C-Link: A Hierarchical Clustering Approach to Large-Scale Near-Optimal Coalition Formation*

Alessandro Farinelli[a], **Manuele Bicego**[a], **Sarvapali Ramchurn**[b], **Mauro Zucchelli**[a]
[a]Computer Science Department, University of Verona, <firstname.lastname>@univr.it
[b]School of Electronics and Computer Science, University of Southampton, sdr@ecs.soton.ac.uk

Abstract

Coalition formation is a fundamental approach to multi-agent coordination. In this paper we address the specific problem of coalition structure generation, and focus on providing good-enough solutions using a novel heuristic approach that is based on data clustering methods. In particular, we propose a hierarchical agglomerative clustering approach (C-Link), which uses a similarity criterion between coalitions based on the gain that the system achieves if two coalitions merge. We empirically evaluate C-Link on a synthetic benchmark data-set as well as in collective energy purchasing settings. Our results show that the C-link approach performs very well against an optimal benchmark based on Mixed-Integer Programming, achieving solutions which are in the worst case about 80% of the optimal (in the synthetic data-set), and 98% of the optimal (in the energy data-set). Thus we show that C-Link can return solutions for problems involving thousands of agents within 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], surveillance and security applications and collective purchasing of goods or services [Vinyals *et al.*, 2012]. Coalition formation typically involves three key computational challenges: i) coalition value calculation and optimisation (i.e., evaluating the worth of each coalition and optimising the actions of individual members), ii) coalition structure generation (CSG) (i.e., partitioning the set of agents into the most beneficial coalitions) and iii) Pay-off Distribution (i.e, dividing the rewards of coalitional actions among the members). 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.

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

To date, a number of approaches employed in this domain aim to solve the CSG problem optimally, ranging from Mixed-Integer programming to Branch-and Bound techniques [Rahwan *et al.*, 2009] through Dynamic Programming (DP) [Yun Yeh, 1986; Rahwan and Jennings, 2008b]. However, none of these solutions are scalable given that the input to the CSG problem is exponential in the number of agents ($O(n^n)$). Hence these optimal approaches can handle relatively small sets of agents (i.e., fewer than 30 [Rahwan *et al.*, 2009]). Recent approaches [Voice *et al.*, 2012] exploit social relationships among the agents to restrict the set of feasible coalitions, and by so doing they can optimally solve the CSG problem for about 50 agents in sparse networks. However, while this is a significant improvement, it cannot claim to solve CSG problem in practical applications, such as, for example, collective purchasing, where thousands of agents might be involved[1]. In contrast, here we focus on providing good-enough solutions (near-optimal) using heuristic sub-optimal approaches. Along this line, previous approaches include Genetic Algorithms [Sen and Dutta, 2000] and swarm intelligence [Dos Santos and Bazzan, 2012], 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, we propose the use of data clustering algorithms, that aim to partition a data-set into groups (or clusters), based on the concept of similarity: objects in the same group should be similar, whereas objects belonging to different groups should be dissimilar. Clustering approaches offer a wealth of solutions that have been developed and empirically validated in practical applications involving large amount of data (e.g., thousands of data points) [Jain and Dubes, 1988; Theodoridis and Koutroumbas, 2008]. Now, a key challenge in this context is *the formulation of an appropriate similarity criterion* so that the data can be clustered in a meaningful way. Hence, here we propose a *suitability* function for the CSG problem that, based on the characteristic function, specifies which coalitions (i.e., clusters in the context of data clustering) are most appropriate for merging, and whether merging them is beneficial.

In more detail, this work advances the state of the art in the following ways: i) we provide a general methodology

[1]http://www.whichbigswitch.co.uk/closed/

to apply clustering techniques to coalition formation. Our key contribution here is to propose a suitability function for coalitions based on the gain that two coalitions would achieve if they merge; ii) we propose an agglomerative hierarchical clustering approach (C-Link) that starts off from singleton coalitions and iteratively merges the most suitable pairs of coalitions. The criterion to evaluate whether coalitions should be merged is based on the above-mentioned gain. In particular, we devise different criteria taking inspiration from standard clustering approaches such as single-link, complete-link and average-link, which consider only the value of coalitions of size two (e.g., they require a specification of the characteristic function only for pairs of agents). Moreover, we propose a new criterion (gain-link) which takes advantage of the full characteristic function significantly improving the quality of solutions; iii) we validate C-Link on a synthetic data-set based on [Rahwan *et al.*, 2009] and on a specific coalition formation problem where users can form groups to buy energy at discounted prices [Vinyals *et al.*, 2012]. In this scenario we use real energy consumption data collected from a set of households. We compare the above-mentioned techniques against an optimal benchmark approach based on Mixed-Integer programming (implemented using CPLEX). In our empirical evalutions, C-Link is shown to provide high quality solutions, about 98% of the optimal on the energy data-set and 80% of the optimal on the synthetic data-set. Moreover, we show that, in general, clustering 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, the C-Link approach can provide solutions for problems involving thousands of agents (more than 2500) in a few minutes (about 4).

The rest of the paper is structured as follows: Section 2 provides necessary background on coalition structure generation and data clustering while Section 3 details our C-Link approach. Section 4 presents our empirical evaluation and Section 5 concludes.

2 Background and related work

Here we provide a formalization of the CSG problem and a brief overview of most prominent data clustering approaches.

2.1 The Coalition Structure Generation Problem

Formally speaking, the optimal coalition structure generation problem finds the solution to:

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

where $\mathcal{CS}$ is the set of all partitions of the set of N agents $\mathcal{A} = \{a_1, \cdots, a_N\}$, $CS \in \mathcal{CS}$ is a coalition structure (i.e., $CS \subseteq 2^{\mathcal{A}}$) where for any $C_i, C_j \in CS$, with $i \neq j$, $C_i \cap C_j = \emptyset$ (i.e., no agent is assigned to more than one coalition) and $\cup_{C \in CS} C = \mathcal{A}$ (i.e., each agent is selected in at least one coalition). Finally, $v(C) \in \Re$ is the characteristic function, which specifies a value (that may represent a cost or profit) for each coalition. A key property of the characteristic function 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 next.

2.2 Data Clustering Approaches

Data Clustering approaches can be broadly divided in two main families: hierarchical clustering (e.g., single/complete/average-link) and partitional clustering (e.g., k-means) [Jain and Dubes, 1988].

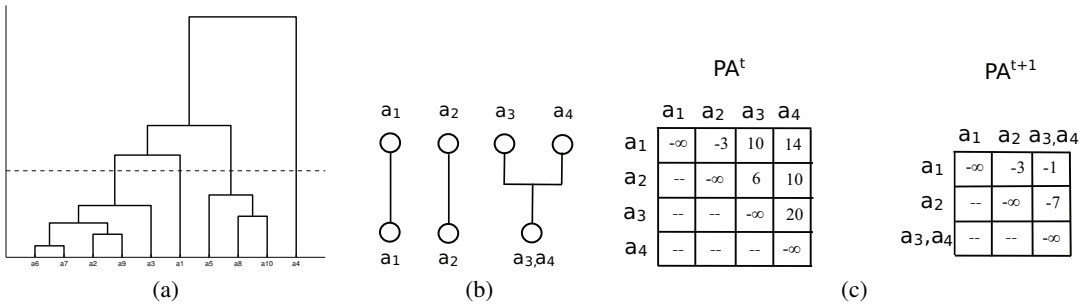
In hierarchical clustering, data is arranged in layers of partitions, where each partition is merged in a partition of the subsequent layer. Hence, a hierarchical clustering can be conveniently represented by a *dendrogram*, a tree structure that consists of layers of nodes, each representing a cluster, where lines connect clusters that are merged in the next layer (see Figure 1(a)). Agglomerative clustering approaches (such as single/complete/average-link) start from clusters formed of single data points and iteratively merge the most suited pair of clusters, where the suitability function depends on a criterion of similarity defined for the clusters. 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 solution exists.

In contrast, partitional clustering techniques consider only a single partition of the data, starting from an initial solution that is iteratively refined. As such, they are more efficient in terms of memory and computation and hence they are typically preferred for applications involving very large data-sets (e.g., millions of data points). However, the most widely used partitional clustering approaches (such as, for example, 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.

Here we adopt hierarchical agglomerative clustering for three key reasons: i) the behaviour of hierarchical clustering approaches is not dependent on any initialization or system parameter; ii) the efficiency typical of partitional clustering is not crucial here, as the number of agents that our approach can handle is already significantly beyond the capability of current coalition formation approaches; iii) in our approach the stopping criterion for the cluster merging process is automatic as we will detail in section 3.2.

3 The Coalition Link approach

The coalition link (C-Link) algorithm follows the Generalised Agglomerative Scheme (GAS) for clustering (See [Theodoridis and Koutroumbas, 2008] Chapter 13.2) which, starting from a set of agents, aims to produce a sequence of *nested* partitions $CS^0, CS^1, \cdots, CS^L$. Following [Jain and Dubes, 1988], we define that a partition CS^i is nested into a partition CS^j if every component of CS^i is a subset of a component of



(a) (b) (c)

Figure 1: Figure (a) shows 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 $\{\{a_6, a_7, a_2, a_9, a_3\}, \{a1\}, \{a_5, a_8, a_{10}\}, \{a_4\}\}$. Figures (b) and (c) 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: (b) shows the dendogram and (c) shows how the PA matrix evolves.

CS^j. Next we provide a pseudo-code description of C-Link and discuss its main features.

3.1 The C-Link algorithm

The C-Link approach is described in Algorithm 1. Essentially, the algorithm is based on the definition of a suitability function $sf(C_i, C_j)$ that indicates how convenient it is for two coalitions $C_i, C_j \in CS$ to be merged. The approach iteratively updates the Partition Suitability matrix $PS^{(t)}$, which stores the value $sf(C_i, C_j)$ 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 PS matrix with suitability 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 good for the system (line 9), 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.

Now, a key element for the C-Link approach is the definition of the suitability function. In the context of CSG, a natural way to define the suitability is to use the concept of gain, which reflects how useful it is for the system if two coalitions merge. In more detail, given two coalitions C_i an d C_j, their gain $G(C_i, C_j)$ can be defined, using the characteristic function v, as follows:

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

In other words the gain indicates how well two coalitions stay together with respect to how well they are before joining.

Given the concept of gain we explore different methods to define the concept of suitability for a pair of coalitions. In particular, we investigate methods which are based on classical concepts used in data clustering (single/complete/average-link) and which require the computation of the gain only for pairs of agents. Moreover, we provide a novel approach (gain-link) that uses directly the definition of gain between coalitions. In more detail, we investigate three definitions for the suitability function that are inspired respectively by the well known single-link (SL), complete-link (CL), and average-link (AL) versions of the clustering algorithms:

Input: $\mathcal{A}$: the set of agents, $sf(\cdot)$: the suitability function.
Output: CS_{opt} the optimal partition of $\mathcal{A}$
// Initialize partitions to singletons
1: $CS^{(0)} = \{\{a_1\}, \{a_2\}, \cdots, \{a_N\}\}$
// Initialize PS for each agent pair
2: **for** i,j = 1 to N, $i \neq j$ **do**
3: $PS^{(0)}(i, j) = sf(\{a_i\}, \{a_j\})$
4: **end for**
// Initialize self suitability to $-\infty$
5: $\forall i, PS^{(0)}(i, i) = -\infty$
// Compute and store the best indices and best suitability
6: $\hat{\imath}, \hat{\jmath} = \arg\max_{i,j} PS^{(0)}(i, j)$
7: $\hat{pa} = \max_{i,j} PS^{(0)}(i, j)$
8: t = 0
// Main Loop: stop if best suitability is negative or the grand coalition was formed
9: **while** ($\hat{pa} \geq 0$) 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 $C_{\hat{\imath}\hat{\jmath}} = C_{\hat{\imath}} \cup C_{\hat{\jmath}}$
12: $CS^{(t)} = CS^{(t-1)} \setminus C_{\hat{\imath}} \setminus C_{\hat{\jmath}} \cup C_{\hat{\imath}\hat{\jmath}}$
// Update $PS^{(t)}$
13: Delete rows and columns of $PS^{(t)}$ relative to $C_{\hat{\imath}}$ and $C_{\hat{\jmath}}$, add one row and one column for $C_{\hat{\imath}\hat{\jmath}}$.
// Compute suitability for each coalition C_k with the newly formed coalition $C_{\hat{\imath}\hat{\jmath}}$
14: **for** $C_k \in CS^{(t)}, C_k \neq C_{\hat{\imath}\hat{\jmath}}$ **do**
15: $PS^{(t)}(\hat{\imath}\hat{\jmath}, k) = PS^{(t)}(k, \hat{\imath}\hat{\jmath}) = sf(C_{\hat{\imath}\hat{\jmath}}, C_k)$
16: **end for**
// Set self suitability to $-\infty$
17: $PS^{(t)}(\hat{\imath}\hat{\jmath}, \hat{\imath}\hat{\jmath}) = -\infty$
// Update best indices and best value of suitability
18: $\hat{\imath}, \hat{\jmath} = \arg\max_{i,j} PS^{(t)}(i, j)$
19: $\hat{pa} = \max_{i,j} PS^{(t)}(i, j)$
20: **end while**
21: **return** $CS^{(t)}$

Algorithm 1: C-Link algorithm.

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

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

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

These definitions are appealing for efficiency because they require the computation of the characteristic function only for pairs of agents.

However, since for the CSG setting considering only pairwise relations may result in poor performance, we also propose the gain-link function (GL), which directly uses the definition of gain (2):

$$sf_{GL}(C_i, C_j) = G(C_i, C_j) \quad (6)$$

Notice that, the definition of suitability based on the gain naturally provides an automatic stopping criterion for the C-Link algorithm: the algorithm stops if there is no advantage in joining together the "most suitable" pair of coalitions, i.e., the best pair has a negative suitability. This is a crucial difference between GAS and C-Link, as the GAS scheme always produces a full dendrogram (i.e., from the singleton to the grand coalition), and given the dendrogram, deciding where to place a cut to obtain the "best" clustering is a key, domain dependent issue (see [Theodoridis and Koutroumbas, 2008] Chapter 13.6).

Figure 1 shows an exemplar matrix update step for our C-Link approach (using GL). In particular, Figure 1(b) shows the dendrogram and Figure 1(c) the update of the PS matrix. Here we assume the optimal coalition structure is $\{\{a_1\}, \{a_2\}, \{a_3, a_4\}\}$ and hence our approach evaluates all the possible coalitions of size two, computing the values reported in the left-hand side matrix in Figure 1(c). Now, the best option is to form the coalition $\{a_3, a_4\}$ and hence the algorithm updates the PS matrix as shown in the right-hand side of Figure 1(c). Notice that in this matrix all elements are negative and hence the algorithm would stop processing.

3.2 C-link analysis and Discussion

The main properties of C-Link are the following: i) *C-Link always converges in at most N steps* (where N is the number of agents). This is because C-Link removes one element at each iteration and stops if the grand coalition forms (or if the best suitability is negative). ii) *Gain-link is anytime*. Gain-link performs at most one merge at each step, and it performs the merge only if the *gain* is positive. Hence the sum of the values of the coalitions in each partition will never decrease. 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 suitability function, an estimation of the gain based only on pairwise relations. iii) *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 $PS^{(t)}$ for all the suitability functions defined in previous section and for all t are non-negative. Consequently, the approach stops only when the grand coalition is formed (i.e., $|CS^{(N)}| = 1$).

In terms of computational complexity, following the analysis of GAS 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, the C-Link approach must store the PS matrix which has N^2 entries. Hence space-wise the complexity of C-Link is $O(N^2)$. However, if the characteristic function is specified in a tabular form (i.e., we store one value for each possible coalition as in Section 4.2), the memory storage for gain-link is exponential in the number of agents ($O(2^N)$), as it requires the values of coalitions of any size, while for the other approaches (e.g., single/complete/average-link) it remains polynomial($O(N^2)$), as they require only the values for coalitions of size two.

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 Figure 2 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 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). Compared with the DP approach our C-Link approach 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 the next section show that the performance of the approach are extremely promising.

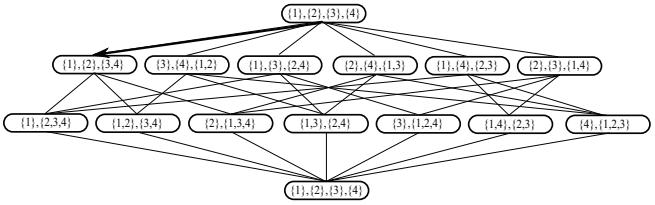


Figure 2: A diagram of the coalition structure graph for 4 agents. The downwards arrow shows the path followed by our approach.

4 Empirical Evaluation

Having described and analysed our approach we now present the empirical evaluation of C-Link. In what follows, we first discuss the methodology we use for comparison and then present results obtained in two settings: a synthetic benchmark data-set where the values of coalitions structures are normally distributed [Rahwan *et al.*, 2009] and the collective energy purchasing scenario [Vinyals *et al.*, 2012].

4.1 Evaluation Methodology

The main goals of the empirical evaluation are: i) to validate the applicability of the C-link method in large scale systems, ii) to evaluate the performance loss due to the myopic nature of the approach, and iii) to assess the relative performance of the different suitability functions defined in Section 3.1.

Hence, we compare the four variants of the C-link approach with an optimal benchmark algorithm based on Mixed-Integer programming and we compute two main performance indicators: the total gain value and the averaged gain ratio. The total gain value $\mathcal{G}^m$ is computed as: $\mathcal{G}^m = \frac{\sum_{C \in CS^m} V(C) - \sum_{a \in \mathcal{A}} V(a)}{\sum_{a \in \mathcal{A}} V(a)}$, where m is a coalition formation

method and CS^m the coalition structure that the method returns. This indicator measures how valuable it is for the system to form the computed coalition structure as opposed to singleton coalitions.

The averaged gain ratio is computed as: $\frac{\mathcal{G}^m}{\mathcal{G}^{opt}}$, where $\mathcal{G}^{opt}$ is the optimal value (computed with the benchmarking algorithm). This indicator measures how far the value of the computed solution is from the optimal, and it is our main performance indicator.

All the heuristics are implemented in MATLAB and executed on a Intel(R) Core(TM)2 Duo CPU, 1.40GHz, with 3GB of memory. The optimal algorithm is implemented using the CPLEX library (V12.4) and Java, and it is executed on a Intel(R) Core(TM) i7 CPU, 2.80GHz, with 8GB of memory.

4.2 Normally Distributed Coalition Structures

Here we report and discuss results using one of the hardest characteristic function benchmarks proposed by [Rahwan *et al.*, 2009], namely the Normally Distributed Coalition Structures (NDCS). 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 $\sigma = \sqrt{|C|}$.[2]

Figure 3, reports the averaged gain ratio for the four C-Link variants, varying the number of agents from 10 to 18. We stop at 18 agents because our CPLEX implementation 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.[3] Based on these results we can see that gain-link achieves solutions which are, in the worst case, about 80% of the optimal, moreover the averaged gain ratio is almost constant with respect to the number of agents. As for the comparison with the other C-Link variants, gain-link clearly shows superior performance; average-link and complete-link have comparable performance and single-link clearly performs the worst (less than 20% of the optimal). This behaviour can be explained by considering the update rules 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. Finally, notice that, since here we specify the characteristic function in tabular form, as mentioned in Section 3.2 the memory requirement of gain-link is exponential in the number of agents (i.e., $O(2^N)$) while all the other approaches must only store the characteristic function for agent pairs. Consequently, the average-link and complete-link approaches might be valid alternatives for scenarios where storing the characteristic function is an issue.

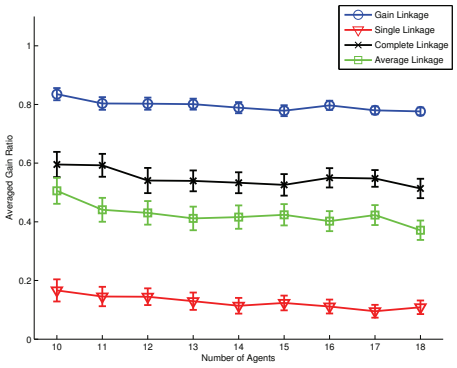


Figure 3: NDCS: Averaged Gain Ratio when varying the number of agents.

4.3 Collective Energy Purchasing Domain

We now turn to the empirical results obtained in the collective energy purchasing domain. We consider here a scenario where energy consumers form groups to purchase energy at better prices [Vinyals *et al.*, 2012].

In particular, in this setting each agent is characterized by an energy consumption profile that represents its energy consumption throughout a day. In more detail, 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.[4]

In particular, following [Vinyals *et al.*, 2012] the characteristic function is defined as:

$$v(S) = \sum_{t=1}^{T} \hat{q}_S^t(S) \cdot p_S + N \cdot \hat{q}_F(S) \cdot p_F + \kappa(S) \qquad (7)$$

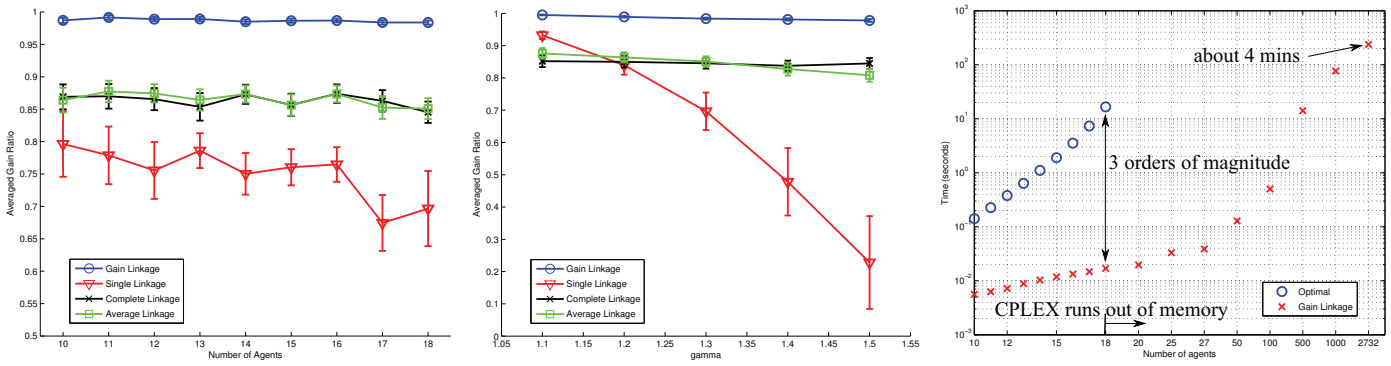
where p_S and p_F represents the unit price of energy in the spot and forward market respectively[5], $\hat{q}_F(S)$ stands for the time unit amount of electricity to buy in the forward market and $\hat{q}_S^t(S)$ for the amount to buy in the spot market at time slot t. These quantities are the ones that optimise the buying strategy of the group while satisfying the group electricity demand:

[2]NDCS ensures that the search process is not biased towards coalitions of smaller sizes (as it is the case with the normal and uniform distributions proposed in [Larson and Sandholm, 2000])

[3]When the error bars do not overlap, the null hypothesis can be validated with $\alpha = 0.05$.

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

[5]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 $p_F = -70$.



(a) (b) (c)

Figure 4: Energy Experiment, results for the Averaged Gain Ratio varying: (a) the number of agents ($\gamma = 1.3$); (b) the parameter γ (18 agents). Figure (c) reports run time (seconds) with the y-axis in logarithmic scale.

$$q_S^t(S) + q_F(S) \geq e_S^t \qquad \forall t = 1 \dots T \tag{8}$$

In other words, to compute the characteristic function for a coalition S 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, $\kappa(S)$ 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 $\kappa(S) = -|S|^\gamma$ 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.

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

As can be seen (see Figure 4(a)), the results confirm the behaviours discussed in Figure 3, but here the averaged gain ratio is significantly higher with gain-link consistently achieving solutions which are very close to the optimal (98% of the optimal in the worst case). Notice that in this domain gain-link does not need to store the values of the characteristic function for all the possible coalitions, as this can be computed using equation 7. Hence, gain-link in this case is definitely the best possible approach among the ones we tested.

To evaluate the sensitivity of the approaches to the γ parameter we fixed the number of agents to 18 and varied γ. Results reported in Figure 4(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 4.2), single-link tends to form big coalitions that get penalized when the γ parameter is increased (see Table 1).

Finally, Figure 4(c) (note the y-axis is in log-scale) reports the run-time for gain-link and the optimal Mixed-Integer programming approach, increasing the number of agents from 10 to 2732 (i.e., the size of the whole data-set). Thus, gain-link is shown to provide solutions for thousands of agents in few minutes (about 4 minutes for 2732 agents). Moreover, the total gain value (not reported here) remains almost constant while increasing the number of agents. This shows that gain-link can provide high quality solutions even when the number of agents increases to thousands.

	Optimal	Gain L.	Single L.	Comp. L.	Avg L.
Coal. Numb.	**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

Table 1: Statistics for coalitions in the energy domain (18 agents, $\gamma = 1.3$). Notice that single-link forms big coalitions, while gain-link forms coalitions that have a very similar structure to the ones formed by the optimal approach

5 Conclusions

In this paper we focus on providing good-enough solutions to the CSG problem. Specifically, we draw the parallels between the CSG problem and data clustering proposing a novel scalable heuristic called C-Link. We compare C-Link against other clustering heuristics and an optimal CSG algorithm. Our experiments show that C-Link outperforms these heuristics and can provide high-quality solutions (at least 80% of the optimal) in both synthetic and real-world applications, solving problems with thousands of agents in few minutes.

When taken together, the analysis of various clustering approaches and our empirical results provide the first benchmarks for large-scale approximate coalition structure generation and open up several promising future directions that include a theoretical study of performance bounds for specific characteristic functions as well as the investigation of other clustering schemes (e.g., partitional clustering).

References

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

[Jain and Dubes, 1988] A. K. Jain and R. Dubes. *Algorithms for clustering data*. Prentice Hall, 1988.

[Larson and Sandholm, 2000] K. S. Larson and T. W. Sandholm. Anytime coalition structure generation: An average case study. *Journal of Experimental and Theoretical AI*, 12:40–47, 2000.

[Rahwan and Jennings, 2008a] T. Rahwan and N. R. Jennings. Coalition structure generation: Dynamic programming meets anytime optimisation. In *Proc 23rd Conference on AI (AAAI)*, pages 156–161, 2008.

[Rahwan and Jennings, 2008b] T. Rahwan and N. R. Jennings. An improved dynamic programming algorithm for coalition structure generation. In *Proc 7th Int Conf on Autonomous Agents and Multi-Agent Systems (AAMAS 08)*, pages 1417–1420, 2008.

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

[Ramchurn *et al.*, 2010] S. D. Ramchurn, M. Polukarov, A. Farinelli, C. Truong, and N. R. Jennings. Coalition formation with spatial and temporal constraints. In *Proc. of 9th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 10)*, pages 1181–1188, 2010.

[Sen and Dutta, 2000] S. Sen and P. S. Dutta. Searching for optimal coalition structures. In *Proc. of 4th Int. Conf. on Multiagent Systems*, pages 287–292, 2000.

[Theodoridis and Koutroumbas, 2008] S. Theodoridis and K. Koutroumbas. *Pattern Recognition, Fourth Edition*. Academic Press, 4th edition, 2008.

[Vinyals *et al.*, 2012] M. Vinyals, F. Bistaffa, A. Farinelli, and A. Rogers. Coalitional energy purchasing in the smart grid. In *Proc. of IEEE Int. Energy Conference and Exhibition (ENERGYCON 12)*, pages 848 –853, 2012.

[Voice *et al.*, 2011] T. Voice, P. Vytelingum, S. Ramchurn, A. Rogers, and N. R. Jennings. Decentralised control of micro-storage in the smart grid. In *AAAI-11: Twenty-Fifth Conference on Artificial Intelligence*, pages 1421–1426, 2011.

[Voice *et al.*, 2012] T. Voice, S. D. Ramchurn, and N. R. Jennings. On coalition formation with sparse synergies. In *Proc. of 11th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 12)*, pages 223–230, 2012.

[Yun Yeh, 1986] D. Yun Yeh. A dynamic programming approach to the complete set partitioning problem. *BIT Numerical Mathematics*, 26(4):467–474, 1986.