Forming Coalitions of Electric Vehicles in Constrained Scenarios

Ana L. C. Bazzan and Gabriel de O. Ramos

Instituto de Informática
Universidade Federal do Rio Grande do Sul
bazzan@inf.ufrgs.br, goramos@inf.ufrgs.br

Abstract. Finding an optimal coalition structure is a hard problem. In order to simplify this process, it is possible to explore some characteristics of the agents organization. In this paper we propose an algorithm that deals with a particular family of games in characteristic function, but is able to search in a much smaller space by considering organizational issues such as constraints in the number of participants. We apply this approach to the domain of smart grids, in which the aim is to form coalitions of electric vehicles in order to increase their reliability when supplying energy to the grid.

1 Introduction

A coalition can be defined as a group of agents that decide to cooperate in order to achieve a common goal. An outcome of a coalitional game is a partition of the set of all players into coalitions, together with an action for each coalition. However, partitioning agents in coalitions is not a trivial problem. In multiagent systems (MASs), there has been an increasing interest in coalition formation involving cooperative agents in order to deal with problems such as task allocation [1]. In general, the research on CGs tackles one or more of these activities: coalition structure generation (CSG hereafter); solving the joint problem; dividing the value among the members of the coalition. The former and the latter have been extensively studied in game-theory, while the second is tackled by the MASs community. This paper focuses on the first (as other works mentioned in the next section) since this is a key aspect for effective organizations in MASs. The bad news is that this problem is equivalent to the set partitioning one, which is NP-complete [2]. In the general case, the number of coalition structures is so large ($O(a^a)$, where a is the number of agents) that it cannot be enumerated for more than a few agents.

The first algorithm to establish a bound with a minimal amount of search was given in [2] (henceforth S99). In 2004, [3] (henceforth DJ04) provided a faster algorithm that is within a finite bound from the optimum and is faster than the S99. Then, Rahwan *et al.* have proposed further algorithms [4–6]. The former (henceforth IP) introduced a near-optimal anytime algorithm for coalition structure generation, which partitions the space in terms of coalitions of

particular sizes. In [5] (henceforth IDP) an algorithm was proposed, which is based on dynamic programming as was the case in [7]. The hybrid of IDP and IP (IDP-IP) was a breakthrough, but it is still in $O(3^a)$. In [6], the quality guarantees provided by [2] were improved. Specifically, it was shown that there is a minimum search required to establish any bound in any subset of the search space. However, a new question has emerged, which regards how to identify the coalition structures that belong to such subsets. These and other works are based on a search in a coalition structure graph (see [2] for an explanation). In these graphs, each node corresponds to a coalition structure. Nodes are grouped in levels according to the number of partitions they have (e.g., coalition structures comprising two coalitions appear in the second level of the graph, and so on). An arc between two nodes represent (i) a splitting of one coalition of the CS into two, or (ii) a merging of two coalitions into a single one. Based on such representation of the search space, [2] showed that searching the two lowest levels of the graph is enough to establish a bound a on the solution quality.

Despite the fact that a bound can be established in linear time in the size of the input, the bad news is that 2^{a-1} nodes of the coalition structure graph have to be searched in order to guarantee the worst case bound from optimum. This prevents the use of that kind of algorithm for a high number of agents. Even if some researchers have shown that it is possible to lower the bound with further and/or smarter search, due to the aforementioned complexity problem, this road is unlike to lead to efficiency when it comes to real-world applications with more than 30 agents. Rather, it is necessary to count on something for which AI is well known, namely exploitation of domain knowledge.

Thus, in the present paper we rely on domain knowledge to define a particular class of CSG in which coalitions are constrained in their size. The knowledge employed here comes from the domain of smart grids[1]. Specifically, we take advantage of the fact that, in order to increase the robustness of providing energy to the grid, eletric vehicles (EVs) must work together in so-called virtual power plants (VPPs). However, to this aim, EVs cannot be arbitrarily located (due to energy transmission constraints), nor can the VPP be arbitrarily large. Thus coalitions of EVs are constrained with respect to their size and distance. Besides, in the game structure proposed here, searching in the two lowest levels of the coalition structure graph, as in S99, is not necessarily the best strategy. For example, electrical vehicles are likely to be partitioned in more than two coalitions so that it seems illogical to start looking at partitions of size two.

We consider games that are neither superadditive nor subadditive (same as in S99, DJ04, IP, IDP, and others) but whose characteristic function has the property of being monotonically increasing in a certain interval. This property allow us to direct the search to levels other than those proposed previously. As we show, for particular characteristic functions, the coalition structure (CS) with maximum value can be located by visiting one single level of the CS graph.

[1] We remark that the CSG class defined here is general, and it is not restricted to the smart grid problem.

This paper is organized as follows. In the next section we give a motivation for forming coalitions in the smart grid domain. We also discuss the background of generation of CSs as well as some of the algorithms already mentioned. Following, in Section 3 we formalize our approach and discuss some of its variants. Examples of its utilization in the smart grid domain are also discussed. In Section 4 we show experimental results, which are then followed by a conclusion (Section 5).

2 Smart Grids: the example of electric vehicles

In most countries, electric power generation strongly depends on non-renewable, polluting energy sources whose availability is becoming scarce. Furthermore, environmental and economic impacts resulting from the use of non-renewable energy sources have become an issue in our society. In order to reverse this scenario, many governments have been focusing on the transition to a low carbon economy. One of the major challenges arising from this transition refers to the modernization of the energy infrastructure (the grid). Electricity grids have evolved very little over the years. In this context, the concept of smart grids emerges, which is described by the U. S. Department of Energy [8] as "a fully automated power delivery network that monitors and controls every customer and node, ensuring a *two-way flow* of electricity and information between the power plant and the appliance [...]". The smart grid concept is closely related to computing, and in particular, given its essentially distributed nature, smart grids represent a large field of research for MASs [9, 10].

In addition to the concept of smart grids, recently the concept of vehicle-to-grid (V2G) has drawn attention, in which electric vehicles (EVs) can provide part of the energy stored in their batteries back to the grid when needed. This mechanism becomes important in situations where the grid relies on intermittent renewable energy, as it is the case of energy provided by wind turbines and solar panels. In such cases, the energy stored in the EVs' batteries can be provided back to the grid when production is not able to meet the demand.

The use of V2G mechanisms has received great attention of many researchers who associate agents and smart grids, as in [11–16]. In essence, the concept of V2G can be seen as a distributed energy storage system. However, EVs are unable to operate alone in a reliable and cost-effective way due to insufficient energy capacity and due to lack of full availability (EVs can supply energy back to the grid just when they are not in use). Therefore, coalition formation is an efficient way to increase the reliability on EVs' supply of energy.

Regarding other works, many of these address coalition formation between distributed energy resources (DERs) to form VPPs. DERs are renewable energy generators with small to medium energy capacity, like wind turbines, solar panels etc. Taking into account that renewable energy sources are intermittent due to weather conditions, many approaches suggest grouping DERs to promote reliability and efficiency on production predictions. Examples are [11–14]. The former focuses on mechanism design rather than in coalition structure generation, disregarding how far the solution is from the optimal one; in the others,

coalition structure generation is made in an ad hoc fashion, without necessarily concerning the optimal solution.

It is then clear that most existing works have given greater importance to applications of smart grids than to coalition formation itself. A more concrete step into this direction was given in [15, 16], where the coalition formation process was extended to dynamic scenarios. In these works, coalition formation is locally negotiated by the agents. However, despite promising experimental results, no quality guarantees were provided.

3 Finding Optimal Coalition Structures for a Sub-class of CFGs: Smart Grid Problems

3.1 Coalition of Electric Vehicles

As mentioned, the motivation for forming coalitions of EVs is that they cannot operate alone in a reliable way [17]. Basically, the capacity/effectiveness of a VPP on supplying energy to the grid is proportional to its size (number of EVs). On the other hand, it is known that the energy loss is proportional to the distance of transmission. Thus, the size of coalitions must be restricted to, for instance, EVs that are within a certain radius. An example could be all EVs parked in a given building. Finally, in some circumstances, VPPs may be able to include not only EVs but also other DERs such as wind turbines. In this case, it is clear that a coalition that includes such a turbine shall have a higher value, since a turbine is a more valuable player in the coalition. The reason for this is the fact that turbines have a higher availability.

Grounded on these facts, we introduce a particular class of CFGs, where coalitions are restricted in their sizes, and show how the optimal CS can be found by searching a single level of the CS graph. The rest of this section first introduces the notation and shows how to find the level l where the CS with maximal value is located (for the particular class of CFGs we tackle here). It then discusses a case in which the CFG is modified to include agents that bring more value to the coalition, and presents special configurations and constraints that allows us to find level l in a straightforward way.

3.2 Definitions

The notation used in this paper is the following:

- A: the set of agents (here EVs), with cardinality $a = |A|$;
- $K \subseteq A$: a coalition of size $|K|$;
- $v(K)$: value of coalition K, given by a characteristic function (CFG) defined on the set of all coalitions (2^a). In the general case, the CFG is defined as $v : 2^a \rightarrow \mathbb{R}$ with $v(\emptyset) = 0$.
- CS is a coalition structure, i.e., a particular partition of A into disjoint and exhaustive sets; the value of the structure $V(CS)$ is given by $V(CS) = \sum_{K \in CS} v(K)$;

- $CS^\star$ is the optimal CS, i.e., the one with maximal value;
- l is a level in the CS graph, meaning that the l-th level has l subsets in each coalition structure (for example, in level $l = 2$ all coalition structures have two coalitions each).

We consider a subclass of CG's which has CFG's with the following structure[2]:

- $v(\emptyset) = 0$
- $v(K) = 0$ if $|K| = 1$
- $v(K) = 0$ if $|K| \geq \alpha$ with $\alpha = \lfloor \frac{a}{\rho} \rfloor$, $\rho = \{1, 2, ..., \frac{a}{3}\}$
- $v(K) = \varphi(K)$ if $2 \leq |K| < \alpha$

This means that coalitions of size one, α, or bigger than α have $v = 0$, and that we wish to partition A in at least two coalitions. Note that this value (two coalitions) can easily be changed, as well as the fact that in what follows we use $\varphi(K) = |K|$.

We show here that, for CFG's as above, the coalition structure with maximum value can be located by visiting one single level of the CS graph. This is similar to the algorithm S99 (which visits levels 1 and 2 of the CS graph) but by visiting level l we guarantee finding the $CS^\star$ while they found a bound. It is important to notice that more than one CS may have the same maximum value. If this is the case, the maximum value may occur in other levels as well, but visiting the level our algorithm computes, it is guaranteed to find at least one of the CSs with maximum value.

After the level containing $CS^\star$ is found, a search must be performed in order to find the $CS^\star$ itself. The size of this search space is given by the Stirling number of the second kind $S(a, l)$, which gives the number of partitions of a set A into l non empty sets: $S(a, l) = \frac{1}{l!} \sum_{i=0}^{l-1} (-1)^i \binom{l}{i} (l - i)^a$

3.3 General Structure of the Algorithm

The level l that has the CS with maximum value ($CS^\star$) is computed by Eq. 1, where:

- $1 \leq j \leq (\alpha - 1)$
- $v(K_j)$ is the value of the coalition of size j
- $i = \lfloor \frac{a}{j} \rfloor$
- $\gamma = \begin{cases} 0 \text{ if mod(a,j)=0} \\ 1 \text{ otherwise} \end{cases}$

$$l = \arg\max_i [i \times v(K_j)] + \gamma \tag{1}$$

[2] As in previous algorithms, we also assume that $v(K) \geq 0$ for any K (or at least those CSs visited by the algorithm), which is not a restrictive assumption given that the CFG can be normalized.

The rationale behind Eq. 1 is that the level l where $V(CS^\star)$ is to be found is the one where it is possible to put the highest number of coalitions with high value, with the least remainder (zero if possible). To do this, it is necessary to test $(\alpha - 1)$ values for $(i \times v(K_j))$ in order to find the maximum value for this product[3]. This maximum corresponds to one or more values of i. When more than one exists, one must use some domain dependent heuristic, or select the i that implies the smallest search space (easily computed by the Stirling number of the second kind for the actual determination of the $CS^\star$.

Notice that Eq. 1 returns the *level* where $CS^\star$ is to be found, not necessarily the $CS^\star$ itself. This only happens if i is a factor of a. In this case, $V(CS^\star)$ is found after $(\alpha - 1)$ computations of $(i \times v(K_j))$, without further search in the space spanned by the Stirling number of the second kind. Here the algorithm finds $V(CS^\star)$ linearly in the number of agents. However, this is not always the case: among the $(\alpha - 1)$ values for $(i \times v(K_j))$, it is not necessarily the case that we find $V(CS^\star)$ because i might not be a factor of a. Thus, in the most general case, level l (but only this level) must be searched completely to find $CS^\star$.

Proposition 1: $CS^\star$ is located in level l given by Eq. 1.

Proof:

Call $(i^\star, j^\star)$ the pair (i, j) that maximizes $(i \times v(K_j))$. When $i^\star$ is a factor of a, the proof is trivial because consequently $j^\star$ is also a factor and there is no better way to partition a (remember that the maximum $(i \times v(K_j))$ was selected). Other partitions can have at most the same value, but never exceed the one given by $(i^\star, j^\star)$.

If i is not a factor of a, then there are $i^\star$ coalitions (of size $j^\star$ each) plus a remainder. These are:

$\{1, ..., j^\star\}\{j^\star + 1, ..., 2j^\star\}...\{(i^\star - 1)j^\star + 1, ..., i^\star j^\star\}\ \{i^\star j^\star + 1, ..., a\}$

When $a = i^\star j^\star + 1$, then the last set has only one element and hence value zero. Therefore, the whole CS has value smaller than a (in fact $a - 1$). If we take one element out of any of the $i^\star$ sets (say, the first), then this one will have value $(j^\star - 1)$ and the last set will have value 2. The total value of the CS is then: $(j^\star - 1) + (i^\star - 1)j^\star + 2 = i^\star j^\star + 1 = a$.

When $a > i^\star j^\star + 1$, then the last set has a non zero value and thus all sets sum to a. □

Next we give examples of how to compute l. In the tables, hatched rows represent those l's where $CS^\star$'s can be found.

Example 1: $a = 20$ EVs, $\alpha = 5$.

j	i	$(i \times v(K_j))$	l
1	20	0	
2	10	20	10
3	6	18	
4	5	20	5

[3] This comes from the fact that, for $j \geq \alpha$ coalitions of size j have value zero and hence $(i \times v(K_j))$ has value zero as well. Thus it makes no sense to compute i's for $j \geq \alpha$.

In this case the least search space refers to $i = 5$, thus Eq. 1 returns $l = 5$. As here i is a factor of a, $V(CS^\star)$ is found directly.

Example 2: $a = 20$ EVs, $\alpha = 4$.
The table is the same as in Example 1, without the last line. In this case, Eq. 1 returns $l = 10$.

Example 3: $a = 13$ EVs, $\alpha = 4$.

j	i	$(i \times v(K_j))$	l
1	13	0	
2	6	12	7
3	4	12	5

In this case, $l = 5$ yields less search than $l = 7$. Thus $i = 4$, and i is not a factor of a. This value for i corresponds to $l = 5$ (see table). In this level, there are partitions like this: 4 coalitions of size 3 and one of size 1, which is not optimal. However, as proven above, level $l = 5$ has also the $CS^\star$: it has 3 coalitions of size 3 and 2 of size 2.

3.4 Coalitions with More Valuable Players

The games with CFG's as in the given examples do not distinguish between individual values of agents (here, DERs). In some cases, more valuable DERs aggregate an additional value $\delta > 1$ to the coalitions they are in.

Games with more valuable agents require slight modifications in the CFG previously defined. We define $G \subset A$ as the set of more valuable agents, and:

- $v(\emptyset) = 0$
- $v(K) = 0$ if $|K| = 1$
- $v(K) = 0$ if $|K| \geq \alpha$ with $\alpha = \lfloor \frac{a}{\rho} \rfloor$, $\rho \in \{1, 2, ..., \frac{a}{3}\}$
- $v(K) = \varphi(K)$ if $2 \leq |K| < \alpha$ and $G \cap K = \emptyset$
- $v(K) = \varphi(K) \times \delta$ if $2 \leq |K| < \alpha$ and $G \cap K \neq \emptyset$

We now show that when $G \neq \emptyset$, the level where the value of CS is maximum can still be found by Eq. 1. However, because $\delta > 1$, bigger coalitions have more value and hence l should be as low as possible. The modification is given in Eq. 2.

$$l = min[\arg\max_i [i \times v(K_j)] + \gamma] \tag{2}$$

For the sketches of the proofs, we assume (without loss of generality) that the coalitions with additional values are the first ones. Each of these have value $j^\star \times \delta$.

We first show that no other level below l contains CSs with higher value than that found in level l. In the level $(l-1)$, there are $(i^\star - 1)$ coalitions. In respect to level l, this means that some coalitions somehow join to form this additional set. The highest possible value for the CSs in level $(l-1)$ occurs when the agent(s) that bring additional value remain partitioned as in level l. Any two

other coalitions that join together would have size higher than $(\alpha - 1)$ and hence value zero. Thus no CS would have higher value in level $(l - 1)$. This holds for lower levels as well due to the same reason.

We now look at level $l + 1$, where one coalition splits into two. If this split yields one coalition of size exactly one (e.g., splitting of a coalition of size 3), then this has value zero and hence the total value is reduced. If the split yields two coalitions of size greater than one, then the two coalitions together will have no more value than when they were together, so the value increases in no case.

Example 4: consider that there are 2 wind turbines and 14 EVs, thus $a = 16$, $A = \{A1, A2, ..., A16\}$, $G = \{A1, A2\}$. Assuming $\alpha = 5$:

j	i	$(i \times v(K_j))$	l
1	16	0	
2	8	16	8
3	5	15	
4	4	16	4

Here $l = 4$ is better than $l = 8$, thus one of the $CS^{\star}$s returned after searching this level is given by: $\{\mathbf{A1}, A3, A4, A5\}, \{\mathbf{A2}, A6, A7, A8\}, \{...\}, \{A13, ..., A16\}$ (value $4\delta + 4\delta + 4 + 4 = 8\delta + 8$). In level $l = 3$ one of the CS's that has the highest value is formed by merging coalitions that do not contain A1 or A2 (otherwise these players would find themselves in a coalition of value zero because it has a cardinality higher than α): $\{\mathbf{A1}, A3, A4, A5\}, \{\mathbf{A2}, A6, A7, A8\}, \{A9, .., A16\}$ (value is only 8δ). In level $l = 5$, splitting coalitions with agents A1 or A2 makes no sense, so the best case is when one of the others split (say, the last): $\{\mathbf{A1}, A3, A4, A5\}, \{\mathbf{A2}, A6, A7, A8\}, \{...\}, \{A13, A14\}, \{A15, A16\}$ (value $8\delta + 8$). Thus in level $l = 4 + 1$, there is no increase in the value.

This example illustrates the above proof that the level where the value of CS is maximum remains as given by Eq. 1 when some agents aggregate an additional value of δ to the coalition they join.

3.5 Heuristics for Finding Level l

Eq. 1 is general and computes the level where the CS of maximum value is in any case (provided the CFG has the structure presented). However, the algorithm to perform the calculation still has to be performed $\alpha - 1$ times in order to find the i for which the product $(i \times v(K_j))$ is maximum. In some special cases, we can compute the level l without using Eq. 1.

Case one: α is a factor of a, and $\rho < \alpha$. For this case, the following holds:

Proposition 2: $l = \alpha$

Lemma 1: If α is a factor of a (and thus ρ is also a factor of a), then dividing A in ρ subsets of size α yields $v(CS) = 0$.
Proof: if A is equally partitioned into ρ sets, then each set has $\frac{a}{\rho} = \alpha$ elements as follows:

$\{1, ..., \alpha\}\{\alpha+1, ..., 2\alpha\}\{2\alpha+1, ...3\alpha\}... ...\{(\rho-1)\alpha+1...a\}$. Clearly, each has value zero because its size is $\frac{a}{\rho}$. □

Lemma 2: If α is a factor of a, and $\rho < \alpha$, then dividing A in α subsets yields $v(CS) = a$.
Proof: dividing A in α subsets (each of size $\frac{a}{\alpha} = \rho$):
$\{1, ..., \rho\}\{\rho+1, ..., 2\rho\}\{2\rho+1, ..., 3\rho\}...\{(a-\rho)-1, ...a\}$
Because $\rho < \alpha$, the value of this CS is $\frac{a}{\rho} \neq 0$ (number of subsets) multiplied by ρ (value of each subset), which yields the value a. □

We can now prove Proposition 2 ($l = \alpha$) by showing that, since we cannot divide A in ρ subsets (Lemma 1), we divide it in α subsets and this coalition structure has value a. This is trivial because, since $\alpha > \rho$, this means that dividing A in α subsets yields more subsets of *smaller* size. Lemma 2 shows that the CS so formed has value a.

Case two: a similar case happens when ρ is a factor of a but α is not. Then, $l = \frac{a}{\rho}$ because ρ is a factor of a and so A can be partitioned into $\frac{a}{\rho}$ subsets. Since $\rho < \alpha$, this means that sets of size ρ do have a value that is not zero (Lemma 2). Hence the value of all $\frac{a}{\rho}$ sets of individual value ρ are $\frac{a}{\rho} \times \rho = a$.

Case Three: $\rho \geq \alpha$. Then, the following holds:

Proposition 3: $l = \lfloor \frac{a}{\alpha-1} \rfloor + \beta$ where $\alpha \geq 2$ and $\beta = 0$ if $\text{mod}(a, \alpha-1) = 0$ and $\beta = 1$ otherwise.
Proof: according to Lemma 1, dividing A in ρ subsets yields $v(CS) = 0$. Since $\rho \geq \alpha$, dividing A in ρ subsets would result in less subsets which would have even bigger sizes and hence value zero as well. Therefore, the solution is to divide A in more subsets of smaller sizes (but as big as possible as those have higher values). This means dividing A in a number of sets having size $(\alpha-1)$ each. There are at least $\lfloor \frac{a}{\alpha-1} \rfloor$ of these subsets. If $(\alpha-1)$ is a factor of a, then there are exactly $\frac{a}{\alpha-1}$ of them; otherwise there is an additional subset which has the remainder of the division of a by $(\alpha-1)$, i.e. there are $\lfloor \frac{a}{\alpha-1} \rfloor + 1$ subsets.

We first handle the case $(\alpha-1)$ is a factor of a.

Let $\lfloor \frac{a}{\alpha-1} \rfloor = \lambda$. Because $(\alpha-1)$ is a factor of a, we can also write $\lambda = \frac{a}{\alpha-1}$. The λ subsets of A that have size $(\alpha-1)$ each are:
$\{1, ..., \alpha-1\}\{\alpha, ..., 2(\alpha-1)\}...\{(\lambda-1)(\alpha-1)+1, ..., a\}$, where each subset has a value $(\alpha-1)$. Since there are $\frac{a}{\alpha-1}$ of them, the value is a. In case $(\alpha-1)$ is *not* a factor of a (remember that here $\lambda = \lfloor \frac{a}{\alpha-1} \rfloor$), the subsets of A in this case are:
$\{1, ..., \alpha-1\}\{\alpha, ..., 2(\alpha-1)\}...\{(\lambda-1)(\alpha-1)+1, ..., \lambda(\alpha-1)\}\{\lambda(\alpha-1)+1, ...a\}$.
The λ first subsets of A have size $(\alpha-1)$ each and this is the maximum value possible (as sets of size α have no value). As already mentioned, there are at least $\lfloor \frac{a}{\alpha-1} \rfloor$ of these subsets. Thus the total value is at least $\lfloor \frac{a}{\alpha-1} \rfloor \times (\alpha-1)$ which is smaller than a because $(\alpha-1)$ is not a factor of a. The last subset has all the remaining elements and hence the maximum value possible (as there will never be α of them). The size (and hence value) of this last subset is $a - [\lambda(\alpha-1)] = a - [\lfloor \frac{a}{\alpha-1} \rfloor \times (\alpha-1)]$. This is equivalent to $a - [a - (\text{mod}(a, (\alpha-1)))]$ and this is always at least 1. □

Table 1. Percentage of the space searched by S99, IP, and our algorithm

a	Algorithms		
	S99	IP	Our
10	0.5×10^{-7}	$\approx 100\%$	4×10^{-6}
20	10^{-21}	$\approx 50\%$	10^{-15}
50	10^{-70}	–	10^{-40}
50			10^{-52}
100	10^{-170}	–	10^{-89}
100			10^{-107}

4 Evaluation of the Algorithm

In order to evaluate the proposed algorithm, we compare the percentage of the effectively searched space to the size of the whole space ($O(a^a)$). This is done using our algorithm, IP, and S99. The algorithm DJ04 could be used (instead of S99) but it would make very little difference since it also searches more than two levels of the CS graph. Please notice that both S99 and DJ04 just find a bound for $CS^\star$. However, they are general in the sense that they work for any CFG. IP also works for any CFG but the time taken to find $CS^\star$ is an issue if the number of agents is high. Contrarily to this, we are able to find $CS^\star$ (for games within the class of CFG defined before). This must be taken into account in the comparison. Due to complexity issues, experiments with IP are reported only up to 20 agents.

Table 1 shows the percentage of the space searched by each algorithm. Our algorithm performs a search in level l, where l depends on α. Therefore we show here examples with different values for α. When $a = 10$ we use $\alpha \in \{3, 4, 5\}$, all yielding $l = 5$ and hence a search space of $S(10, 5) = 4.2 \times 10^4$ that corresponds to $\approx 10^{-6}\,\%$ of the search space. Similarly, we use: $\alpha = 5$ ($l = 5$) when $a = 20$; $\alpha = 5$ (third line in table) and $\alpha = 16$ (fourth line) when $a = 50$; $\alpha \in \{6, 12\}$ (fifth line), and $\alpha = 20$ (sixth line) when $a = 100$.

It is clear that both our algorithm and S99 search only a very small portion of the space while IP searches much more because here we compare with the situation when their algorithm is run until it finds the optimum. Thus the first conclusion is a consequence of IP's complexity: their algorithm is the best for a small number of agents.

For a high number of agents, [2] searches less than our algorithm but finds a bound from optimum, while we find the optimum value (again, for a particular family of CFG). Moreover, the bound they found is $a = |A|$ (from optimum) which means that, for 10 agents the value found is only 10% of $V(CS^\star)$. For more agents the bound is much farther from $V(CS^\star)$. Therefore our algorithm is a good compromise between performance, generality, and number of agents.

Finally, Table 2 shows the running times for IP compared to our approach, for different number of agents. A machine running Linux Ubuntu 12.04 with processor Intel(R) Core(TM) i7-2600 3.40GHz and 16GB of RAM was used.

Table 2. Running time (milliseconds) for finding $CS^\star$

	Algorithms	
a	IP	Our
13	477	416 ($\alpha = 4$, $G = \emptyset$)
16	1,391,534	125,955 ($\alpha = 5$, $G = \{1, 2\}$)
20	28,820,231	2,328,965 ($\alpha = 4$, $G = \emptyset$)

5 Concluding Remarks

We presented an algorithm for finding the optimal coalition structure that works for a family of CFGs related to several games of practical interest. In particular, we have illustrated the example of DERs in smart grids. In these applications, coalitions must respect some constraints as for instance on the minimum and maximum number of agents, and/or value of agents that contribute more to the coalition.

The proposed algorithm is efficient, searching only a small portion of the space (a single level of the CS graph). The search is directed by domain knowledge in the sense that this knowledge is used in the CFG to valuate the coalitions according to restrictions they have (e.g., number of participants). We remark that, in spite of not being general, the CSG class proposed in this work is not restricted to the smart grids scenarios used here.

Two related issues will be tackled next: the modification necessary to accommodate more agents having different values, and how to distribute the payoff among them. Because the game is neither subadditive nor superadditive, the core is empty for the resulting coalitions. Thus other solution concepts must be used, such as the Shapley value.

Acknowledgments

This research was partially supported by CNPq.

References

1. Shehory, O., Kraus, S.: Methods for task allocation via agent coalition formation. Artificial Intelligence **101** (1998) 165–200
2. Sandholm, T., Larson, K., Andersson, M., Shehory, O., Tohmé, F.: Coalition structure generation with worst case guarantees. Artificial Intelligence **111** (1999) 209–238
3. Dang, V.D., Jennings, N.R.: Generating coalition structures with finite bound from the optimal guarantees. In: 3rd International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2004), New York, NY, IEEE Computer Society (2004) 564–571
4. Rahwan, T., Ramchurn, S.D., Dang, V.D., Jennings, N.R.: Near-optimal anytime coalition structure generation. In: Proc. of the Int. Joint Conf. on Art. Intelligence (IJCAI 07). (2007) 2365–2371 available at http://ijcai.org/proceedings07.php.

5. Rahwan, T., Jennings, N.R.: An improved dynamic programming algorithm for coalition structure generation. In: Proceedings of the Seventh International Conference on Autonomous Agents and Multiagent Systems (AAMAS'08), Estoril, Portugal (2008) 1417–1420
6. Rahwan, T., Michalak, T., Jennings, N.R.: Minimum search to establish worst-case guarantees in coalition structure generation. In Walsh, T., ed.: Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence. IJCAI'11, Barcelona, AAAI Press (2011) 338–343
7. Yeh, D.Y.: A dynamic programming approach to the complete set partitioning problem. BIT Numerical Mathematics **26** (1986) 467–474
8. U. S. Department of Energy: Grid 2030: A national vision for electricity's second 100 years (2003)
9. Ramchurn, S., Vytelingum, P., Rogers, A., Jennings, N.: Putting the "smarts" into the smart grid: A grand challenge for artificial intelligence. Communications of the ACM **55** (2012) 86–97
10. Rigas, E., Ramchurn, S., Bassiliades, N.: Managing electric vehicles in the smart grid using artificial intelligence: A survey. IEEE Transactions on Intelligent Transportation Systems (2015) 1–17
11. Chalkiadakis, G., Robu, V., Kota, R., Rogers, A., Jennings, N.R.: Cooperatives of distributed energy resources for efficient virtual power plants. In: Proceedings of 10th International Conference on Autonomous Agents and Multiagent Systems, Taipei, Taiwan (2011) 787–794
12. Vasirani, M., Kota, R., Cavalcante, R., Ossowski, S., Jennings, N.: An agent-based approach to virtual power plants of wind power generators and electric vehicles. Smart Grid, IEEE Transactions on **4** (2013) 1314–1322
13. Kamboj, S., Kempton, W., Decker, K.S.: Deploying power grid-integrated electric vehicles as a multi-agent system. In: Proceedings of 10th International Conference on Autonomous Agents and Multiagent Systems, Taipei, Taiwan (2011) 13–20
14. Mihailescu, R.C., Vasirani, M., Ossowski, S.: Dynamic coalition adaptation for efficient agent-based virtual power plants. In Klügl, F., Ossowski, S., eds.: Multiagent System Technologies. Volume 6973 of Lecture Notes in Computer Science. Springer Berlin Heidelberg (2011) 101–112
15. Ramos, G.de.O., Burguillo, J.C., Bazzan, A.L.C.: Self-adapting coalition formation among electric vehicles in smart grids. In: 2013 IEEE Seventh International Conference on Self-Adaptive and Self-Organizing Systems (SASO), Philadelphia, USA, IEEE (2013) 11–20
16. Ramos, G.de.O., Burguillo, J.C., Bazzan, A.L.C.: Dynamic constrained coalition formation among electric vehicles. Journal of the Brazilian Computer Society **20** (2014) 1–15
17. Pudjianto, D., Ramsay, C., Strbac, G.: Virtual power plant and system integration of distributed energy resources. Renewable Power Generation, IET **1** (2007) 10–16