Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

journal homepage: www.elsevier.com/locate/engappai

A hierarchical clustering approach to large-scale near-optimal coalition formation with quality guarantees

Alessandro Farinelli[a,*], Manuele Bicego[a], Filippo Bistaffa[a], Sarvapali D. Ramchurn[b]

[a] *Computer Science Department, University of Verona, Verona CAP 37134, Italy*
[b] *Department of Electronics and Computer Science, University of Southampton, Southampton SO17 1BJ, United Kingdom*

ARTICLE INFO

Keywords:
Coalition formation
Large scale systems
Collective energy purchasing
Ride-sharing
Hierarchical clustering

ABSTRACT

Coalition formation is a fundamental approach to multi-agent coordination, and a key challenge in this context is the coalition structure generation problem, where a set of agents must be partitioned into the best set of coalitions. This problem is NP-hard and typical optimal algorithms do not scale to more than 50 agents: efficient approximate solutions are therefore needed for hundreds or thousands of agents. In this paper we propose a novel heuristic, based on ideas and tools used in the data clustering domain. In particular, we present a coalition formation algorithm inspired by the well known class of hierarchical agglomerative clustering techniques (Linkage algorithms). We present different variants of the algorithm, which we call Coalition Linkage (C-Link) and demonstrate how such algorithm can be adapted to graph restricted coalition formation problems (where an interaction graph defined among the agents restricts the set of feasible coalitions). Moreover, we discuss how we can provide an upper bound on the value of the optimal coalition structure, and we show that for specific characteristic functions we can provide such bounds while maintaining polynomial computational costs and memory requirements. We empirically evaluate the different variants of the C-Link algorithm on two synthetic benchmark data-sets, as well as in two real world scenarios, involving a collective energy purchasing and a ride-sharing application. In these settings C-Link achieves promising results providing high quality solutions and solving problem involving thousands of agents in few minutes.

1. Introduction

The formation of collectives or coalitions is central to many practical applications that involve coordinating large numbers of agents as in emergency management scenarios (Ramchurn et al., 2010), manufacturing (Sabar et al., 2009), crowdsourcing (Peleteiro et al., 2015) and collective purchasing of goods or services (Vinyals et al., 2012). The coalition formation problem has been addressed from several perspectives: analysing how agents can form coalitions when operating in an environment where knowledge is distributed (Billard and Pasquale, 1995), designing algorithms to organize the agents into the most beneficial coalitions (i.e., Coalition Structure Generation or CSG) and distributing payoffs to agents so that formed coalitions are stable (Elkind et al., 2013; Saad et al., 2011). In this paper we focus on CSG, which involves partitioning the set of all agents so as to maximise the sum of the values (as given by a characteristic function) of the chosen coalitions.

To date, a number of approaches employed in this domain aim to solve the CSG problem optimally, ranging from Integer Programming to Branch-and-Bound techniques (Rahwan et al., 2009) through Dynamic Programming (DP) (Yun Yeh, 1986; Rahwan and Jennings, 2008b; Rahwan et al., 2012) (see Section 2 for more details). However, none of these solutions are scalable given that their complexity grows exponentially in the number of agents. Hence these optimal approaches can handle relatively small sets of agents (i.e., fewer than 30 Rahwan et al., 2009), excluding practical applications, such as, for example, collective energy purchasing or ride sharing, where thousands of agents might be involved.

A feasible alternative is to develop algorithms that scale to hundreds and thousands of agents at the expense of optimality. Along this line, previous approaches include Genetic Algorithms (Sen and Dutta, 2000), swarm intelligence (Dos Santos and Bazzan, 2012) and Meta-heuristics (Di Mauro et al., 2010), but these typically do not provide guarantees on convergence and solution quality for generic CSG problems and depend on several domain specific parameters to be tuned by the system designers.

Against this background, in this paper we propose a novel approach to CSG, that aims at providing near-optimal solutions along with

[*] Corresponding author.
E-mail addresses: alessandro.farinelli@univr.it (A. Farinelli), manuele.bicego@univr.it (M. Bicego), filippo.bistaffa@univr.it (F. Bistaffa), sdr1@soton.ac.uk (S.D. Ramchurn).

http://dx.doi.org/10.1016/j.engappai.2016.12.018
Received 30 December 2015; Received in revised form 28 September 2016; Accepted 20 December 2016
0952-1976/

quality guarantees (in the form of upper bounds on the optimal solution). The proposed approach is based on ideas and techniques coming from the data clustering research field, where the goal is to partition a data-set into groups (or clusters), based on the concept of similarity. Clustering approaches offer a wealth of solutions that have been developed and empirically validated in several practical applications involving large datasets (e.g., thousands of data points) (Theodoridis and Koutroumbas, 2008). Among the large number of clustering algorithms proposed in the past, here we concentrate on a specific class of algorithms called hierarchical agglomerative clustering schemes, and in particular on the Linkage algorithms (Ackerman et al., 2010). A linkage algorithm is a greedy strategy which starts from clusters formed by single data points and iteratively merges the "closest" pair of clusters – different choices of the cluster similarity criterion (called in some contexts the "linkage function" Ackerman et al., 2010) lead to different linkage algorithms. Our proposed approach exploits the idea of linkage algorithms for the CSG scenario: in particular, we tailor the linkage scheme to the CSG context by proposing a set of linkage functions that are based on the coalition characteristic function and specifically on the concept of *gain*.

In more detail, this work makes the following contributions to the state of the art. First, we propose a linkage approach (C-Link) that starts off from singleton coalitions and iteratively merges the most suitable pair of coalitions. The criterion used to evaluate whether coalitions should be merged is based on the concept of the gain that two coalitions would achieve if they merge. On this basis we devise different criteria, taking inspiration from standard linkage approaches used in the clustering domain (such as single-link, complete-link and average-link) but also pursuing new ideas, resulting in a novel criterion (Gain-Link) which takes advantage of the full characteristic function, significantly improving the quality of solutions. One key design concept for C-link is the simplicity of the proposed scheme which is based on one of the most basic clustering approaches. The use of a simple scheme allows us to analyse several properties of the approach. Specifically, we show that the C-Link approach is guaranteed to converge in N iterations at most (where N is the number of agents), and when using the Gain-Link criterion, it is anytime.[1] Moreover, we also provide a refinement of C-Link, able to solve the CSG problem when feasible coalitions are defined by an interaction graph[2] (Chalkiadakis et al., 2016; Voice et al., 2012, 2012).

Second, we show how we can provide guarantees on the quality of the solution for the CSG problem, by giving an upper bound on the value of the optimal coalition structure. Crucially, we can provide such guarantees while maintaining polynomial computational costs and memory requirements for a class of characteristic functions that can be expressed as a sum of a super-additive and a sub-additive components (called m+a functions). In recent work in CSG it has been shown that several characteristic functions have this m+a structure, and specifically the collective energy purchasing characteristic function we consider here (Bistaffa et al., 2014). Building on these results, we show that, for the collective energy purchasing domain, our approach can provide guarantees for a very large number of agents (more than 2700).

Third, we validate the C-Link approach in different data-sets: (i) two synthetic benchmarks, one employing uniform distribution for coalition values (Di Mauro et al., 2010) and the other with Normally Distributed Coalition Structures (NDCS) (Rahwan et al., 2009); (ii) a coalition formation problem where users can form groups to buy energy at discounted prices (Vinyals et al., 2012); and (iii) a ride-sharing scenario where commuters arrange one-time rides, forming groups to minimize transportation costs (Bistaffa et al., 2014). We compare the quality of solutions generated by the different variants of our approach against the optimal solution (computed using standard Integer Programming) when this is computationally feasible: C-Link is shown to provide high quality solutions. Moreover, we show that, in general, our approaches require much less memory and time to achieve good-enough solutions and therefore provide the first benchmarks for large-scale approximate CSG algorithms. Crucially, our C-Link approach can provide high quality solutions and it is able to solve problems involving thousands of agents (more than 2000) in few minutes (less than 4) for both the collective energy purchasing and ride-sharing scenarios.

2. Background and related work

In this section, we provide some background to our work including concepts from cooperative game theory, the Coalition Structure Generation problem, and data clustering techniques.

2.1. Characteristic function games

Cooperative Game theory provides abstract mathematical models to study scenarios where agents interact and cooperate (Chalkiadakis et al., 2011). In this framework, one of the most widely-studied model for cooperative games are *Characteristic Function Games* (CFG). A CFG is defined by a set of N agents $A = \{a_1, \ldots, a_N\}$ and a *characteristic function* v: $2^A \rightarrow \mathbb{R}$ which specifies a value (i.e., a real number) for each coalition $C \subseteq A$ of agents. An outcome for CFG is formed by two elements: (i) a partition of the agents into coalitions: (i.e., a coalition structure) and a payoff vector which distributes the value of each coalition among its members. In more detail, we represent with $\mathcal{CS}$ the set of all partitions of the set A, and we indicate with $CS \in \mathcal{CS}$ a specific coalition structure (i.e., $CS \subset 2^A$) where for any C_i, $C_j \in CS$, with $i \neq j$, $C_i \cap C_j = \varnothing$ (i.e., no agent is assigned to more than one coalition) and $\cup_{C \in CS} C = A$ (i.e., each agent is selected in at least one coalition). A payoff vector $X = \langle x_1, \ldots, x_N \rangle$ assigns a payoff to each agent. A key question within cooperative game theory is to identify outcomes that have desirable properties such as fairness (i.e., the payoff of an agent should reflect its contribution) and stability (i.e., agents should have incentives to stay in the coalition structure). There are several solution concepts that focus on such properties (such as the Shapley Value or the Core respectively) and relevant literature in this area typically focuses on devising a distribution of the payoffs that ensures such properties (Chalkiadakis et al., 2011). Since here we focus only on the CSG problem and not on the payoff distribution we will not consider such solution concepts in the rest of the paper.

In contrast, for the CSG problem the form of the characteristic function v is a key concept that significantly impacts on the classes of possible solution techniques. Notice that, in general there is no indication or restriction on how the characteristic functions v is defined. In fact in most related work such function is provided as a table that associates a value for each coalition. However, there are notable classes of CFG where v has specific properties, in particular a well known class of CFG is that of super-additive games where the characteristic function is super-additive: $v(\{C \cup S\}) \geq v(C) + v(S)$ for every pair of disjoint coalitions S, $C \subseteq A$. In super-additive games it is always beneficial for groups of agents to form larger coalition hence the grand coalition is the best coalition structure. Therefore, while this class of games has been widely studied as the basis for the construction of stable and fair payoff schemes, they are not is not interesting for the CSG problem as the optimal coalition structure is simply the grand coalition (Chalkiadakis et al., 2011). Another notable class of characteristic functions that has been recently considered are the m+a characteristic functions (Bistaffa et al., 2014). Given a coalition C an m

[1] Notice that anytime approaches for CSG problems are very valuable and several previous work focus on this aspect (Dang and Jennings, 2004; Rahwan and Jennings, 2008a; Rahwan et al., 2009).

[2] Notice that, the term synergy graph has also been used (for example in Voice et al., 2012; Bistaffa et al., 2014, 2015) to refer to a concept that is essentially equivalent to the interaction graph. Here we use the term interaction graph to be in line with most recent literature.

+*a* characteristic function can be expressed as the sum of a super-additive $v^+(C)$ and sub-additive component $v^-(C)$, [3] hence we have that $v(C) = v^+(C) + v^-(C)$. In contrast to super-additive characteristic functions, the best coalition structure for CSG with *m*+*a* functions is not always the grand coalition hence the CSG problem is not trivial.[4] However, branch and bound approaches for CSG can exploit the structure of *m*+*a* functions to provide efficient solution techniques with quality guarantees for large scale systems (thousands of agents).

Finally, in many scenarios the formation of some coalitions might be restricted by specific property of the environment, for example the communication infrastructures or social relationships might impose constraints on which coalitions can be formed. In particular, the model of graph restricted games proposed by Myerson (1977) focuses on the use of an interaction graph that captures relationships among the agents and restricts the feasible coalitions. Specifically, agents are the vertices of an undirected graph and edges represent peer to peer relationships, a coalition *C* is feasible if the vertices that corresponds to the agents form a connected subgraph of the interaction graph (i.e., the subgraph has a path connecting every pair of nodes). For example, consider a communication graph, where agents connected in the graph can communicate directly. In this case two agents that cannot communicate directly cannot form a coalition on their own but could be part of the same coalition if a third agent that can communicate with both of them is included. Another example is a network describing social relationships, where two agents are connected only if they know each other. Again, agents that do not know each other cannot form coalitions but can be part of the same coalition if they have a common friend. This model has been widely used both focusing on stability (Chalkiadakis et al., 2016) and on CSG (Voice et al., 2012, 2012; Bistaffa et al., 2014, 2015).

2.2. The coalition structure generation problem

Considering the notation introduced in Section 2.1 The optimal coalition structure generation problem involves finding the solution to the following maximization task:

$$\arg\max_{CS \in \mathcal{CS}} \sum_{C \in CS} v(C) \tag{1}$$

i.e., finding the coalition structure that maximizes the sum of the values of the coalitions. A key property of the characteristic function that we consider here is that the value it defines for one coalition is independent of the memberships of any other coalition selected in the coalition structure. Specifically, we assume no externalities. This property allows us to look at each coalition in isolation and therefore evaluate the benefits of merging one coalition with another using simple arithmetic operations. Given this, we can exploit this function in a similarity criterion that can be, in turn, used in large-scale data clustering algorithms (which we describe in Section 2.3). We next describe the exact and approximate approaches to solving the CSG problem and differentiate them from our work. For a more thorough analysis of such approaches, we point the reader to Chapter 8 in Elkind et al. (2013).

2.2.1. Exact approaches

The first solution to the CSG problem was based on a solution to the set partitioning problem using dynamic programming by Yun Yeh (1986). This complete algorithm grows in $O(3^N)$ but is not anytime. In turn, Sandholm et al. (1999) and Dang and Jennings (2004) showed how anytime solutions could be computed with quality guarantees. However, their solutions do not scale (growing in $O(N^N)$). More recently, in a series of papers on the subject, Rahwan and Jennings (2008a), Rahwan et al. (2009), Rahwan et al. (2012) were able to improve on DP by avoiding certain redundant operations (in the Improved-DP algorithm or IDP) and by supplementing such an approach with an anytime algorithm (IP) to solve the problem even more efficiently in the IDP-IP* algorithm (the current state of the art). They also showed how to distribute the solution on multiple cores, and how to solve CSG on the GPU (Michalak et al., 2010). These advances have made it possible to solve CSG *optimally*, *anytime*, on *multiple cores* and with quality guarantees. However, these solutions are limited to tens of agents (30 at most) due to their large memory requirements. In particular, a key drawback of the above approaches is that they need to hold all coalition values in memory ($O(2^N)$) during the search. We address this issue in our approach and show that our representation of the problem has minimal memory requirements as it needs to hold at most N^2 coalition values (beyond the input).

In an attempt to combat this complexity, several other paradigms that aim to restrict the number of feasible coalition structures have been proposed. For example, Ohta et al. (2009), unpack the coalition value function to define constraints on memberships of agents to coalitions and values associated with these constrained coalitions. Similarly, Rahwan et al. (2011) constrain the combination of agents that may exist in coalitions and provide a compact representation of the problem, proposing solution algorithms that outperform the approach proposed by Ohta et al. (2009), for small numbers of agents (no more than 30). Following the seminal paper by Myerson (1977) on graph restricted games, Voice et al. (2012) restrict feasible coalitions to be the connected sub-graph of an interaction graph that mirrors real-world social networks. By exploiting such sparse synergies, they provide novel (distributed) algorithms to compute coalition values and extend the IDP algorithm to generate the optimal coalition structure. While their approach is not anytime, they can optimally solve the CSG problem for about 50 agents in sparse networks. Along the same line, Voice et al. (2012) consider coalition formation over graphs, however in this work they focus on a specific class of valuation functions which are independent of disconnected members (IDM).

In this setting, authors provide complexity analysis, and algorithms that guarantee linear time bounds (when the treewidth of the graph is bounded). While this is an interesting result, the IDM property might not hold in several practical applications whenever adding an agent to a coalition has a cost that is typically not dependent on whether the agents are connected or not, but it is given by the specific application domain (Shehory and Kraus, 1998).

Along this line of research, Bistaffa et al. (2014, 2015) propose CFSS (Coalition Formation with Sparse Synergies), a branch and bound approach for coalition formation explicitly designed for interaction graphs. Authors show that, for *m*+*a* functions, CFSS can provide anytime solutions with quality guarantees for large scale systems (thousands of agents). Notice that, the *m*+*a* functions do not need in general to respect the IDM property. For example, the collective energy characteristic function considered in Bistaffa et al. (2014) (and that we use in our experiment) includes cost penalties for adding agents to the coalitions that are not dependent on whether the agents are connected or not, hence the IDM property is not valid in this case. The performance of CFSS is strongly dependent on the structure of the interaction graph, and on the possibility to efficiently provide tight bounds for the characteristic function. In Section 5 we compare the quality of solution of our approach with CFSS in the ride-sharing scenario (a characteristic function for which Bistaffa et al. devised a specific bounding technique) showing that CFSS is superior when the number of agents is very large (i.e., from 1500 to 2000) and when a social network constrains possible coalitions. However, our approach provides significantly better results when such social networks are not present.

Recently, Wang and Jiang investigated the use of communities to regulate cooperation of agents (Wang and Jiang, 2014). Specifically,

[3] Similar to the super-additive characteristic function for sub-additive characteristic functions we have that $v(\{C \cup S\}) \leq v(C) + v(S)$ for every pair of disjoint coalitions $S, C \subseteq A$.

[4] While investigating the tractability of the CSG problem when using m+a functions is definitely an interesting topic, it falls outside the scope of the current contribution.

they propose a model for task assignment where agents can form groups to cooperate only with members of their communities. Their model is relevant to our work as we also consider relationships among agents (expressed as interaction graphs) to restrict possible coalitions. However, we explicitly focus on the CSG problem, while they propose a model for task assignment and do not provide an approach to partition the agents into coalitions.

Finally, in a recent paper, Chalkiadakis et al. (2016) investigate the complexity for computational tasks associated with the core solution concept. While this work focuses on providing stable outcome for characteristic function games (rather than on CSG as we do in this work), their work is relevant as they consider non super-additive settings and focus on interaction graphs considering different graph topologies. Specifically, they show interesting tractability results for specific topologies (such as lines, trees, cycles, bounded tree-width graphs and graphs with bounded degree). While this is definitely an important contribution to derive tractable algorithms for the CSG problem, in this work we take a different perspective and aim at designing efficient algorithms that provide near-optimal solutions in general settings, i.e., when the CSG may not restricted by an interaction graph or when such interaction graph is similar to a social network.

2.2.2. Heuristics and approximate solutions

Very few heuristic solutions to the CSG problem exist. For example, Sen and Dutta (2000) propose a solution based on genetic algorithms while Dos Santos and Bazzan (2012) propose an approach based on swarm intelligence (the bee clustering algorithm) for task allocation in the RoboCup Rescue domain. Meta-heuristic approaches to CSG have also been investigated, for example Di Mauro et al. (2010) use a stochastic local search approach (GRASP) to iteratively build a coalition structure of high quality. Similar to what our approach does, such GRASP-based method tries to build a coalition structure of good quality by merging coalitions. However, in contrast to C-Link, this GRASP-based approach does not consider the comparative benefit of merging two coalitions (against not merging), but it only considers the value of the best merge of two coalitions. Moreover, such approach considers other operations beyond coalition merging (such as, for example, splitting a coalition in two and exchanging a pair of agents between two coalitions) and it performs randomized search. While this helps the algorithm to escape from local minima, it prevents the algorithm from providing guarantees on performance (e.g., it is not anytime). In general, all previous heuristic approaches are typically dependent on several domain specific parameters and, most importantly, they are not able to provide guarantees on the solution quality.

More recently, Service and Adams (2010, 2011) presented new approaches to approximating the CSG problem using the DP algorithm to find optimal solutions for parts of the search space. Thus, they are able to guarantee constant factor approximations within $O(3^N)$ ($\frac{2}{3}$ in $O(\sqrt{N}2.587^N)$ and $\frac{1}{4}$ in $O(N^2)$). Their approach is shown to find high quality solutions in reasonable time for up to 27 agents. However, their algorithm requires similar worst case memory requirements to DP (i.e., $O(3^N)$). Moreover, their runtimes are distribution-dependent. In contrast, our proposed approach requires significantly lower memory ($O(N^2)$) beyond the input and its run-time does not depend on the distribution of coalition values. In fact, our algorithm is guaranteed to complete in $O(N)$. The only trade-off is that the quality guarantees are distribution dependent but, as we show in our evaluation, these guarantees are, in particular settings, of high quality.

Note that an initial version of this paper appeared in Farinelli et al. (2013). However, here, we provide more results on the performance of our algorithms on benchmark distributions and on the ride-sharing scenario. Moreover, we now present analytical results that prove the performance guarantees of the C-Link approach. Finally, we show how the algorithm can be adapted to work on interaction graphs maintaining its performance guarantees.

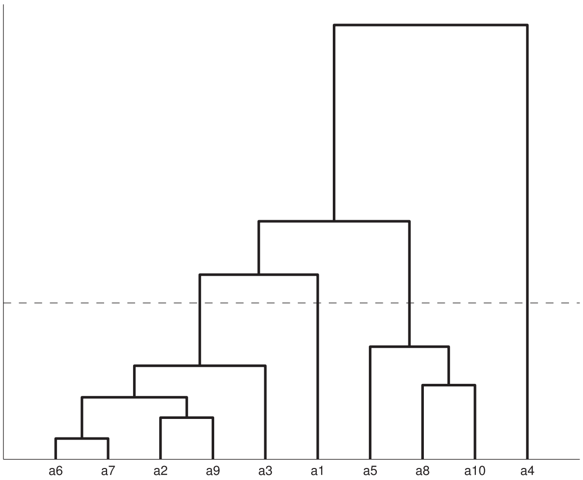


Fig. 1. An exemplar dendrogram representing a possible hierarchical clustering process for 10 agents. The horizontal dashed line represents a cut of the dendrogram and defines the coalition structure $\{\{a_6, a_7, a_2, a_9, a_3\}, \{a_1\}, \{a_5, a_8, a_{10}\}, \{a_4\}\}$.

2.3. Data clustering

The huge amount of data clustering approaches proposed in the past can be broadly divided in two main families (Theodoridis and Koutroumbas, 2008): partitional clustering and hierarchical clustering. In the former class, the typical output is a partition of the original dataset, whereas in hierarchical clustering data is arranged in layers of partitions, where each partition is merged in a partition of the subsequent layer. Hence, a hierarchical clustering approach can be conveniently represented by a *dendrogram*, a tree structure that consists of layers of nodes, each representing a cluster, where lines connect clusters that are merged in the next layer (see Fig. 1). An important class of hierarchical clustering approaches is represented by the linkage algorithms (Ackerman et al., 2010) (such as single/complete/average-link), a class of agglomerative approaches which start from clusters formed of single data points and iteratively merge the closest pair of clusters; what distinguishes different linkage-based algorithms is the definition of the similarity between clusters, which is used to determine the clusters to be merged.

Here we decided to adopt hierarchical agglomerative clustering, and in particular linkage algorithms, for two key reasons: (i) the most widely used partitional clustering approaches (e.g., the popular k-means clustering) are dependent on several system parameters (e.g., the number of groups to be formed) and on the choice of the initial solution, whereas the behaviour of linkage approaches does not depend on that; (ii) even if partitional clustering techniques are more efficient in terms of memory and computation (deriving a single partition of the dataset), we consider this efficiency not to be crucial here, as the number of agents that our approach can handle is already significantly beyond the capability of all current coalition formation approaches. Finally, notice that, in standard linkage approaches, the merging process always results in a single cluster containing all the elements of the initial data-set. Hence to obtain the best suited data partition, the system designer must choose when to stop the merging process, i.e., at which level the dendrogram should be cut. This is typically a complex, domain dependent problem, for which no general and widely accepted solution exists, especially because of the inherent vagueness in the definition of a cluster, and the difficulty in defining an appropriate similarity measure and objective function (Von Luxburg et al., 2012). Nevertheless, in the approach proposed in this paper, a natural and meaningful stopping criterion for the cluster merging process can be automatically derived, as we will detail in Section 3.3.

3. The coalition link approach

The coalition link (C-Link) algorithm starts and develops over the linkage approach for clustering. Instead of data points (as discussed in Section 2), as in standard linkage algorithms, here we consider coalitions C (including those of single agents) as the entities to be assembled into coalition structures CS. Hence, starting from a set of agents, the algorithm aims to produce a sequence of *nested* coalition structures or partitions $CS^0, CS^1, \ldots, CS^L$. A partition CS^i is nested into a partition CS^j if every component of CS^i is a subset of a component of CS^j. Given a partition at level i, the algorithm evaluates each new partition to be formed by determining the value of all the possible *merge* operations (i.e., the union of two coalitions into one); the level i +1 is then formed by merging the most *suitable* coalitions. In the clustering scenario, this value is typically defined by the so called linkage function: the choice of this function leads to the different versions of the linkage algorithm (as described in Section 2). In the next section we provide a description of possible linkage functions (involving different computational costs) that can be used in a linkage algorithm to solve the CSG problem. Building upon this, we then describe the C-Link algorithm.

3.1. Linkage functions for CSG

In this section we introduce four different linkage functions: three of them are based on concepts and ideas peculiar of the well known Single Link, Complete Link and Average Link variants of the clustering Linkage algorithm, whereas the fourth fully exploits the particular Coalition Formation context. In any case, all of them are based on the concept of Gain, which can be defined for any pair of coalitions C_i, C_j as:

$$G(C_i, C_j) = v(\{C_i \cup C_j\}) - v(C_i) - v(C_j) \tag{2}$$

where $v(C)$ is value of the characteristic function v for coalition C. In other words, this gain function captures the value of synergies between coalitions and is computed in constant time. Please note that, if the coalition value function is strictly *super-additive*, the gain is always positive, while if the coalition value function is strictly *sub-additive* (i.e., $v(\{C \cup S\}) < v(C) + v(S)$), the gain is always negative.

A first straightforward possibility for the linkage function is to directly exploit and adapt the well known Single Link, Complete Link and Average Link variants, which define the value of a merged pair of clusters (namely the closeness) by analysing the pairing of individual data points belonging to the two clusters. In particular, the closeness of two clusters is measured as the closeness of the nearest pair of objects (for Single Link), the closeness of the farthest pair (for Complete Link), and the averaged closeness of all pairs of objects (for the Average Link). Here we translate these concepts to the coalition formation case: this means defining the value of merging coalitions based on the value of pairing agents together. In more detail, we defined the following linkage functions for our algorithm, calling them single-link (*SL*), complete-link (*CL*), and average-link (*AL*), respectively:

$$lf_{SL}(C_i, C_j) = \max_{a_h \in C_i, a_l \in C_j} (G(\{a_h\}, \{a_l\})) \tag{3}$$

$$lf_{CL}(C_i, C_j) = \min_{a_h \in C_i, a_l \in C_j} (G(\{a_h\}, \{a_l\})) \tag{4}$$

$$lf_{AL}(C_i, C_j) = \frac{1}{|C_i|\cdot|C_j|} \sum_{a_h \in C_i, a_l \in C_j} (G(\{a_h\}, \{a_l\})) \tag{5}$$

where $G(\{a_h\}, \{a_l\}) = v(\{a_h, a_l\}) - v(\{a_h\}) - v(\{a_l\})$, and represents the Gain obtained while joining together two agents a_h and a_l. Thus, in the context of coalition formation $lf_{SL}(\cdot)$ defines the value of a merge as the value of the strongest pairing of agents from two coalitions, $lf_{CL}(\cdot)$ defines this value as the value of the weakest pairing between two agents, and, finally $lf_{AL}(\cdot)$ defines the average value of pairing all possible combinations of agents from two coalitions. All these functions can be computed in $O(N^2)$ operations[5] as they only consider pairwise interactions between the agents (e.g., coalitions of size 2). Consequently, to use the above linkage functions, we only need to define a coalition value function for coalitions of size 1 and size 2 instead of 2^N coalitions. While this represents a significant reduction in the computational and practical task of defining coalition values, such definitions are likely to lead to poor performance if larger coalitions tend to be most valuable.

In order to capture gains from merges of coalitions rather than pairings of agents, we define a new linkage function called the *Gain-Link* (GL), that captures the synergy between coalitions using Equation (2):

$$lf_{GL}(C_i, C_j) = G(C_i, C_j) \tag{6}$$

Notice that, the definition of the linkage function on the basis of the gain (both considering only pairs of agents or general coalitions) naturally provides an automatic stopping criterion for the C-Link algorithm: the algorithm stops if there is no advantage in joining together the “best” pair of coalitions, i.e., if the linkage function evaluated on the best pair has a negative or zero value (i.e., $lf(C_i, C_j) \le 0$). As explained in the previous section, this is a crucial difference between the linkage variants used in the clustering scenario and our approach, as classical schemes always produce a full dendrogram: deciding where to place a cut to obtain the “best” clustering is a key, domain dependent issue (see Von Luxburg et al., 2012; Theodoridis and Koutroumbas, 2008 Chapter 13.6).

3.2. The C-Link algorithm

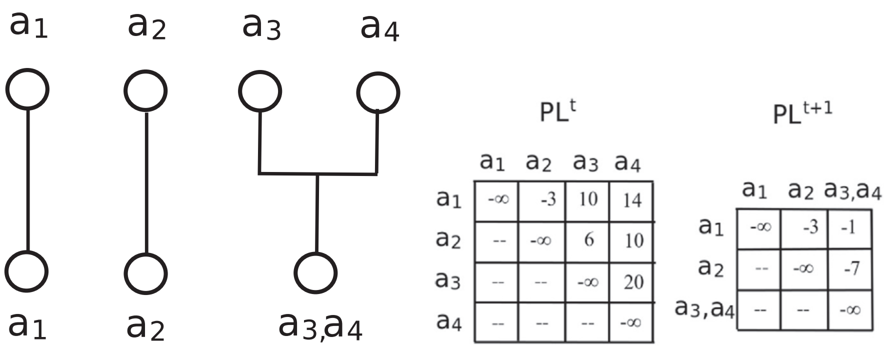
The C-Link approach is described in Algorithm 1. Essentially, the algorithm is based on the definition of the linkage function $lf(C_i, C_j)$ that indicates how convenient it is for two coalitions $C_i, C_j \in CS$ to be merged. The approach iteratively updates the Partition Linkage matrix $PL^{(t)}$, which stores the value $lf(C_i, C_j)$ in the entry (i,j) (13–17).

In more detail, the approach starts from the completely disjoint case: a partition where every coalition is composed of a single agent (line 1), and initializes the *PL* matrix with the linkage value of agent pairs (lines 2–5). Then, at every iteration, we compute the most suitable pair of coalitions (see lines 6, 7 and 18, 19); if merging a coalition pair is convenient for the system (line 9), namely if there is a pair of coalitions for which the linkage function is positive, such coalitions are removed from the current partition and replaced with their union, in order to define the next level of the hierarchy (lines 11–13). Otherwise the algorithm stops, and the current partition is returned.

Algorithm 1. C-Link algorithm.

```
Input: $A$: the set of agents, $lf(\cdot)$: the linkage function.
Output: $CS_{opt}$ the optimal partition of $A$
      // Initialize partitions to singletons
1:    $CS^{(0)} = \{\{a_1\}, \{a_2\}, \ldots, \{a_N\}\}$
      // Initialize PL for each agent pair
2:    for i,j=1 to N, $i \neq j$ do
3:        $PL^{(0)}(i, j) = lf(\{a_i\}, \{a_j\})$
4:    end for
      // Initialize self linkage to $-\infty$
5:    $\forall\ i,\ PL^{(0)}(i, i) = -\infty$
      // Compute and store the best indices and best linkage value
6:    $\hat{\imath}, \hat{\jmath} = \mathrm{argmax}_{i,j} PL^{(0)}(i, j)$
7:    $\widehat{pa} = \max_{i,j} PL^{(0)}(i, j)$
```

[5] Where we assume that computing or reading the value of a coalition is one operation



PL^t

	a_1	a_2	a_3	a_4
a_1	-∞	-3	10	14
a_2	--	-∞	6	10
a_3	--	--	-∞	20
a_4	--	--	--	-∞

PL^{t+1}

	a_1	a_2	a_3,a_4
a_1	-∞	-3	-1
a_2	--	-∞	-7
a_3,a_4	--	--	-∞

(a) (b)

Fig. 2. Figures (a) and (b) describe an artificial example where the optimal coalition structure is $\{\{a_1\}, \{a_2\}, \{a_3, a_4\}\}$ and show one update step for the Gain-Link method: (a) shows the dendogram and (b) shows how the *PL* matrix evolves.

```
8:  t=0
    // Main Loop: stop if best linkage value is negative or the grand coalition was formed
9:  while ($\widehat{pa} \geq 0$) AND ($|CS^{(t)}| > 1$) do
10:     t=t + 1;
        // Update Partition: remove the two coalitions that should be merged and add the merged coalition
11:     define $C_{\widehat{ij}} = C_{\hat{\imath}} \cup C_{\hat{\jmath}}$
12:     $CS^{(t)} = CS^{(t-1)} \setminus \{C_{\hat{\imath}}\} \setminus \{C_{\hat{\jmath}}\} \cup \{C_{\widehat{ij}}\}$
        // Update $PL^{(t)}$
13:     Delete rows and columns of $PL^{(t)}$ relative to $C_{\hat{\imath}}$ and $C_{\hat{\jmath}}$, add one row and one column for $C_{\widehat{ij}}$.
        // Compute linkage value for each coalition $C_k$ with the newly formed coalition $C_{\widehat{ij}}$
14:     for $C_k \in CS^{(t)}$, $C_k \neq C_{\widehat{ij}}$ do
15:         $PL^{(t)}(\widehat{ij}, k) = PL^{(t)}(k, \widehat{ij}) = lf(C_{\widehat{ij}}, C_k)$
16:     end for
        // Set self linkage value to $-\infty$
17:     $PL^{(t)}(\widehat{ij}, \widehat{ij}) = -\infty$
        // Update best indices and best value of linkage
18:     $\hat{\imath}, \hat{\jmath} = \text{argmax}_{i,j} PL^{(t)}(i, j)$
19: $\widehat{pa} = \max_{i,j} PL^{(t)}(i, j)$
20: end while
21: return $CS^{(t)}$
```

Fig. 2 shows an exemplar matrix update step for our C-Link approach (using GL). In particular, Fig. 2(a) shows the dendrogram and Fig. 2(b) the update of the *PL* matrix. Here we assume the optimal coalition structure is $\{\{a_1\}, \{a_2\}, \{a_3, a_4\}\}$. At every stage, our approach evaluates all possible merge operations: in other words, in the situation shown, it evaluates all the possible coalitions of size two, computing the values reported in the left-hand side matrix in Fig. 2(b). Now, the best option is to form the coalition $\{a_3, a_4\}$ and hence the algorithm updates the *PL* matrix as shown in the right-hand side of Fig. 2(b). Notice that in this new matrix all elements are negative and hence the algorithm would stop processing in the next iteration.

3.3. C-Link analysis and discussion

The main properties of C-Link are the following:

Property 1 (*Convergence***).** *C-Link always converges in at most N steps, where N is the number of agents.*

This is because C-Link reduces by one the number of elements in *CS* at each iteration (see Algorithm 1 line 12) and stops if the grand coalition forms or if the best linkage value is non-positive (see Algorithm 1 line 9).

Property 2 (*Optimality***).** *C-Link always returns the grand coalition for super additive functions (i.e., it gives an optimal solution).*

If the characteristic function is super-additive, the gain (as defined in Equation (2)) cannot be negative. Hence, the entries of $PL^{(t)}$ for all the linkage functions defined in previous section and for all t are all non-negative. Consequently, the approach stops only when the grand coalition is formed (i.e., $|CS^{(N)}| = 1$).

Notice that, as stated in Section 2.1 the CSG problem in super-additive games is not interesting as in this context the grand-coalition is always the optimal solution. However, it is interesting to consider this property because in general an heuristic approach is not guaranteed to be optimal for super-additive games.

Property 3 (*Anytime***).** *The Gain-Link variant of C-Link is anytime*

C-Link performs at most one merge at each step, and with the Gain-Link function it performs the merge only if the *gain* (as defined in 2) is positive. Hence, if we consider a partition $CS^{(t)}$, the gain link algorithm will either stop because no couple of coalitions in C^k provides a positive gain, or produce a new partition $CS^{(t+1)} = CS^{(t)} \setminus \{C_{\hat{\imath}}\} \setminus \{C_{\hat{\jmath}}\} \cup \{C_{\widehat{ij}}\}$ (see Algorithm 1 line 12). Consequently, if a new partition is formed then we have that $v(C_{\widehat{ij}}) - v(C_{\hat{\imath}}) - v(C_{\hat{\jmath}}) \geq 0$, and since the value of the new partition can be written as $V(CS^{(t+1)}) = V(CS^{(t)}) - v(C_{\hat{\imath}}) - v(C_{\hat{\jmath}}) + v(C_{\widehat{ij}})$ we have that $V(CS^{(t+1)}) \geq V(CS^{(t)})$, where $V(CS) = \sum_{C \in CS} v(C)$.

Notice that this property does not hold, in general settings, for the other approaches (i.e., single/complete/average-link), as those approaches use, as linkage function, an estimation of the gain based only on pairwise relations.

In terms of computational complexity, following the analysis for Generalised Agglomerative Schemes for clustering reported in Theodoridis and Koutroumbas (2008) we can show that the C-link approach requires in the worst case $O(N^3)$ operations. As for memory requirements, C-Link stores the *PL* matrix which has N^2 entries. Hence, the space complexity of C-Link is $O(N^2)$ beyond the input.[6] This contrasts sharply against typical optimal approaches (as discussed in Section 2) that grow either exponentially in $O(N^N)$ (e.g., IP) or in $O(3^N)$ (e.g., IDP or IDP-IP*).

To further understand the behaviour of the C-Link approach it is useful to compare its execution to the operations performed by the Dynamic Programming approach on the coalition structure graph (see Fig. 4 for an example) (Rahwan and Jennings, 2008a). The DP approach first evaluates every possible movement on the graph (i.e., every possible split for coalitions of every size), then it starts from the

[6] If we also consider the memory required to specify the characteristic function, which is the input to the CSG problem, and if such characteristic function is specified in a tabular form (i.e., we store one value for each possible coalition as in Section 5.2), the memory storage is exponential in the number of agents ($O(2^N)$). However, this is not related to the operations of the algorithm, which stores only the *PL* matrix.

bottom node (i.e., the grand coalition) and moves upwards until an optimal node is reached (i.e., a node from which no splitting is beneficial). In comparison with DP, C-Link is essentially a myopic version that progresses top down (i.e., from singleton coalitions towards the grand coalition). In fact, at each level the C-link approach only evaluates possible merges of coalition pairs and once a coalition is formed it will never be split, hence the approach can be trapped in local maxima of the objective function. Nevertheless, the results we discuss in Section 5 show that the performance of the approach is extremely promising.

3.4. Applying C-Link to interaction graphs

As mentioned in the introduction, recent approaches for coalition structure generation such as Voice et al. (2012), focus on interaction graphs where sparse synergies among the agents restricts the *feasible* coalitions. These types of constraints are very common in practical settings such as, for example, applications involving users that are connected by a social network, hence it is crucial for a coalition structure generation method to be able to operate considering such constraints. Consequently, here we present an alternative version of our C-Link algorithm that is able to operate in such settings. More specifically, we present the C-Link algorithm for interaction graphs and then an analysis of the C-Link algorithm when operating in such settings.

In what follows, we assume that the interaction graph defining the feasible coalitions is connected, i.e., there exists a path between any two agents in the interaction graph. This has the important consequence that the grand coalition is always part of the feasible coalitions, which is crucial for some of the properties of the C-Link approach (as detailed below). Notice that, if the graph is not connected we can isolate the connected components of the graph and run C-Link on each of such components independently. This is because, coalitions that belong to different connected components will always be disjoint. Moreover, finding the connected components of a graph is computationally easy (i.e., polynomial in the number of vertices Cormen et al., 2001), hence we can easily decompose the problem in separate sub-problems, one for each connected component of the interaction graph.

Now, the main idea to adapt the C-Link approach for interaction graphs is to consider the edges of the graph when deciding which pair of coalitions to join. In particular, following Voice et al. (2012), we represent the interaction graph with an undirected graph $G = (V, E)$ where the vertices represent the agents, and edges correspond to connections in the interaction graph. Given this representation, feasible coalitions correspond to the set of connected subgraphs (i.e., subgraphs for which there is a path between any couple of nodes) as originally proposed in Myerson (1977).

To ensure correctness of our C-Link procedure on interaction graphs we must ensure that our approach will never produce a coalition which is not feasible, i.e., a coalition whose members do not form a connected sub-graph of the original interaction graph. To this end, at each step of the C-Link algorithm we maintain a graph that represents the connectivity for the current coalition structure. In particular, given a coalition structure $CS = \{C_1, \ldots, C_k\}$ we have the graph $net_{CS} = (V_{CS}, E_{CS})$, where V_{CS} is a set of vertices, with one vertex for each coalition $C_i \in CS$. Two coalitions C_i and C_j are connected by an edge if, and only if, in the original interaction graph the members of such coalitions are connected. Such graph can be easily built and maintained by initialising $net_{CS^{(0)}}$ with the interaction graph G, and then updating such structure by performing an edge contraction operation on the edge that connects the two coalitions that we decide to join. For a pseudo-code description of the C-Link algorithm on interaction graphs see the supplementary material.

Notice that the analysis for Algorithm 1 reported in Section 3.3 remains valid. Specifically, Property 2 (i.e., optimality for super-additive functions) is still valid because, as mentioned above, we assume that the interaction graph is connected. Consequently the grand coalition is always part of the set of feasible coalitions and the C-Link approach will always be able to find this coalition.

Moreover, the worst case complexity (in terms of memory and computation) of the algorithm does not change. In fact, while the C-Link algorithm applied on interaction graphs requires additional memory and computation to store and update the net_{CS} structure, the net_{CS} matrix must only store a polynomial number of Boolean values in the number of agents and the algorithm must perform a polynomial number of operations to update such structure. Nonetheless, the C-link algorithm applied on interaction graphs does not need to compute the linkage function for coalitions that are not feasible. The most important benefit of such algorithm is the ability to directly encode the constraints on coalitions defined by the interaction graph without holding all feasible coalitions in memory. This renders the algorithm particularly suitable for large-scale applications.

Notice that, C-Link does not analyze the topology of the graph. On the one hand this prevents the approach to exploit known properties for specific topologies: for example, it has been shown Chalkiadakis et al. (2016) that the CSG problem is tractable over lines (for arbitrary characteristic functions), however the C-Link approach would not exploit this property and as such it might return a sub-optimal coalition structure even when the problem is tractable. On the other hand the C-Link algorithm was not designed for this specific cases but rather to provide near-optimal solutions in general settings, e.g., for interaction graphs defined by social networks, where this specific type of topologies are unlikely to appear.

4. Bounds on solution quality

In this section, we show that we can provide an upper bound for the value of the optimal coalition structure for a given instance of our CSG problem. In more detail, following the basic ideas of the C-Link approach, our guarantees on solution quality are based on the concept of *gain* for a characteristic function and on an upper bound on the *maximum gain*. Finding such an upper bound allows us to provide guarantees on the solution quality that an heuristic approach can provide because we can bound the distance between the quality of the solution provided and the value of the unknown optimal solution (see Section 5.3.1).

In more detail, given a specific instance of the CSG problem I and a solution $\widetilde{CS}(I)$ returned by any heuristic approach on such instance, we can consider the *Performance Ratio* (PR) (Ausiello et al., 2012), a standard measure to evaluate the quality guarantees of algorithms, defined as the ratio between the provided solution and the optimal one on the given instance. As computing the optimal solution for large instances of the CSG is not possible, we define the *Maximum Performance Ratio* (MPR) as the ratio between the heuristic solution and the upper bound on the optimal solution $UB(I)$: $MPR(I) = \max\left(\frac{V(\widetilde{CS}(I))}{UB(I)}, \frac{UB(I)}{V(\widetilde{CS}(I))}\right)$. The MPR provides a significant quality guarantee on the approximate solution $\widetilde{CS}(I)$, since $V(\widetilde{CS})(I)$ cannot be worse than by a factor of $MPR(I)$ w.r.t. the optimal solution.

While for general characteristic functions the computation of an upper bound on the value of the optimal coalition structure is not tractable[7] in this paper we provide a tractable procedure to compute such upper bound for the $m+a$ characteristic functions.

4.1. Upper-bound for the optimal coalition structure

We start by detailing the concept of *gain* for a coalition structure,

[7] Following the results in Sandholm et al. (1999), to provide any bound for a general characteristic function one should at least visit all coalitions, hence the required computational effort would have an exponential element in the number of agents.

on the basis of the hierarchy that the C-link algorithm builds (i.e., the dendogram) and of the gain the system achieves when merging two coalitions.

In more detail, consider a set of agents A and a partition CS of this set with $|CS| = K$ (and $K < |A|$), we indicate with $\mathcal{S}$ the coalition structure formed by the singletons (i.e., $|\mathcal{S}| = |A|$). We can always build a hierarchy of coalition structures (i.e., a dendogram) such that, by repeatedly joining two coalitions, we obtain CS from $\mathcal{S}$. That is, we can build a sequence $H(CS) = \langle L_{|A|} = \mathcal{S}, L_{|A|-1}, \ldots, L_K = CS \rangle$ such that, $\forall\, k \in [K, \ldots, |A| - 1]$:

$$L_k = L_{k+1} \setminus \{C_i\} \setminus \{C_j\} \cup \{C_i \cup C_j\}$$

where $\forall\, k$ L_k is a partition of A and $C_i, C_j \in L_{k+1}$.

Notice that the index of each level k encodes the number of coalitions that can be part of the coalition structure in that level, and as the hierarchy goes from the singletons towards the grand coalition, k decreases.

Moreover, observe that, for any coalition structure CS we can always build a hierarchy $H(CS)$ that connects the singletons with CS (Algorithm 3 reported in the supplementary material provides a procedure to build $H(CS)$).

For example, Fig. 3 shows one possible hierarchy for the coalition structure $CS = \{\{1, 2\}, \{3, 4\}\}$.

We now define the gain for a given coalition structure CS as the sum of the worth of each coalition minus the sum of the worth of the singletons:

Definition 1 (*Gain of a coalition structure***).**

$$G(CS) = \underbrace{\sum_{C \in CS} v(C)}_{V(CS)} - \underbrace{\sum_{a \in A} v(a)}_{V(\mathcal{S})}$$

We can show that the gain of a coalition structure can be decomposed into the sum of the gains for each level L_k along the hierarchy previously defined. In more detail, we show that the following theorem holds:

Theorem 1 (*Gain sum decomposition*). *Given a CS with* $|CS| = K$ ($K \le |A|$), *and given a specific hierarchy H(CS) we have that*:

$$G(CS) = \sum_{k=K}^{|A|-1} G^{k+1,k} = V(CS) - V(\mathcal{S}) \tag{7}$$

where $G^{k+1,k} = G(C_i, C_j)$ *and* C_i,C_j *are the coalitions merged going from level k+1 to level k of hierarchy H(CS).*

Proof. See the supplementary material.

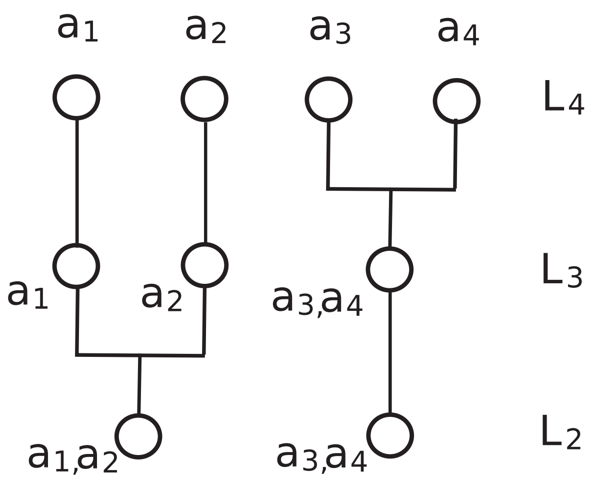


Fig. 3. An exemplar hierarchy for the coalition structure $CS = \{\{1, 2\}, \{3, 4\}\}$.

Next, we want to use this gain-sum decomposition to bound the value of the found coalition structure. Before that, we consider the following definitions to characterise the characteristic function in terms of gain:

Definition 2 (*Maximum gain for a coalition***).**

$$G_C = \max\left\{0, \max_{S \subset C} G(C \setminus S, S)\right\} \tag{8}$$

where $G(\cdot)$ is the gain as defined in Eq. (2). Intuitively, Eq. (8) expresses the maximum gain for all possible splits of a given coalition C. For example, to compute the maximum gain for coalition $C = \{1, 2, 3\}$ we must find the maximum of all the following possible splits of coalition $\{1, 2, 3\}$: $\{1\}\{2, 3\}$, $\{2\}\{1, 3\}$, $\{3\}\{1, 2\}$.

Based on this concept, we can define the maximum gain for a characteristic function as:

Definition 3 (*Maximum gain for a characteristic function***).**

$$G_M = \max_{C \subseteq A} G_C \tag{9}$$

Finally, using the above definition of gain and exploiting Theorem 1, we can provide an upper bound on the value of the optimal coalition structure (that we indicate with CS^*). Specifically, based on Theorem 1, it is easy to see that the following lemma holds:

Lemma 1 (*Maximum Gain Bound (*use $\mathrm{G}_{\mathbf{M}}$*)***).**

$$V(CS^*) \le V(\mathcal{S}) + (|A| - 1) G_M$$

Proof. Intuitively, the lemma holds because the hierarchy of coalitions starting from singletons can never have more than $(|A| - 1)$ levels, hence if we sum to the value of the singletons (i.e., the value at the bottom of the hierarchy) an upper bound on the maximum gain obtained by merging any two coalitions we clearly obtain an upper bound on the value of the optimal coalition. See the supplementary material, for a more detailed proof.□

A tighter bound can be obtained by exploiting the concepts of gain for a characteristic function and the hierarchy built by C-Link. In particular, we can decompose the gain along the hierarchy, deriving an upper bound for the gain at each level of the hierarchy. Aggregating these upper bounds for all the possible levels of the hierarchy we can then provide a tighter upper bound on the optimal solution. The interested reader can find further details on this upper bound, together with an empirical evaluation, in the supplementary material (see Section 7).

4.2. Estimating the maximum gain for m+a characteristic functions

The above analysis provides a bound for the optimal solution which is based on the computation of the maximum gain for the characteristic function (or its decomposition through the levels of the hierarchy). However, for characteristic functions that do not exhibit any particular structure (such as the uniform or the NDCS functions we use in Section 5), computing the maximum gain for all possible splittings of a coalition (i.e., Definition 2) requires searching through all the subsets of such a coalition. This results in a prohibitive computational effort.[8] The computation of the refined bound (i.e. that based on hierarchical decomposition of the gain) is less demanding, but it is still intractable as it requires to iterate through all possible coalitions. Therefore, such bound can not be computed for large scale systems.

In what follows, we focus on quality guarantees that can not be

[8] One way to implement this is to use dynamic programming to compute all possible splittings for all sizes of coalitions. This would result in an algorithm that has the same computational complexity of DP ($O(3^n)$) and that could return the optimal solution to the problem, thus defeating the purpose of an upper bound.

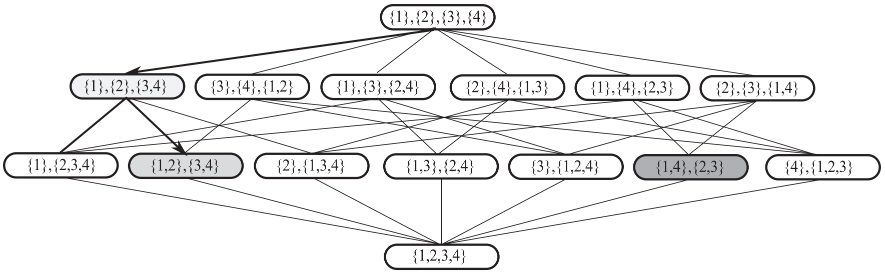


Fig. 4. A diagram of the coalition structure graph for 4 agents. The downwards arrows show the path followed by our approach. The darker the node, the higher the coalition structure value at that node.

provided for general characteristic functions but that can be computed for large scale systems. In particular, here we describe an approach to efficiently estimate the maximum gain for *m+a* functions (Bistaffa et al., 2014). We then consider a notable example of such class (i.e., the collective energy purchase function) in Section 5.

Recall that for *m+a* characteristic functions we can decompose v in the sum of a super-additive and sub-additive component $v(C) = v^+(C) + v^-(C)$. Let us consider the gain for the super-additive part of such characteristic function and call this Partial Gain. We can now give the following definition:

Definition 4 (*Partial Maximum Gain for a coalition***).**

$$G_C^+ = \max\left\{0, \max_{S\subset C} v^+(C) - v^+(C\backslash S) - v^+(S)\right\}$$

next, we can show the following lemma:

Lemma 2 (Partial maximum gain is an upper bound for the gain). *For every coalition $C \subseteq A$*

$$G_C \leq G_C^+$$

Proof. Intuitively, the lemma holds because the partial maximum gain discards the gain related to the sub-additive component of the characteristic function, which is always negative. See the supplementary material for a more detailed proof.□

Now, we consider the marginal gain of a coalition as the difference between the worth of such a coalition with respect to the sum of the worth of the singletons that form this coalition, i.e., $v(C) - \sum_{a\in C} v(a)$, and we show that for every coalition, the marginal, partial gain of the grand coalition is an upper bound of the partial gain of such coalition:

Theorem 2 (*Partial marginal gain is an upper bound on partial maximum gain***).**

$$G_C^+ \leq v^+(A) - \sum_{a\in A} v^+(a)$$

Proof. See the supplementary material.

Theorem 2 together with Lemma 2 provide us with a tractable method to compute an upper bound of the maximum gain for the energy characteristic function:

$$\widehat{G}_M = v^+(A) - \sum_{a\in A} v^+(a) \tag{10}$$

Hence, based on the analysis provided in Section 4.1 we can use $\widehat{G}_M$ (instead of G_M) to compute an upper bound of the optimal solution. We call this the **Estimated Maximum Gain Bound (EMGB)**: $V(\mathcal{S}) + (|A| - 1)\widehat{G}_M$.

More important, the computation of such an upper bound has no exponential element, and it only requires computing the partial value of the grand coalition and summing up the partial values of all singletons. As such, it enables us to provide bounds for very large scale systems (i.e., more than 2700 agents as shown in Section 5). Clearly, since $\widehat{G}_M$ is an upper bound of G_M, the tightness of such upper bound depends on the shape of the characteristic function and we need an empirical evaluation to establish the significance of this bound. We provide such an analysis in the next section.

5. Empirical evaluation

Having described and proved the theoretical properties of C-Link, we now evaluate it under various settings, in order to determine its performance both on synthetic and real-world datasets. In particular, our algorithm has been evaluated on four data sets: (i) a synthetic benchmark data-set where the values of coalitions follow a uniform distribution (Di Mauro et al., 2010), (ii) a synthetic data-set where the values of coalition structures are normally distributed (Rahwan et al., 2009), (iii) a collective energy purchasing scenario (Vinyals et al., 2012) and (iv) a ride-sharing scenario (Bistaffa et al., 2015). Crucially, our evaluation of CSG algorithms on two real-world large scale datasets is the first of its kind, and hence is our main focus.[9]

5.1. Evaluation methodology

The main goals of the empirical evaluation are:

1. to validate the applicability of the C-link method in large scale systems;
2. to evaluate the performance loss due to the myopic nature of the approach;
3. to assess the relative performance of the different linkage functions defined in Section 3.1, and
4. to evaluate the significance of the guarantees on the solution quality detailed in Section 4.

To achieve these goals, it is important to compare C-Link against other existing approaches for CSG. In our evaluation, we focus on the quality of solutions, since the runtime (typically employed to validate current optimal and approximate approaches) is not a concern for C-Link: it terminates in *fractions of a second* for problems involving 30 agents compared to the best anytime optimal CSG algorithm (IDP-IP*) which is only shown to terminate in around 100 s.

The challenge here is that it is impossible to find the optimal solution for large numbers of agents (since none of the optimal CSG algorithms can solve for more than 30 agents).

Hence, we define three metrics of performance: the total gain value for settings that are not solvable using optimal algorithms, and the averaged gain/optimal ratio for settings involving small numbers of agents and solvable using existing optimal or approximate algorithms (e.g., MIP, IDP-IP, or Service and Adams):

- **Total Gain Value (TGV)** $\mathcal{G}^m$ is computed as: $\mathcal{G}^m = \frac{\sum_{C\in CS^m} v(C) - \sum_{a\in A} v(a)}{\sum_{a\in A} v(a)}$, where m is a coalition formation method and CS^m the coalition structure that the method returns. This

[9] The matlab code for C-Link that we used in our experiments can be downloaded from this link http://profs.sci.univr.it/~farinelli/CLink-Code.zip

indicator measures how valuable it is for the system to form the computed coalition structure as opposed to singleton coalitions. The TGV is our main performance indicator for large numbers of agents as it does not require running an optimal algorithm.

- **Averaged Gain Ratio** (**AGR**) is computed as: $\frac{\mathcal{G}^m}{\mathcal{G}^{opt}}$, where $\mathcal{G}^{opt}$ is the optimal total gain value, i.e., the total gain value for the optimal coalition structure (computed with a benchmarking algorithm). This indicator measures how far the total gain of the computed solution is from the maximum achievable total gain. This is our main performance indicator for small numbers of agents since it requires knowing the optimal solution.
- **Averaged Optimal Ratio** (**AOR**) is computed as: $\frac{V(CS^m)}{V(CS^*)}$, where CS^m is the coalition structure returned by the method m and CS^* is the optimal coalition structure. The AOR measures how far the value of the coalition structures computed by the heuristic method is from the optimal solution.

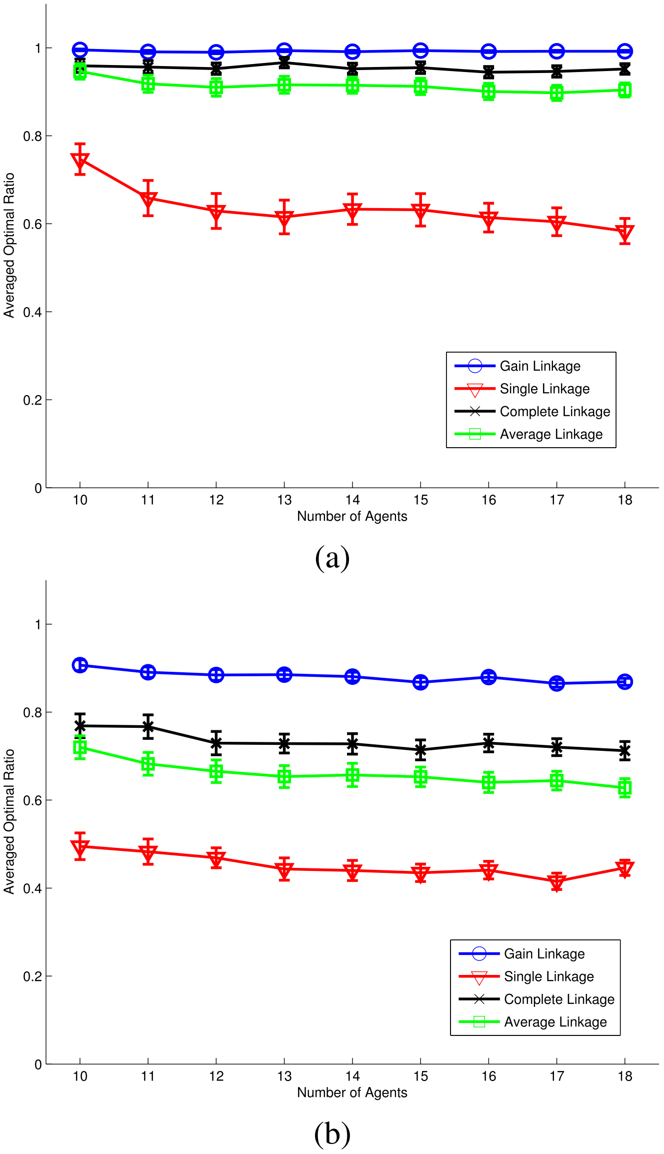
The AOR is similar to the AGR, however since the AGR removes the sum of the values of singleton coalitions, it is less dependent on the identity of the agents. This is particularly important in the energy purchasing dataset, where the identity of the agents (i.e., which users participate in the collective energy purchasing process) makes a significant difference for the value of the coalitions. Consequently, if we use the AOR to analyse the performance of the heuristic approaches, our results would be significantly biased by which agents we include in the empirical analysis, which is not an element we try to optimize (i.e., we aim at evaluating how well the heuristic approaches behave given a set of participating agents on which we have no control). In contrast, for the synthetic datasets, where the value of the characteristic function does not depend on the identity of the agents, we use the AOR as it has been used in previous work (specifically in Di Mauro et al., 2010) and hence it allows us to compare our results with such previous approach. Finally, for the energy purchasing case, we evaluate the tightness of the bounds proposed in Section 4, by computing the Maximum Performance Ratio.

In our experiments we computed the optimal solution for the CSG problem by using a standard linear programming formulation (Rahwan, 2007) (see pages 38–39). In such formulation we have one binary variable for each possible coalition x_j where x_j=1 indicates that coalition C_j has been selected. The problem is then to maximize $\sum_{j=1}^{2^N} v(C_j)*x_j$ subject to the constraints $\forall\ a_j \in A,\ \sum_{j=1}^{2^N} y_{i,j}*x_j = 1$ where $y_{i,j}$ is a binary variable and $y_{i,j} = 1$ if agent $a_i \in C_j$. The constraints enforce that the selected coalitions form a partition of the set A. Our implementation uses the CPLEX library (V12.4).

5.2. Synthetic data-sets

Here, we discuss the results obtained in two synthetic datasets, where the characteristic functions are built following specific distributions. In particular, in the first case the characteristic function follows the uniform distribution, i.e., $v(C) \sim U(a, b)$ where a=0 and b=1. This distribution has been widely used, in particular in Di Mauro et al. (2010) to evaluate the GRASP-based heuristic approach.

In more detail, Fig. 5(a) reports the AOR for the four C-Link variants varying the number of agents from 10 to 18. We stop at 18 agents because our CPLEX implementation for the optimal benchmarking algorithm runs out of memory when adding more agents. Results are averaged over 100 repetitions of experiments with 100 different instantiations of the characteristic function. Error bars report the 95% confidence interval.[10] Based on these results we can see that C-Link with the Gain-Link linkage function achieves solutions which are very close to the optimal value, with a worst case AOR of 99% (in fact, Gain-Link achieves exactly the optimal solution on average in 30% of the cases), moreover the averaged optimal ratio is almost constant with respect to the number of agents. These results suggest that the Gain-Link variant of C-Link achieves similar performance when compared to previous state-of-the-art heuristic approaches: according to Di Mauro et al. (2010), in similar settings (i.e, a uniform distribution for the characteristic function and a range of agents that goes from 14 to 20), the GRASP-based approach achieves an AOR which is always above 99% (see Fig. 3 in Di Mauro et al., 2010). Hence, in this setting, the GRASP-based approach is slightly superior to Gain-Link. This is not surprising because, as mentioned in the introduction, GRASP offers a richer variety of operations to build the coalition structure (e.g., coalition splitting and agent exchange) as well as a randomized search component to avoid local minima. However, by using a simpler mechanism to build the coalition structure, Gain-Link can provide guarantees on performance while maintaining a high quality of the final solution.



(a)

(b)

Fig. 5. Averaged optimal ratio when varying the number of agents for: (a) Uniform distribution and (b) NDCS.

As for the comparison with the different C-Link variants, the Gain-Link one clearly shows superior performance over the other approaches; average-link and complete-link have comparable performance and single-link clearly performs the worst. This behaviour can be explained by

[10] When the error bars do not overlap, the null hypothesis (i.e., not equivalence in performance) can be validated with α=0.05.

Table 1
Statistics for coalitions in the energy domain (18 agents, γ=1.3).

	Optimal	Gain L.	Single L.	Comp. L.	Avg L.
Numb. of Coal.	**7.9700**	7.7700	1.7100	8.1900	7.0200
Avg Size	**2.4166**	2.4836	12.7350	2.2702	2.6717
Max Size	**4.4700**	5.1500	17.2900	4.1900	5.5300
Min Size	**1.1500**	1.1500	8.9900	1.0400	1.1000

considering the different linkage functions that define these approaches. In particular, complete-link sets the suitability between C_i and C_j as the worst pairwise case (the minimum of suitability between all possible pairs of agents in C_i,C_j). Hence, two groups are likely to be joined together only if for *all* pairs of agents we have a high suitability, that is a coalition is formed only if it is convenient for all agent pairs. A similar reasoning applies to the average-link scheme. In contrast, single-link uses the maximum operator, hence if two agents of two different groups work very well together the two groups will be merged no matter how well the other agents fit. Consequently, single-link tends to form big coalitions (as Table 1 confirms) and does not properly take into account the synergies between groups of agents that are bigger than two.

The second synthetic data-set we consider uses the Normally Distributed Coalition Structures (NDCS) proposed by Rahwan et al. (2009), a challenging distribution for which it was shown to be difficult to provide high quality anytime solutions.[11] The authors showed that NDCS can be generated as follows: $v(C) \sim \mathcal{N}(\mu, \sigma)$ where C is the coalition, $\mu = |C|$ and $\sigma = \sqrt{|C|}$. Fig. 5(b), reports the AOR for the four C-Link variants, varying the number of agents from 10 to 18. Such results confirm the good behaviour of Gain-Link (which achieves in the worst case solutions that are about 90% of the optimal) and the relative behaviours of the other variants (i.e., average-link and complete-link have comparable performance and single-link performs the worst).

5.3. Collective energy purchasing domain

We now turn to the empirical results obtained in the collective energy purchasing domain.

In more detail, in this setting, each agent is characterized by an energy consumption profile that represents its energy consumption throughout a day. In particular, a profile records the energy consumption of a household at fixed intervals (every half hour in our case). Hence each profile is a vector of T elements (where T=48 in our case). In the following experiments we use a set of energy profiles collected, over a month, from 2732 households in UK.

The characteristic function of a group of agents is the total payment that the group would incur if they buy energy as a collective. A collective of agents buys its aggregated demand (i.e., the point-wise sum of energy profiles) in the electricity market and optimizes its buying strategy by exploiting reduced tariffs available in the forward market.[12]

In particular, following Vinyals et al. (2012) the characteristic function is defined as:

$$v(C) = \sum_{t=1}^{T} \hat{q}_S^t(C)\cdot p_S + T\cdot\hat{q}_F(C)\cdot p_F + \kappa(C) \tag{11}$$

where $C \subseteq A$, p_S and p_F represents the unit price of energy in the spot and forward market respectively,[13] $\hat{q}_F(C)$ stands for the time unit amount of electricity to buy in the forward market and $\hat{q}_S^t(C)$ for the amount to buy in the spot market at time slot t. These quantities are the ones that optimise the buying strategy of the group while satisfying the group electricity demand:

$$q_S^t(C) + q_F(C) \geq e_C^t \quad \forall\ t = 1\ldots T \tag{12}$$

In other words, to compute the characteristic function for a coalition C one has to solve a maximization task so as to optimize the buying strategy of the group. However, this maximization task is not a computational bottleneck for our coalition formation problem as it can be easily solved by using a linear programming approach (with a linear number of constraints) or the ad-hoc procedure proposed in Vinyals et al. (2012). In our experiments we use this second method.

Finally, $\kappa(C)$ stands for a coalition management cost that depends on the size of the coalition and captures the intuition that larger coalitions are harder to manage. The definition of this cost depends on several low level issues (e.g., the power network capacity of customers in the groups, legal fees, and other costs associated to group contracts etc.), hence a precise definition of this term goes beyond the scope of the present paper. Here we use $\kappa(C) = -|C|^\gamma$ to introduce a non-linear element that penalizes the formation of big coalitions, so that the grand coalition is not always the best coalition structure.

Following Bistaffa et al. (2014) we observe that $v(C)$ can be decomposed into the sum of two components:

$$v(C) = \underbrace{\sum_{t=1}^{T} \hat{q}_S^t(C)\cdot p_S + T\cdot\hat{q}_F(C)\cdot p_F}_{v^+(C)} + \underbrace{\kappa(C)}_{v^-(C)} \tag{13}$$

where, $v^+(C)$ is super-additive and $v^-(C)$ is sub-additive. To see why $v^+(C)$ is super-additive, consider that by aggregating two energy profiles we will always buy more (or the same amount) of energy in the forward market, hence, since energy prices for the forward market are lower than prices for the spot market, we can never pay more by aggregating profiles. On the other hand, $v^-(C)$ is sub-additive by definition (when $\gamma \geq 1$). Hence, based on the analysis reported in Section 4.2 for such characteristic function we can provide an upper bound on the value of the optimal coalition with a tractable procedure (see Section 5.3.1).

In all the following experiments, the results are averaged over 100 repetitions and, as before, in all figures the error bars represent the 95% confidence interval. In each run, a group of agents is randomly sampled from the whole dataset (the size of each group is specified for each experiment).

We performed various experiments to evaluate different aspects of the C-Link approach. In particular, in the first experiment, we compare the solution quality achieved by all variants of C-Link by measuring the AGR and varying the number of agents from 10 to 18. The results reported in Fig. 6(a), confirm the behaviours discussed in Fig. 5(b), with Gain-Link consistently achieving solutions which are very close to the optimal (98% of the optimal in the worst case).

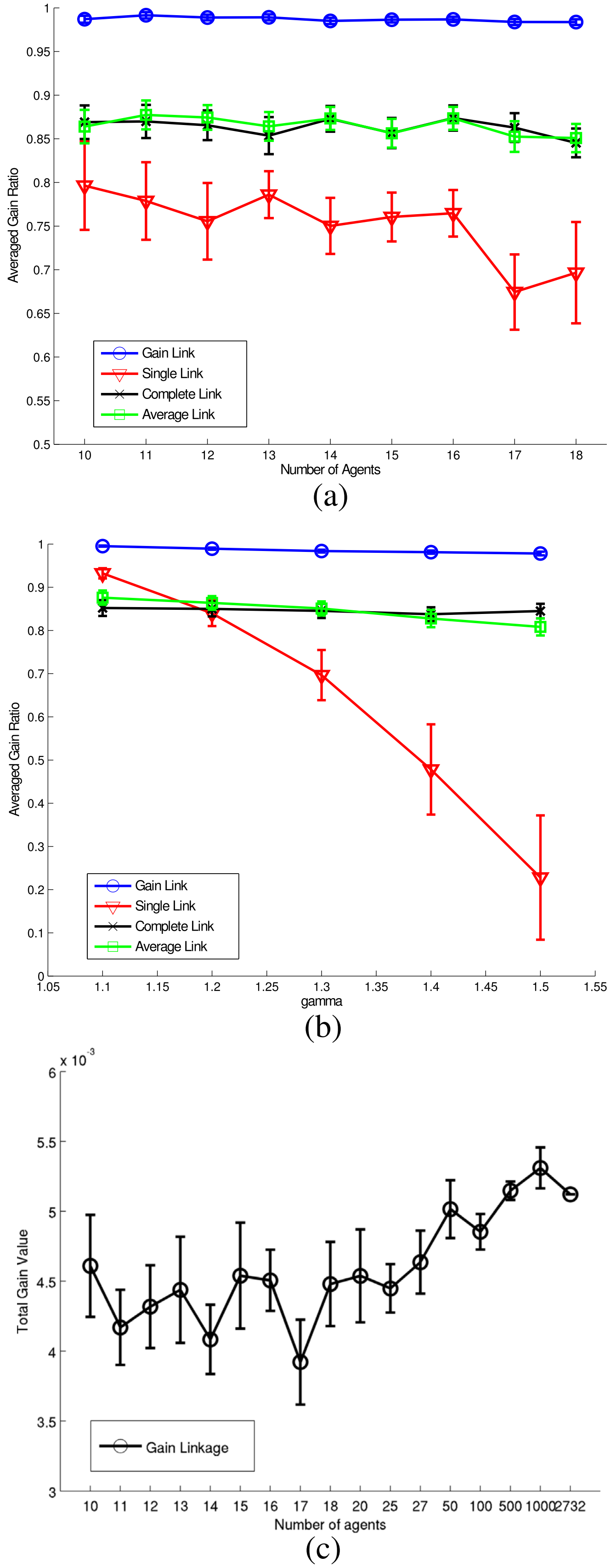
To see whether the C-Link approach can provide significant results for large numbers of agents, we now analyse how the total gain value metric evolves when increasing the number of agents to more than 2700. In particular, Fig. 6(c) reports the total gain value (TGV) against the number of agents, up to 2732 agents.[14] As the plot shows, the total gain value remains almost constant when increasing the number of agents, indicating that Gain-Link can provide high quality solutions even when the number of agents increases to thousands. Notice that the absolute value of the TGV is between 0.003 and 0.006, which indicates a small increment in this metric. However, such small

[11] Specifically, Rahwan et al. (2009) prove that the NDCS distribution ensures that every coalition structure value is drawn from the same distribution, and hence it ensures that the search space is not biased towards coalition structures that have specific features (i.e., large or small coalitions).

[12] In the forward electricity market agents can buy energy bulks in advance at reduced tariffs (see Voice et al., 2011).

[13] Unit prices are negative values to reflect the direction of payment; following Vinyals et al. (2012) in our experiments we fixed $p_S = -80$ and $p_F = -70$.

[14] In this experiment we performed 200 repetitions for each number of agents. This is because in some of these experiments we are selecting a relatively small number of agents (i.e., from 10 to 50,) from a large data-set, and the profiles of the agents that we select have a significant impact on the results. By performing more runs we managed to have an acceptable standard error of the mean and significant results.



(a)

(b)

(c)

Fig. 6. Energy Experiment, results for the Averaged Gain Ratio varying: (a) the number of agents (γ=1.3); (b) the parameter γ (18 agents). Figure (c) reports the Total Gain Value for Gain-Link varying the number of agents (γ=1.3).

increase in the TGV is not due to poor performance of Gain-Link but to a specific feature of the collective energy purchasing scenario. This is confirmed by our experiments in the ride-sharing scenario (see Section 5.4) where the TGV significantly varies when increasing the number of agents.

We now evaluate the sensitivity of the C-Link variants to the γ parameter. In particular, we fix the number of agents to 18 and vary γ. Results reported in Fig. 6(b) largely confirm the behaviour of Gain-Link, complete-link and average-link and show that the approaches are not sensitive to this parameter. In contrast, single-link shows a strong decrease in performance when γ increases. This is because, as mentioned before (see Section 5.2), single-link tends to form big coalitions that get penalized when the γ parameter is increased, as clearly shown in Table 1. From this table it is also evident that Gain-Link forms coalition structures with a number of coalitions that is very similar to the ones formed by the optimal approach.

To investigate the runtime requirements for Gain-Link when scaling to large numbers of agents, we measure the runtime in seconds for Gain-Link and the optimal linear programming approach, increasing the number of agents from 10 to the total number of users in our data-set (2732 agents). The results are reported in Figs. 7(a) and (b), from 10 to 18 agents and from 20 to 2732, respectively.

These results show that Gain-Link can provide solutions for thousands of agents in few minutes (about 4 min for 2732 agents). To put this in context, IDP-IP*, the current state of the art algorithm to solve CSG optimally in the NDCS data-set is reported to take nearly 2 min for 25 agents (Rahwan et al., 2011) while C-Link terminates in a few milliseconds for the same number of agents and the same data-set.

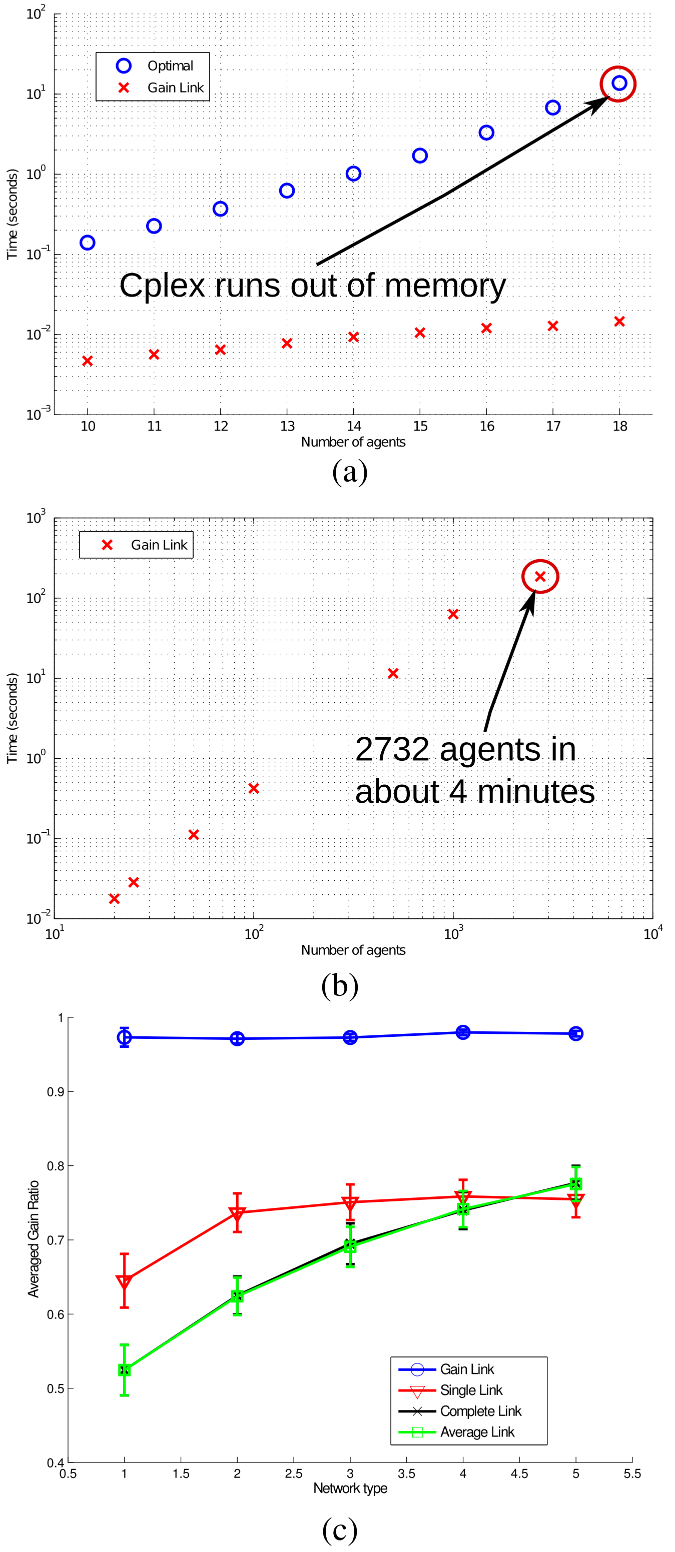
Finally, we evaluate the quality of solutions returned by all C-Link variants when feasible coalitions are restricted by a social network that connects the agents, as discussed in Section 3.4. To this end, we fix the number of agents to 18 and vary the type of social network that connects the agents. In more detail, following Voice et al. (2012), we consider a scale-free network (using the Barabasi-Albert model) and vary the connectivity parameter from 1 to 5. Results reported in Fig. 7(c) confirm that Gain-Link is superior to all the other approaches and show that it is not sensitive to the sparsity of the network. Single-link performs better than complete/average-link when the network is very sparse. This is because, as mentioned before, the bad performance for single-link is mainly due to its tendency to form big coalitions but this is mitigated by the presence of the interaction graph that constrains coalitions that can be formed.

5.3.1. *Bounds on solution quality*

Here we discuss results concerning the guarantees on solution quality that we can provide by using the bounds defined in Section 4. Recall that we provided two classes of bounds: the general Lemma 2, which however suffers from computational issues, and its computationally tractable version, which uses Eq. (10) to approximate $G_{\mathcal{M}}$. This last approximation holds only when the characteristic function exhibits a $m+a$ nature, as in this collective energy purchasing context. In this subsection we use the Maximum Performance Ratio (MPR) (See Section 4) to evaluate the tightness of such bounds.

In particular, we compare the MPR obtained with the Estimated Maximum Gain Bound (EMGB, see Equation (10)) and the MPR obtained with the Maximum Gain Bound (MGB, see Lemma 1) – clearly this can be done only for very few agents. We report such results in Fig. 8(a) and (b). In this case we can see that the EMGB is less tight than the MGB, however the computation of such bound is very cheap (from a computational standpoint) and hence we are able to provide such bound for more than 2700 (i.e., 2732 which is our full data set).

When taken together, our results show that we can provide significant optimality guarantees for more than 2000 agents, which goes well beyond the current state of the art. Again, following Sandholm et al. (1999) we can provide any bound only after looking at all the coalitions. This is clearly impractical for 2700 agents, and in

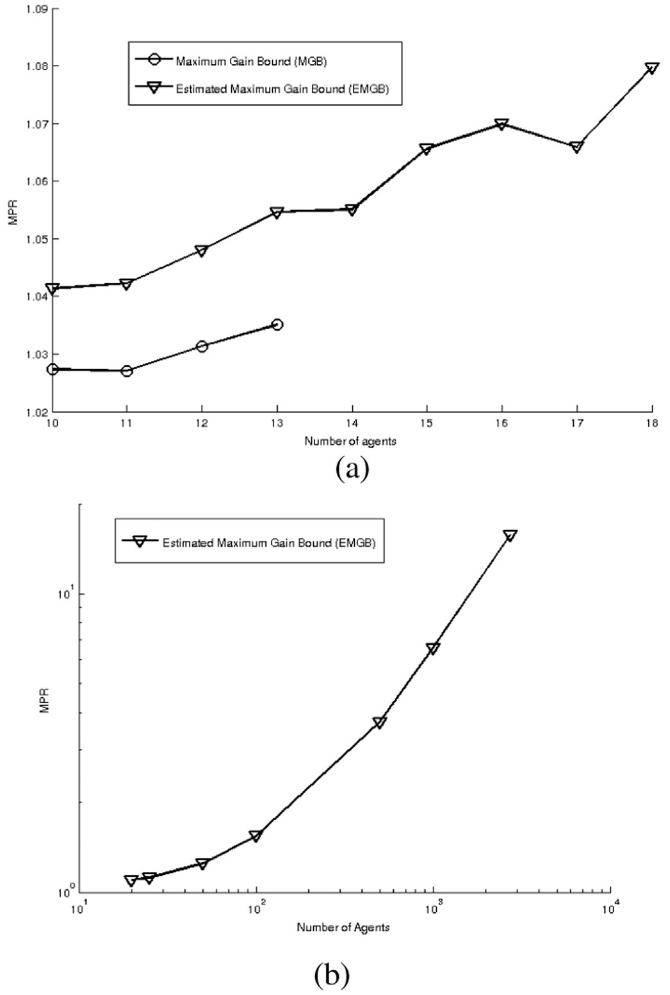


(a)

(b)

(c)

Fig. 7. Energy Experiments: run time (seconds) for Gain-Link and our optimal benchmarking approach varying the number of agents: (a): results from 10 to 18 agents (y axis in log-scale); (b): results from 20 to 2732 agents (x and y axis in log-scale). Averaged Gain Ratio when varying the kind of Barabasi Network: 18 agents, with γ=1.3 (c).



(a)

(b)

Fig. 8. Guarantees on solution quality, Maximum Performance Ratio (MPR) varying the number of agents for (a) the energy domain (up to 18 agents), and (b) the energy domain for the full data-set (up to 2732 agents).

fact most existing approaches cannot do so for anything more than 30 agents (Service and Adams, 2011; Rahwan et al., 2012).

5.4. Ride-sharing

As a final test, we evaluate our approach on another challenging real world scenario, related to the car sharing domain. In such domain each agent is a commuter that must reach a desired destination from its current position by moving on a road network. Commuters can form car and share rides aiming at minimizing the transportation cost (e.g., fuel). Hence the underlying Coalition Structure Generation problem is that of forming coalitions of commuters (i.e., join in a car) so that each agent can reach its destination while minimizing the transportation cost. In particular, here we focus on the model provided by Bistaffa et al. (2015) where users can be connected by a social network, and therefore the problem is a Graph Constrained Coalition Formation, where the interaction graph (i.e., the social network) restricts the coalitions that may be formed.

In particular, the ride-sharing problem we consider here is defined by a set of riders $\mathcal{R} = \{r_1, \ldots, r_R\}$, and a set of drivers $\mathcal{D} \subseteq \mathcal{R}$. Every driver $r_i \in \mathcal{D}$ can host up to $s(r_i)$ riders in his car, including himself, where the function $s\colon \mathcal{D} \to \mathbb{N}^+$ provides the number of seats of each car. Given a set of riders $C \subseteq \mathcal{R}$, C is said to be a *valid* coalition if at least one rider is a driver and owns a car with enough seats for all the riders. Moreover, here we consider also that each car will have at most one driver (i.e., a driver always drive his/her car). As mentioned in Bistaffa et al. (2015) this is an additional constraint that holds in several established real-world services (e.g., *Uber*). The road network is represented as a graph $\mathcal{M} = (\mathcal{P}, \mathcal{Q})$, where each node is a geographical location and a path over such graph is represented as an n-tuple of locations (i.e., $P \in \mathcal{P}^n$). Each rider $r_i \in \mathcal{R}$ has a starting point $(p_i{}^{a})$ and a destination point $(p_i{}^{b})$. Given a valid coalition C we indicate all valid paths for such locations as $\mathcal{VP}(C)$, where a path P is valid for a coalition C if P goes from the driver's starting point to its destination, and for each rider in C, its starting point precedes its destination. The characteristic function for a coalition C is then defined as:

$$v(C) = \begin{cases} cost(P_C), & \text{if } C \cap \mathcal{D} \neq \varnothing \\ k(C), & \text{otherwise.} \end{cases} \tag{14}$$

where (based on Kamar and Horvitz, 2009) P_C is the optimal path for C and $cost(P_C): \mathcal{P}^n \to \mathbb{R}^-$ is a negative cost function which typically involves different costs (such as the time cost, the cognitive cost and the fuel cost of driving through a given path). Moreover, $k(\{r_i\})$ is the cost for a rider to use public transportation.[15]

Furthermore, P_C is defined as follows:

$$P_C = \underset{P_i \in \mathcal{VP}(C)}{\operatorname{argmax}}\, cost(P_i) \tag{15}$$

As described in Bistaffa et al. (2015) the cost model considers only fuel expenses, i.e., $v(C) = K_{fuel} \cdot \overline{P_C}$, where $\overline{P_C}$ represents the length of P_C in km, $K_{fuel} = -0.0\overline{6}$ €/km (considering a fuel cost of −1 € per litre and an average consumption of 1 litre of fuel every 15 km) and $k(\{r_i\}) = -3$ € $\forall\ r_i \in \mathcal{R}$, which represents the average public transportation cost, i.e., a bus or a train ticket. Moreover, we assume that each car has a capacity of 5 seats, i.e., $s(r_i) = 5\ \ \forall\ r_i \in \mathcal{D}$.

In order to test how C-Link behaves in this specific context, we used the same data-set defined in Bistaffa et al. (2015). Briefly such data-set considers realistic data, both for the map and the social network. In particular, the map is a realistic representation of the city of Beijing, derived from the GeoLife[16] dataset provided by Microsoft, which comprises 17621 trajectories with a total distance of about 1.2 million km, recorded by different GPS loggers and GPS-phones with a variety of sampling rates. This pool of trajectories is adopted to sample random paths used to provide the start and destination points of the riders in our experiments.

Given this scenario, in the first experiment we compare the solution quality achieved by all variants of C-Link by measuring the AGR and varying the number of agents from 10 to 18. Here we use as interaction graph a subgraph of a large crawl of the Twitter social network in 2010. The results reported in Fig. 9(a), confirm the behaviours of C-link observed in the previous experiments, with Gain-Link consistently achieving solutions which are very close to the optimal (91% of the optimal in the worst case).

Fig. 9(b) compares the total gain value (TGV) for Gain-Link against the TGV achieved by CFSS, the solution approach used in Bistaffa et al. (2015). CFSS is designed for CSG where the coalitions are restricted by an interaction graph, and is an anytime approach that has been shown to perform well beyond state of the art competitors in the ride-sharing scenario. Specifically, here we let Gain-Link run until termination and then we let CFSS run for the same amount of time used by Gain-Link. In these experiments the number of agents varies from 1000 to 2000, and we again used a large crawl of Twitter as an interaction graph. Results show that Gain-Link can provide solutions of better quality up to 1500 agents, but then its performance degrades and CFSS significantly outperforms Gain-Link for 2000 agents. This is due to the greedy nature of Gain-Link coupled with the restriction imposed by the Twitter social network: Gain-Link can not recover from greedy choices that do not result in high quality solutions because the possible choices on coalition merges are limited by the social network. In contrast CFSS, which is based on a Branch-and-Bound method, exploits the presence of the network to reduce the search space and can search for better solutions.

This behaviour is largely confirmed by the results shown in Fig. 9(c)

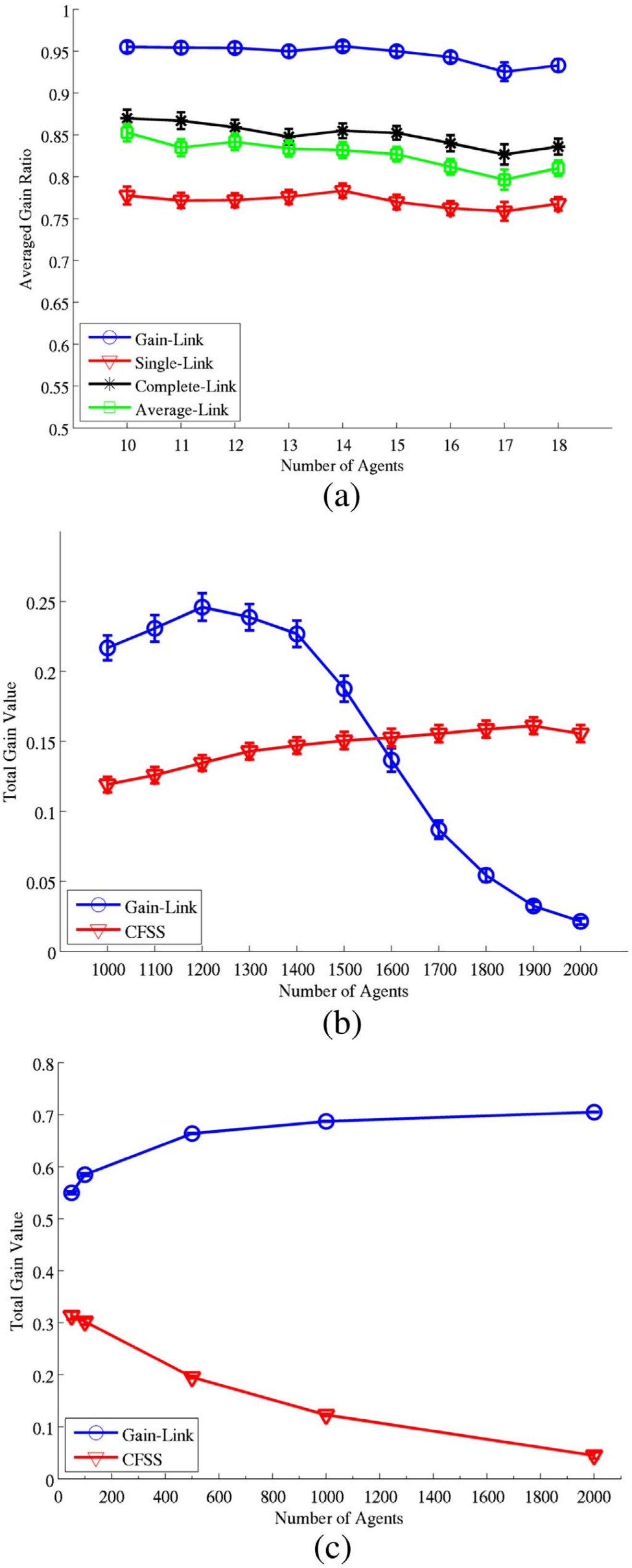


Fig. 9. Ride-sharing experiments: (a) reports the Averaged Gain Ratio for the four variants of C-Link varying the number of agents from 10 to 18 (using Twitter network); (b) compares the Total Gain Value for Gain-Link and CFSS (Bistaffa et al., 2015) varying the number of agents from 1000 to 2000 (using Twitter network); (c) compares the Total Gain Value for Gain-Link and CFSS varying the number of agents from 50 to 2000 (without any social network).

[15] Notice that, based on previous discussion if a coalition does not contain any drivers (i.e., $C \cap \mathcal{D} = \varnothing$) then C must be formed by a single rider without a car, hence its cost is provided by $k(\cdot)$.

[16] http://research.microsoft.com/en-us/projects/geolife

where we compare the TGV between Gain-Link and CFSS varying the agent numbers from 50 to 2000 without imposing any social network. In this case, Gain-Link is significantly superior to CFSS and its performance increases when the number of agents increases.

As a final observation, notice that, as discussed in Bistaffa et al. (2015), the characteristic function for ride-sharing is not an $m+a$ function, since it depends on P_C, and specifically on the actual position of the start and destination points of the riders. In particular, this characteristic function exhibits a monotonic behaviour for some configurations, for example, when a rider's start and end positions are aligned with the path of a driver. However, it is antimonotonic for other configuration for example, when a rider's start and end positions are both off the path of a driver (see Bistaffa et al., 2015 for a more detailed discussion on this topic). As a consequence, we can not provide tractable bounds for this characteristic function.

6. Conclusions

In this paper we focus on providing good-enough solutions to the CSG problem. Specifically, we draw a parallelism between the CSG problem and data clustering proposing a novel scalable heuristic called C-Link. Our analysis shows that C-link has minimal requirements, when compared to previous approaches for CSG, in terms of memory ($O(N^2)$) and computation ($O(N^3)$). We show that the C-link is guaranteed to converge in N iterations at most, and that the Gain-Link variant of the C-Link approach is anytime. Moreover, we provide a refinement for C-link that can solve the CSG problem when interaction graphs constrain the formation of coalitions, maintaining all the appealing properties of C-link. Furthermore, we show that by bounding the gain that can be achieved when merging two coalitions, we can provide an upper bound on the value of the optimal coalition structure. We were also able to provide a computationally tractable version of such bound for specific characteristic functions ($m+a$ functions).

Our empirical analysis compares C-Link against an optimal CSG algorithm in 4 different datasets, showing that our approach can provide high quality solutions; in particular, our approach provides high quality solutions *and* quality guarantees in specific settings (i.e. for $m+a$ functions), such as the collective energy purchasing domain. More importantly, it is able to solve problems involving thousands of agents (more than 2000) in few minutes (less than 4) for any characteristic function, with and without a network that constrains the possible coalitions. When taken together, the analysis of several variants of the approach and our empirical results provide the first benchmarks for large-scale approximate coalition structure generation and open up several promising future directions. Specifically, as mentioned in the introduction a key design concept for C-link is the simplicity of the proposed clustering scheme which fosters an analysis of the approach; even if we believe that a basic clustering technique is very appropriate for a first attempt to use clustering for coalition formation, an interesting future direction is to investigate other clustering schemes to improve on the performance of C-Link. In this spirit, we believe that employing partitional clustering methods for coalition formation might be a promising research direction and it is definitely part of our future work in this space.

Overall, the analysis we provide here suggests that in most realistic application domains achieving optimal solutions for the CSG problem is not practical. In fact, the inherent complexity of the CSG problem forces exact approaches to have computational costs and memory requirements that do not allow to scale beyond few tens of agents (which is far below the number of agents of realistic CSG problems). Nonetheless, it possible to devise algorithms that can provide good quality solutions and bounds by considering specific families of characteristic functions (such as the $m+a$ function). In this sense the concept of gain is a key element to consider, and having a good estimation of how beneficial it is to merge coalitions is a powerful indicators that can help the system designer to devise high quality heuristic approaches. Nonetheless, we also highlight that the current bounds should be refined to be useful in practical scenarios. In particular, we believe this to be a very important and promising research direction for applications where a large number of agents may be involved, such as for example the formation of collectives in virtual power plants for intelligent energy management (Chalkiadakis et al., 2011) or coalition formation for first responders involved in search and rescue operations (Ramchurn et al., 2010).

As a final remark, we believe clustering techniques could also benefit from coalition formation approaches, as there are challenges for clustering/data mining to look into more complex high order clustering problems where the relationship between entities are combinatorial in nature. We believe in this setting clustering techniques should be augmented with approaches used in coalition formation research (such as dynamic programming or branch-and-bound), to move away from standard greedy approaches and provide solutions of higher quality and/or quality guarantees.

Acknowledgment

This work was supported by the EPSRC-Funded ORCHID Project EP/I011587/1

Appendix A. Supplementary data

Supplementary data associated with this article can be found in the online version at http://dx.doi.org/10.1016/j.engappai.2016.12.018.

References

Ackerman, M., Ben-David, S., Loker, D., 2010. Characterization of linkage-based clustering. In: Proceedings of COLT, pp. 270–281.

Ausiello, Giorgio, Crescenzi, Pierluigi, Gambosi, Giorgio, Kann, Viggo, Marchetti-Spaccamela, Alberto, Protasi, Marco, 2012. Complexity and Approximation: Combinatorial Optimization Problems and their Approximability Properties. Springer, Heidelberg, Germany.

Billard, E.A., Pasquale, J.C., 1995. Probabilistic coalition formation in distributed knowledge environments. IEEE Trans. Syst. Man Cybern. 25 (2), 277–286.

Bistaffa, Filippo, Farinelli, Alessandro, Cerquides, Jesús, Rodríguez-Aguilar, Juan, Ramchurn, Sarvapali D., 2014. Anytime coalition structure generation on synergy graphs. In: Proceedings of AAMAS, pp. 13–20.

Bistaffa, Filippo, Farinelli, Alessandro, Ramchurn, Sarvapali D., 2015 Sharing rides with friends: a coalition formation algorithm for ridesharing. In: Proceedings of AAAI, pp. 608–614.

Chalkiadakis, Georgios, Elkind, Edith, Wooldridge, Michael, 2011. Computational Aspects of Cooperative Game Theory. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool Publishers, Heidelberg, Germany.

Chalkiadakis, Georgios, Greco, Gianluigi, Markakis, Evangelos, 2016. Characteristic function games with restricted agent interactions: core-stability and coalition structures. Artif. Intell. 232, 76–113.

Chalkiadakis, Georgios, Robu, Valentin, Kota, Ramachandra, Rogers, Alex, Jennings, Nick, 2011. Cooperatives of distributed energy resources for efficient virtual power plants. In: The Tenth International Conference on Autonomous Agents and Multiagent Systems (AAMAS-2011), May, pp. 787–794.

Cormen, T.H., Leiserson, C.E., Rivest, R.L., Stein, C., 2001. Introduction to Algorithms second edition. The MIT Press, Heidelberg, Germany.

Dang, V.D., Jennings, N.R., 2004. Generating coalition structures with finite bound from the optimal guarantees. In: Proceedings of AAMAS, pp. 564–571.

Di Mauro, Nicola, Basile, Teresa M A., Ferilli, Stefano, Esposito, Floriana, 2010. Coalition structure generation with GRASP. In: Proceedings of AIMSA, vol. 6304, pp. 111–120.

Dos Santos, D.S., Bazzan, A.L.C., 2012. Distributed clustering for group formation and task allocation in multiagent systems: a swarm intelligence approach. Appl. Soft Comput. 12 (8), 2123–2131.

Elkind, Edith, Rahwan, Talal, Jennings, Nicholas R., 2013. Multi-Agent Systems. In: Computational Coalition Formation. MIT Press, Heidelberg, Germany, pp. 329–380.

Farinelli, A., Bicego, M., Ramchurn, S.D., Zucchelli, M., 2013. C-link: a hierarchical clustering approach to large-scale near-optimal coalition formation. In: Proc. of IJCAI, pp. 106–112.

Kamar, Ece, Horvitz, Eric, 2009. Collaboration and shared plans in the open world: Studies of ridesharing. In: Proc. of IJCAI, pp. 187–194.

Michalak, Tomasz, Sroka, Jacek, Rahwan, Talal, Wooldridge, Michael, Mcburney, Peter, Jennings, Nicholas, 2010. A distributed algorithm for anytime coalition structure generation. In: Proc. of AAMAS, pp. 1007–1014.

Myerson, Roger B., 1977. Graphs and cooperation in games. Mathematics of Operations Research 2 (3), 225–229.

Ohta, Naoki, Conitzer, Vincent, Ichimura, Ryo, Sakurai, Yuko, Iwasaki, Atsushi, Yokoo, Makoto, 2009. Coalition structure generation utilizing compact characteristic function representations. In: Proc. of CP, pp. 623–638.

Peleteiro, Ana, Burguillo, Juan C., Luck, Michael, Arcos, Josep Ll., Rodríguez-Aguilar, Juan A., 2015. Using reputation and adaptive coalitions to support collaboration in competitive environments. Engineering Applications of Artificial Intelligence 45, 325–338.

Rahwan, T., Jennings, N.R., 2008. Coalition structure generation: Dynamic programming meets anytime optimisation. In: Proc. of AAAI, pp. 156–161.

Rahwan, T., Jennings, N.R., 2008. An improved dynamic programming algorithm for coalition structure generation. In: Proc. of AAMAS, pp. 1417–1420.

Rahwan, T., Ramchurn, S.D., Jennings, N.R., Giovannucci, A., 2009. An anytime algorithm for optimal coalition structure generation. Journal of Artificial Intelligence Research 34, 521–567.

Rahwan, Talal, 2007. Algorithms for Coalition Formation in Multi-agent Systems. University of Southampton.

Rahwan, Talal, Michalak, Tomasz, Elkind, Edith, Faliszewski, Piotr, Sroka, Jacek, Wooldridge, Michael, Jennings, Nicholas, 2011. Constrained coalition formation. In: Proc. of AAAI, pp. 719–725.

Rahwan, Talal, Michalak, Tomasz, Jennings, Nicholas R., 2012. A hybrid algorithm for coalition structure generation. In: Proceedings of AAAI, pp. 1443–1449.

Ramchurn, S.D., Polukarov, M., Farinelli, A., Truong, C., Jennings, N.R., 2010. Coalition formation with spatial and temporal constraints. In: Proceedings of AAMAS, pp. 1181–1188.

Ramchurn, Sarvapali D., Polukarov, Maria, Farinelli, Alessandro, Truong, Cuong, Jennings, Nicholas R., 2010. Coalition formation with spatial and temporal constraints. In: The Ninth International Conference on Autonomous Agents and Multiagent Systems (AAMAS-2010), pp. 1181–1188.

Saad, W., Han, Zhu, Basar, T., Debbah, M., Hjorungnes, A., 2011. Coalition formation games for collaborative spectrum sensing. IEEE Trans. Veh. Technol. 60 (1), 276–297.

Sabar, M., Montreuil, B., Frayret, J.-M., 2009. A multi-agent-based approach for personnel scheduling in assembly centers. Eng. Appl. Artif. Intell. 22 (7), 1080–1088.

Sandholm, Tuomas, Larson, Kate, Andersson, Martin, Shehory, Onn, Tohmé, Fernando, 1999. Coalition structure generation with worst case guarantees. Artif. Intell. 111 (1–2), 209–238.

Sen, S., Dutta, P.S., 2000. Searching for optimal coalition structures. In: Proceedings of ICMAS, pp. 287–292.

Service, Travis C., Adams, Julie A., 2010. Approximate coalition structure generation. In: Proc. of AAAI. Atlanta, GA, USA, pp. 854–859.

Service, Travis.C., Adams, Julie.A., 2011. Randomized coalition structure generation. Artif. Intell. 175 (16–17), 2061–2074.

Shehory, O., Kraus, S., 1998. Methods for task allocation via agent coalition formation. Artif. Intell. 101 (1–2), 165–200.

Theodoridis, S., Koutroumbas, K., 2008. Pattern Recognition Fourth Edition. Academic Presshttp://store.elsevier.com/Pattern-Recognition/Sergios-Theodoridis/isbn-9781597492720/.

Vinyals, M., Bistaffa, F., Farinelli, A., Rogers, A., 2012. Coalitional energy purchasing in the smart grid. In: Proceedings of ENERGYCON, pp. 848 –853.

Voice, T., Ramchurn, S.D., Jennings, N.R., 2012. On coalition formation with sparse synergies. In: Proceedings of AAMAS, vol. 1. pp. 223–230.

Voice, T., Vytelingum, P., Ramchurn, S., Rogers, A., Jennings, N.R., 2011. Decentralised control of micro-storage in the smart grid. In: Proceedings of AAAI, pp. 1421–1426.

Voice, Thomas, Polukarov, Maria, Jennings, Nicholas R., 2012. Coalition structure generation over graphs. J. Artif. Int. Res. 45 (1), 165–196.

Von Luxburg, U., Williamson, R.C., Guyon, I., 2012. Clustering: science or art? In: Proceedings of ICML, pp. 65–80.

Wang, W., Jiang, Y., 2014. Community-aware task allocation for social networked multiagent systems. IEEE Trans. Cybern. 44 (9), 1529–1543.

Yun Yeh, D., 1986. A dynamic programming approach to the complete set partitioning problem. BIT Numer. Math. 26 (4), 467–474.