communications of the ACM*, vol. 55, no. 4, pp. 86–97, 2012.

[21] G. Chalkiadakis, V. Robu, R. Kota, A. Rogers, and N. R. Jennings, "Cooperatives of distributed energy resources for efficient virtual power plants," in *Proc. of 10th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 11)*, Taipei, Taiwan, May 2011, pp. 787–794.

[22] S. Kamboj, K. Decker, K. Trnka, N. Pearre, C. Kern, and W. Kempton, "Exploring the formation of electric vehicle coalitions for vehicle-to-grid power regulation," in *Proc. of 1st Int. Workshop on Agent Technology for Energy Systems (ATES 10)*, 2010, pp. 1–8.

[23] R. C. Mihailescu, M. Vasirani, and S. Ossowski, "Dynamic coalition adaptation for efficient agent-based virtual power plants," in *Multiagent System Technologies*, ser. Lecture Notes in Computer Science, F. Klügl and S. Ossowski, Eds. Springer Berlin Heidelberg, 2011, vol. 6973, pp. 101–112.

[24] ——, "An organizational approach to agent-based virtual power stations via coalitional games," in *Highlights in Practical Applications of Agents and Multiagent Systems*, ser. Advances in Intelligent and Soft Computing, J. Pérez, J. Corchado, M. Moreno, V. Julián, P. Mathieu, J. Canada-Bago, A. Ortega, and A. Caballero, Eds. Springer Berlin Heidelberg, 2011, vol. 89, pp. 125–134.

[25] U. Wilensky, "Netlogo," 1999, Center for Connected Learning and Computer-Based Modeling, Northwestern University, Evanston, IL. [Online]. Available: http://ccl.northwestern.edu/netlogo