Socially Motivated Partial Cooperation in Multi-agent Local Search

Tal Ze’evi and Roie Zivan
Ben-Gurion University of the Negev,
Beer-Sheva, Israel

talzee@post.bgu.ac.il
zivanr@bgu.ac.il

Abstract. Partial Cooperation is a paradigm and a corresponding model that was proposed in order to represent Multi-agent systems in which agents, like standard people in many realistic situations, are willing to cooperate for achieving a global goal, as long as some minimal threshold on their personal utility is satisfied, e.g., a working environment in which people receive salaries. Distributed local search algorithms were proposed in order to solve asymmetric distributed constraint optimization problems (ADCOPs) in which agents are partially cooperative. We contribute to the research on partial cooperative multi-agent local search by: 1) extending the partial cooperative model in order to allow it to represent dynamic cooperation intentions, affected by changes in agents wealth. Such changes in agents intentions correspond to studies on human cooperation intentions in social studies literature. 2) proposing a novel local search algorithm in which agents receive indications on preferences of others, regarding their actions (assignment selections) and thus, can perform actions that are socially beneficial. Our empirical study reveals the advantage of the proposed algorithm in multiple benchmarks.

Keywords:

1 Introduction

Most studies investigating multi-agent systems consider either fully cooperative agents, which are willing to exchange information and take different roles in the process of achieving a common global goal (cf. [1–3]), or self-interested agents, which are considered to be rational when they take actions that will increase their personal gains (cf. [4]). A notable exception to this two class division includes cooperative game theory, which considers cases where self-interested agents will join or form coalitions in which they will fully cooperate [5].

A partially cooperative model that applies to scenarios, which do not fall into any of these two extreme classes, was proposed in [6, 7]. In such scenarios, agents act cooperatively, motivated by a desire to increase global (group) utility or by altruistic incentives, as long as a minimum condition on their personal utility is satisfied. Such scenarios are common in many realistic applications,
e.g., in a working environment where a worker is expected to follow orders of his superiors, as long as he is receiving his salary (for more relevant examples see [7]).

The partial cooperation model represents the willingness of agents to cooperate by defining thresholds of minimum requirements for cooperation, i.e., agents are willing to perform actions that may result in more utility for the group and less utility for themselves, as long as some minimum on their personal utility is preserved. Such willingness of people to perform philanthropic acts (share their wealth) as long as they are beyond the point in which they feel financially insecure has been indicated in social science theory [8].

Previous attempts to design partially cooperative models assumed a fixed reference point according to which the agents determine whether to cooperate or not. This point described the minimum utility requirements threshold of an agent, such that when her utility was above this threshold, the agent performed cooperatively, but, when her personal utility was reduced beyond this point, the agent acted selfishly [6, 7].

However, social science literature also indicates that people have diverse attitudes towards the distribution of personal wealth (monetary or otherwise) [8], and that their behavior, including their intentions for cooperation is affected by changes in their wealth [9]. Thus, the existing partial cooperation model fails to represent the dynamic nature of peoples willingness to cooperate, which is based on the amount of utility they gain or lose during the solving process.

In order to express this dependency between the change in the amount of utility or cost during a distributed search process and the willingness of agents to cooperate, we propose to expand the partial cooperation model proposed in [6, 7], by relaxing the assumption that the cooperation threshold is fixed, and investigate how realistic change patterns affect distributed local search algorithms.

The adjustment of the partial cooperative model to realistic social behavior of humans, allowed us to analyze the results produced by the two local search algorithms proposed in [7] when solving problems that include different types of agents, i.e., agents whose intentions for cooperation change in a different patterns following changes in the amount of their personal utility. This investigation revealed a weakness of these distributed local search algorithms. The agents in these algorithms attempt to find feasible solutions (solutions that satisfy the minimum personal utility requirements of all agents) that maximize their own gains according to their own knowledge. Hence, the willingness of agents to cooperate under some conditions was used only to maintain the validity of the solutions and not to improve their quality. The optimization process was identical to standard distributed local search algorithms as long as the minimum requirements were satisfied. Thus, when agents performed operations that decreased the utility of other agents, they were not aware of the damage they incur, as long as the utility did not drop beneath the minimal requirements. This limited the ability of agents to perform actions that are socially beneficial.

Following these insights regarding the existing partial cooperative distributed local search algorithms, we propose a novel approach towards partial cooperative
local search in which agents indicate to their neighbors which value assignments are preferred by them. These indications allow agents to make socially beneficial selections of value assignments.

Thus, in this paper we advance the research on partial cooperative models of distributed optimization by:

1. Propose an extended model for representing partial cooperative agents. The model includes dynamic points of reference that are updated according to the personal utility gained or lost by the agent during the algorithm run.

2. Propose a socially motivated local search algorithm in which agents share with their neighbors preferences on the assignment selection of the neighbors. These indications help agents select value assignments that are beneficial for other (neighboring) agents as well as for themselves.

Our empirical results demonstrate the advantage of the proposed algorithm over the previously proposed partially cooperative local search algorithms in solving structured, unstructured and realistic DCOPs including agents of different partial cooperative types.

2 Background

2.1 Distributed Constraint Optimization

Without loss of generality, in the rest of this paper we will assume that all problems are minimization problems. Our description of a DCOP is consistent with the definitions in many DCOP studies, e.g., [2].

A DCOP is a tuple \( \langle A, \mathcal{X}, D, R \rangle \). \( A \) is a finite set of agents \( \{ A_1, A_2, \ldots, A_n \} \). \( \mathcal{X} \) is a finite set of variables \( \{ x_1, x_2, \ldots, x_m \} \). Each variable is held by a single agent. \( D \) is a set of domains \( \{ D_1, D_2, \ldots, D_m \} \). Each domain \( D_i \) contains the finite set of values that can be assigned to variable \( x_i \). An assignment of value \( d \in D_i \) to \( x_i \) is denoted by an ordered pair \( \langle x_i, d \rangle \). \( R \) is a set of relations (constraints). Each constraint \( C \in R \) defines a non-negative cost for every possible value combination of a set of variables, and is of the form \( C : D_{i_1} \times D_{i_2} \times \ldots \times D_{i_k} \rightarrow \mathbb{R}^+ \cup \{0\} \). A binary constraint refers to exactly two variables and is of the form \( C_{ij} : D_i \times D_j \rightarrow \mathbb{R}^+ \cup \{0\} \). A binary DCOP is a DCOP in which all constraints are binary. A partial assignment (PA) is a set of value assignments to variables, in which each variable appears at most once. \( \text{vars}(PA) \) is the set of all variables that appear in PA. A constraint \( C \in R \) of the form \( C : D_{i_1} \times D_{i_2} \times \ldots \times D_{i_k} \rightarrow \mathbb{R}^+ \cup \{0\} \) is applicable to PA if \( x_{i_1}, x_{i_2}, \ldots, x_{i_k} \in \text{vars}(PA) \). The cost of a PA is the sum of all applicable constraints to PA over the assignments in PA. A complete assignment (or a solution) is a partial assignment that includes all the DCOP’s variables (\( \text{vars}(PA) = \mathcal{X} \)). An optimal solution is a complete assignment with minimal cost.

---

\(^1\) We say that a variable is involved in a constraint if it is one of the variables the constraint refers to.
For simplicity we make the standard assumptions that all DCOPs are binary DCOPs in which each agent holds exactly one variable. These assumptions are commonly made in DCOP studies, e.g., [2].

2.2 Asymmetric DCOP

ADCOPs generalize DCOPs by explicitly defining for each combination of assignments of constrained agents, the exact cost for each participant in the constraint [10]. That is, domain values are mapped to a tuple of costs, one for each constrained agent and each agent holds only its part of the constraint.

More formally, an ADCOP is defined by the following tuple \( \langle A, X, D, R \rangle \), where \( A \), \( X \) and \( D \) are defined the same as in DCOPs. Each constraint \( C \in R \) of an asymmetric DCOP defines a set of non-negative costs for every possible value combination of a set of variables, and takes the following form: \( C : D_{i_1} \times D_{i_2} \times \cdots D_{i_k} \rightarrow R^k_+ \).

Notice that here \( R^k_+ \) is a vector that includes for each agent \( A_j \), \( 1 \leq j \leq k \) its cost for each combination of value assignments, so in practice each agent \( 1 \leq j \leq k \) holds its part of the constraint \( C_j \), \( C_j : D_{i_1} \times D_{i_2} \times \cdots D_{i_k} \rightarrow R^k_+ \).

2.3 Partial Cooperation

In contrast to the early studies of ADCOPs, which assumed full cooperation by the agents [11, 12], partial cooperation models represent agents that cooperate under some conditions. The level of cooperation (which is represented by \( \lambda \)) determines the reference point according to which agents intentions are modeled. In order to allow the agents to consider solutions with high global quality, which may reduce their personal utility, the parameter \( \lambda \) bounds the losses that an agent is willing to undertake in order to contribute to the global objective, i.e., agents perform actions only if they do not result in a cost that exceeds the maximum cost they are willing to endure. Formally, the following parameters are used by the model:

**Definition 1** We note by \( \mu_i \) the base-line cost of agent \( i \) (i.e., the cost for agent \( i \) that it assumes she can pay when acting selfishly).

**Definition 2** The cooperation intention parameter \( \lambda_i \geq 0 \) defines the maximal increase in the value of \( \mu_i \) which is acceptable by agent \( i \).

These cooperation bounds can significantly decrease the number of feasible outcomes for a distributed incomplete algorithm, as can be seen in the next definition.

**Definition 3** A feasible outcome for a distributed algorithm is defined to be any outcome (solution) \( o \) in the set of all possible outcomes \( O \), that satisfies the following condition.

\[
O^{feasible} = \{ o \in O \mid \forall A_i \in A, \ c_i(o) \leq \mu_i + \lambda_i \}
\]

Where \( c_i(o) \) is the cost for agent \( A_i \) in outcome \( o \).
2.4 Partial Cooperative Local Search

Two distributed local search algorithms were proposed in [7]. Both were based on the Max Gain Messages (MGM) algorithm [13]. In the first, unsatisfied agents (agents whose current costs exceeds the limit) indicate to their neighbors that an assignment replacement is required (by sending a nogood). The best, valid, solution is then maintained with a modified version of a distributed Anytime scheme [14]. In addition to exchanging maximal gain and value assignments, as in standard MGM, agents send nogood messages that signal to their neighbors that the value assignment of the neighbor causes them to exceed the cost limit, and good messages that remove a restriction imposed by a nogood when it does not apply any more. Thus, this algorithm is called Goods-MGM.

The second algorithm, Asymmetric Gain Coordination (AGC), guarantees that the personal cost of an agent does not exceed the predefined cooperative bound, while constantly seeking globally improving solutions. Agents executing this algorithm exploit possible improvements until they converge to some local optima, which can’t be further improved without breaching the cooperation bound of one of the agents. Before replacing a value assignment, an agent requests her neighbors’ approval, which is given only if this value assignment replacement does not cause a breach of the cooperative bound for the neighbor. Only if all neighbors approve, the agent replaces her value assignment.

3 Reference Dependent Partial Cooperation

While the partial cooperative paradigm was designed such that agents determine their intentions for cooperation using a fixed point of reference, behavioral economics literature indicate that peoples’ intentions change with respect to changes in their wealth. This behavioral trait is known as ‘reference-dependence’, and is one of the fundamental principles of prospect theory, and an important factor in explaining people’s attitudes towards risk [9, 8]. Reference-dependent theories indicate that people are more sensitive to changes in wealth rather than to absolute wealth level.

In order to allow the partial cooperative model to represent dynamic reference points, we redefine some of the parameters of the model:

Definition 4 Let $\mu_{i,t}$ be the reference cost of agent $A_i$ at iteration $t$ of the algorithm (where $\mu_{i,0}$ is the baseline cost as defined above).

Definition 5 Let $\lambda_{i,t}$ be the cooperation intention parameter for agent $A_i$ at iteration $t$ of the algorithm, which as before defines the maximal increase in the cost acceptable by agent $A_i$, only with respect to $\mu_{i,t}$.

Definition 6 A complete assignment $S$ is feasible in iteration $t$ if it satisfies the following condition:

$$\forall A_i \in A, c_i(S) \leq \mu_{i,t} \cdot (1 + \lambda_{i,t})$$
The outcome of a distributed algorithm that runs for \( m \) iterations is the complete assignment at the end of the \( m \)'th iteration \((S_m)\), and it is feasible if the definition above holds for \( \mu_{i,t} \) and \( \lambda_{i,t} \).

We next present a number of examples of types of agents that can be represented by the extended model:

- **Type 1:** Fixed reference and cooperation parameters, i.e., for each agent \( A_i \), for each iteration \( t \), \( \mu_{i,t} = \mu_{i,0} \) and \( \lambda_{i,t} = \lambda_{i,0} \). This type of agents is identical to the types described in [7], in which changes in wealth do not change the intentions for cooperation of the agent.

- **Type 2:** Agents that allocate a fixed percentage of their wealth for the benefit of the environment in every interaction, so that for every iteration \( t \): \( \lambda_{i,t} = \lambda_{i,0} \) and \( \mu_{i,t} = c_i(S) \), i.e., the reference point is the current cost of the agent.\(^2\)

- **Type 3:** Agents that separate their expenses from their income, by pre-determining an expendable budget to be used for all the costs involved in their interactions with the environment. When actions produce a negative balance the budget is gradually reduced. Formally this behavior is represented by a fixed \( \lambda \), i.e., \( \lambda_{i,t} = \lambda_{i,0} \) and a calculation of \( \mu \) in each iteration as follows:

\[
\mu_{i,t} = \min\{\mu_{i,t-1}, \mu_{i,t-1} + \frac{c(S_t) - c(S_{t-1})}{\lambda_{i,0}}\}
\]

In Section 5, we demonstrate the impact of the different behavior patterns on the social welfare in different scenarios.

4 Socially-Motivated Local Search

In the partial cooperative local search algorithms described above, agents, besides exploiting their local knowledge, cooperate in order to preserve a level of personal utility that is acceptable by all agents.

In order to allow agents to exploit the cooperative intentions of their neighboring agents, in order to improve solution quality (social welfare), we propose a novel approach towards partial cooperative local search, in which agents take an extra step in the interaction process, prior to selecting an assignment, during which each agent shares with her neighbors, some information regarding the preferences she has over their assignment selection, i.e., an indication of her anticipated benefits, or costs, if they decide to change their current value assignment. After exchanging such indications, each agent attempts to find a value assignment while taking into consideration her own preferences and the indications received regarding her neighbors’ preferences. We combine this approach with the AGC algorithm (cf. [7]) and propose Socially Motivated (SM) AGC.

\(^2\) The calculation for minimization problems is a bit more complicated, we omit the details in order to avoid confusion.
SM_AGCG

input: baseLineAssignment, baseLineCost, λ, and Ω.

value ← baseLineAssignment;
µ₀ ← baseLineCost;
localView ← null;
send(value) to N(i);
while stop condition not met do

PHASE 1:
Collect all value messages and update localView
for each A_j ∈ N(i) do
πᵢ,j ← preferences(A_j);
send(πᵢ,j) to A_j;

PHASE 2:
Collect all π messages;
Πᵢ ← π_j ∈ N(i) ∪ preferences(A_i);
socialImprovingAssignment(Π_i, Ω_i) to N(i);

PHASE 3:
Collect all ⟨alterVal_j, socialGain_j⟩ messages;
a_j ← agent in N(i) ∪ A_i with maximal socialGain s.t.
c_i(v_j ← alterVal_j|S_i) ≤ µ_i,t · (1 + λ_i,t);
send(Neg!) to N(i) \ a_j;

PHASE 4:
Collect Neg! messages;
if did not receive Neg! & can improve then
value ← alterVal_i;
send(value) to N(i);

Fig. 1. SM_AGCG

Definition 7 Let ωᵢ,j ∈ Ωᵢ be the importance that agent i ascribes to agent A_j’s preferences.

s.t.: 0 ≤ ωᵢ,j ≤ 1, and \[ \sum_{j \in N(i) \cup \{A_i\}} \omegaᵢ,j = 1 \]

Where N(i) is the set of agent A_i’s neighbors.

When ωᵢ,i = 0, agent A_i will be completely altruistic, selecting its value assignment in accordance with the preferences of its neighbors and ignoring her personal interest. In contrary, when ωᵢ,i = 1, A_i completely ignores the preferences of her neighbors and chooses her value assignment taking into consideration only her own interest.

The pseudo code of the socially-motivated AGC algorithm is presented in Figure 1. Like the original AGC version, the algorithm begins after agents computed a baseline assignment by performing a simple non cooperative interaction between them. Thus, the agent can select its baseline value assignment and use the baseline cost as a reference point. After exchanging their value assignments
the agents loop over the four phases of the algorithm until a termination condition is met, e.g., a predefined number of iterations.

In Phase 1, each agent, after receiving the value assignments from her neighbors, sends to each of them an indication regarding her preferences on their value assignment selection. In Phase 2, after receiving preferences indications, each agent attempts to find an alternative social improving value assignment, i.e., selects a value while taking into consideration her own preferences and the indications regarding the neighbors’ preferences. Different heuristics for selecting such an alternative social improving value assignment can be used. After selecting the alternative value, the agent sends to her neighbors the calculated expected social gain. In Phase 3 agents collect social gain messages from their neighbors, and select a neighbor that her gain was the highest among the neighbors that proposed alternative assignments, which did not violate the receiver’s current cooperation threshold. Then, Neg! messages are sent to all neighbors but this selected neighbor. If their is no such neighbor, then Neg! messages are sent to all neighbors. In Phase 4 agents that did not receive any Neg! messages and can improve their cost by replacing their value assignments, replace and inform their neighbors.

4.1 Socially Motivated Assignment Selection Heuristic

The selection of the alternative value assignment (alterVal) in Phase 2 is different than the selection in standard local search algorithms, e.g., DSA and MGM, since it takes into consideration besides the agent’s own preferences, the indications received on the preferences of its neighbors and attempts to select a social improving assignment.

We propose (and use in our experiments) the following heuristic for this selection: After collecting the indications \( \pi_j \) received from its neighbors, the agent calculates a sampling probability for each value in its domain, as follows:

\[
p_{val} = \begin{cases} 
\frac{\Gamma_{val}}{\sum_{val \in \text{Domain}_i , \Gamma_{val} \geq 0} \Gamma_{val}} & , \Gamma_{val} < 0 \\
0 & , \text{otherwise}
\end{cases}
\]

Where:

\[
\Gamma_{val} = \sum_{j \in N(i) \cup \{A_i\}} \omega_{i,j} \cdot \pi_j(val)
\]

Afterwards, the agent randomly samples a value from the underlying distribution.

4.2 Preference Indications

The main novelty of the proposed algorithm is in the sharing of indications regarding preferences on the selection assignment of neighboring agents. Thus, the information that agents share in this stage of the algorithm (Phase 1) is
expected to have a dramatic effect on the performance of the algorithm. We propose five versions of the algorithm, which we compare in our experimental study.

We make a distinction between two categories of indications that agents share with their neighbors. The first, we call ‘taboo’ assignments, i.e., an agent informs her neighbor which of the neighbors’ value selections will cause a breach of the current cooperation threshold. The second, which we call a ‘vote’ allows agents to direct their neighbors to a specific value that they wish the neighbor will select. Such a vote can be binary or weighted, i.e., it can include an indication that the neighbor agent is favor of this value selection or include the actual benefit that the agent expects to gain from this selection. Table 1 summarizes the differences between the various versions of the SmAGC algorithm that will be further discussed.

### Table 1. Different Versions of the SM\_AGC algorithm

<table>
<thead>
<tr>
<th>Version</th>
<th>Taboo</th>
<th>Vote</th>
</tr>
</thead>
<tbody>
<tr>
<td>SmAGC_BI</td>
<td>-</td>
<td>Binary</td>
</tr>
<tr>
<td>SmAGC_CI</td>
<td>-</td>
<td>Cost</td>
</tr>
<tr>
<td>SmAGC_T</td>
<td>+</td>
<td>None</td>
</tr>
<tr>
<td>SmAGC_T_BI</td>
<td>+</td>
<td>Binary</td>
</tr>
<tr>
<td>SmAGC_T_CI</td>
<td>+</td>
<td>Cost</td>
</tr>
</tbody>
</table>

5 Experimental Evaluation

Several experiments were conducted in order to evaluate the performance of the different versions of the algorithm in scenarios including agents with different types of dynamic cooperation intention updates. Four types of problems were considered: (1) uniform random problems. (2) K-regular graphs. (3) Scale Free problems and (4) Graph-Coloring problems.

All problems included in the experiments were binary minimization DCOPs, with 100 agents (\(n = 100\)), each holding a single variable with 10 values in its domain (\(d = 10\)), with the exception of Graph-Coloring problems, in which domains included 3 colors. All agents were given the same initial cooperation value, i.e., \(\lambda_{i,0} = \lambda_{j,0}, \forall A_i, A_j \in \mathcal{A}\). Each algorithm ran for 1000 iterations on each problem. Results were averaged over 50 random instances.

In all SM\_AGC versions agents ascribe equal importance to their neighbors’ votes and taboo values had zero sampling probability.

**Setup 1 - Random uniform DCOPs:** unstructured problems in which the network of constraints was generated randomly by adding an asymmetric constraint for each pair of agents independently with probability \(p_1 = 0.1\).\(^3\) The cost of a joint assignment by two constrained agents was set to 0 with a probability of 0.5 or randomly sampled from a discrete uniform distribution in the

---

\(^3\) Similar results were obtained for problems with \(p_1 = 0.7\) and were omitted for lack of space.
range \([0, 100]\). Figures 2 and 3 presents the aggregated solution cost, in each iteration of the search procedure when the initial cooperation parameter is set to \(\lambda = 0.1\), and \(\lambda = 0.8\) respectively. In both settings, for all types of agents, both, Goods-MGM and AGC achieved inferior social welfare than the various versions of SM_AGCA. Goods-MGM failed to find any feasible solutions for smaller values of \(\lambda\). Versions in which agents share 'taboo' indications, found better solutions for \(\lambda = 0.1\), however for \(\lambda = 0.8\), the advantage was less significant. Under both settings, agents of type 3 were first to converge to solutions with low social welfare across all algorithms. In contrary, agents of type 1, continued to explore the solution space for better solutions.

![Fig. 2. Solution cost for uniform random problems (\(\lambda = 0.1\)).](image)

![Fig. 3. Solution cost for uniform random problems (\(\lambda = 0.8\)).](image)

**Setup 2 - K-regular graphs:** randomly generated graphs in which all agents had the same number of neighbors (5). The cost of a joint assignment by two
constrained agents was randomly sampled from a discrete uniform distribution in the range [0,100]. Figures 4 and 5 present the aggregated solution cost, when the cooperation parameter is set to \( \lambda = 0.1 \) and \( \lambda = 0.8 \) respectively. Similarly to the previous setup, in both settings, for all types of agents, Goods-MGM and AGC found solutions with higher cost than the SM AGC versions. However, in this setup the version of SM AGC in which agents do not share ‘votes’ (only ‘taboo’) dominated. The AGC algorithm, presents its best performance for agents of type 1 for lower values of \( \lambda \) and for agents of type 2 for higher values of \( \lambda \). As before, the differences in the performances between the different types of agents, and between the different algorithms decrease with the growth of \( \lambda \).

**Fig. 4.** Solution cost for regular graphs (\( \lambda = 0.1 \)).

**Fig. 5.** Solution cost for regular graphs (\( \lambda = 0.1 \)).

**Setup 3 - Scale Free networks:** were constructed by using the Barabasi-Albert model. An initial set of 20 agents was randomly selected and connected.
At each iteration of the BA procedure an agent was added and connected to 3 other agents with a probability proportional to the number of links that the agents already have. Figure 6 presents results for scenarios in which $\lambda = 0.1$. The versions including 'taboo' exchange, found solutions with lower costs for all types of agents. Interestingly in problems with agents of type 2, binary voting was more beneficial than weighted voting.

![Fig. 6. Solution cost for Scale Free networks ($\lambda = 0.1$).](image)

**Setup 4 - Graph-Coloring problems:** The asymmetric graph coloring problems included 3 colors in each domain and a density parameter $p_1 = 0.05$. Figure 7 presents the solution cost, in each iteration for initial cooperation parameter $\lambda = 0.1$. In contrast to the previous setups, Goods-MGM and standard AGC achieved results which are competitive with some versions of SM AGC. While the version of SM AGC in which agents share only 'taboo', found solutions with lower costs for problems with agents of type 3, it found solutions high costs for problems with agents of type 1.

**5.1 Discussion**

The existing distributed local search algorithms, Goods-MGM and AGC, include sharing of information (goods and nogoods) regarding assignments that cause solutions to be unacceptable to agents. The versions of SM AGC we propose include two types of shared information. The first follows the former trend and shares indications regarding value assignments that will cause the solution to become unacceptable, i.e., 'taboo' indications are actually 'nogood's. The second are positive indications ('vote's) that direct the search towards high quality acceptable solutions. These were not included in previous algorithms and depend on the willingness of the agents to reveal private information. Nevertheless, our results indicate that the version of SM AGC, which does not reveal such information (only 'taboo' indications) produces solutions with significantly less cost.
than standard AGC and Goods-MGM, thus, even in scenarios where agents are reluctant to share votes, SM_AGCA has a drastic impact.

6 Conclusion

The Partial Cooperative paradigm allows simulation of realistic distributed scenarios in which agents are willing to cooperate as long as some minimum on their personal utility is preserved. The realistic motivation of this algorithmic framework makes it essential to allow the representation of realistic scenarios, where agents have dynamic cooperation intentions. We presented an extension of the existing partial cooperative model that allows the representation of such scenarios.

We further proposed a local search algorithm in which the cooperative intentions of agents can be exploited, not only for making sure that the solution obtained is acceptable by all agents (as done in existing partial cooperative algorithms), but also, in order to select a high quality solution. Our empirical results show the advantage of this approach in different scenarios and when agents have different levels of intentions for sharing information. A significant advantage of the proposed algorithm over the existing partial cooperative algorithms was found even when only insatiability indications (‘taboo’) were shared.

In future work we intend to extend the study of incomplete partial cooperative distributed algorithms to inference algorithms.

References


