

A Multi-Agent Method for Forming and Dynamic Restructuring of Pareto Optimal Coalitions

Philippe Caillou
LAMSADE, Université Paris 9
Place du Maréchal De Lattre De
Tassigny
75775 Paris, France
caillou@lamsade.dauphine.fr

Samir Aknine
LIP6, Université Paris 6
8, rue du Capitaine Scott
75015 PARIS, France
Samir.Aknine@lip6.fr

Suzanne Pinson
LAMSADE, Université Paris 9
Place du Maréchal De Lattre De
Tassigny
75775 Paris, France
pinson@lamsade.dauphine.fr

ABSTRACT

The first part of this paper presents a coalition formation method for multi-agent systems which finds a Pareto optimal solution without aggregating the preferences of the agents. This protocol is adapted to problems requiring coordination by coalition formation, where it is undesirable, or not possible, to aggregate the preferences of the agents. The second part proposes an extension of this method enabling dynamic restructuring of coalitions when changes occur in the system.

Categories and Subject Descriptors

I.2.11. [Artificial Intelligence]: Distributed Artificial Intelligence – *Coherence and coordination*

General Terms: Algorithms, Economics, Experimentation

Keywords: Coalition formation, dynamic restructuring, Pareto optimum, class scheduling,

1. INTRODUCTION

The search for economic efficiency has led to the division of labor between specialists. Today, similar reasoning explains the success of multi-agent systems. Using a set of specialized agents which coordinate their complex tasks gives more flexibility, efficiency and necessary evolutivity to programs. To perform complex tasks, agents need to coordinate, either because tasks require many resources if they are to be performed by a single agent, or because certain sub-tasks can be carried out more efficiently by specialized agents [2] [7].

How can autonomous agent be coordinated efficiently? One solution is to look for groups of agents which are able to perform the desired tasks better. This means that agents may form coalitions. A coalition is defined as a temporary association between agents in order to carry out joint projects. The aim is a better distribution of competences in order to achieve a complex

project. This is not the only method of coordination. The choice of making coalitions depends on the type of problem. Coalitions are well adapted when there are strong externalities and/or when interactions between agents are such that the contribution of an agent within a coalition depends on which agents a coalition contains, in which case a bilateral contract would be difficult to negotiate.

Once coalition formation is chosen as a coordination method, the definition of the corresponding protocol remains problematic. A coalition formation protocol is strongly dependent on the type of problem studied. The fact that the agents do or do not have the same objective, do or do not trust in others, are examples of parameters which may generate completely different protocols. To enable the agents to form coalitions, all current protocols make the assumption that the utility functions of agents, which measure their degree of satisfaction for each suggested solution, must be comparable or identical. This means that agents must be able to agree on a common utility function, either of all the agents as in [11], or of their coalition as in [1] and [13]. This assumption seems acceptable for most multi-agent systems, in particular for productive projects where all utilities can often be calculated in terms of profit. However, in many cases comparing the utilities of agents, and even more so their aggregation, is delicate. The numerical measurement of the utility of an agent is already a strong assumption compared with the simple classification of available choices. Comparing the utilities of two individuals is stronger. Why should a solution weighted 8 by one agent and 6 by another be preferred to one weighted respectively 4 and 7? Our model does not suppose that the utilities of agents must be aggregated or compared.

A second limitation of current models lies in assuming that all calculations are recomputed as one condition changes (an agent joins or leaves a coalition, a task is added or canceled, etc.). However these protocols are very complex and these changes can be very frequent. Using the information obtained in a previous execution of the protocol, i.e. a dynamic reorganization of the coalitions formed, could decrease calculations. This is the second aim of our model.

This article is organized as follows. Section 2 describes the application of the protocol. Section 3 presents some definitions. Section 4 details our method for coalition formation and dynamic reorganization of coalitions. Our model gives a wide choice of agent behavior. Section 5 proposes a set of behavior models in order to improve the computational complexity. Section 6 presents an application example of the protocol and discusses the

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

AAMAS '02, July 15-19, 2002, Bologna, Italy.

Copyright 2002 ACM 1-58113-480-0/02/0007...\$5.00.

implementation of our model. Section 7 analyzes related work. Section 8 draws a general conclusion from this work and proposes some perspectives.

2. APPLICATION

The suggested protocol is particularly suitable for problems with complex tasks (where there is a need for several agents and for coalitions) and for dynamic problems (tasks may be added, others canceled or modified constantly) with different utility functions of agents. We assume that agents are cooperative, i.e. they trust each other in searching for and applying the solutions. Their utility functions are unknown by the other agents and do not need to be cardinal, an ordinal utility is enough. Agents just need to be able to choose between two situations (or to be indifferent). A good example of this problem is a distributed teaching schedule at university. This application illustrates the dynamic evolution of the coalitions, as often a course may be added or removed, or a professor or a group of students may join the establishment. In this example we consider two types of agents: professors and students. Student agents represent homogeneous groups of students with a common utility function and with the same classes. Of course, it is possible to have an agent for each student and thus to enable him or her to have their own utility function. However, the computational complexity will be greatly increased.

The classes correspond to the tasks to be carried out. Thus, agents form a coalition for each class. Most coalitions are formed of two agents: a professor agent and a student agent (having more agents in a coalition is also possible, for instance for lectures with several groups of students). Each (student or professor) agent defines the utility it assigns to each schedule. Since its utility function is ordinal, it just needs to be able to compare two schedules and to say which it prefers or that it is indifferent. Agents are free to choose their parameters while computing their utility. A professor can thus prefer the morning, refuse Monday, prefer certain classes, like a stable schedule, etc. In a general way, the choice of an agent depends on the members of the coalitions in which it will take part. But its appreciation of a coalition may also vary according to the other coalitions. This introduces externalities or an ordered processing of tasks. Thus, if a task T_i must be carried out before T_j , the utility that the agent will associate with T_j will be null if no agent takes part in the coalition which performs T_i (task T_i is then not performed, and T_j has no interests). The agent choice may also depend on parameters which are related to its preferences and which vary in time. Thus, it can be against change. For instance, a professor may prefer one schedule to another because it is closer to the current situation. The only constraint is that these external parameters need to be stable during a negotiation step.

3. DEFINITIONS

Coalition: a coalition is formed for each task. It contains zero, one or more agents which will carry out actions in order to achieve a task. Each action and its parameters are defined (for instance, the parameters of the action "taking a class" are: the week, the day and the time).

Coalition set: a coalition set represents a solution to the problem of coalition formation. It contains as many coalitions as tasks to be performed at a given moment (in our application, a set corresponds to one schedule).

Group of coalition sets: a group of coalition sets corresponds to several sets of coalitions brought together in order to be computed and transmitted collectively (for instance, several possible schedules). In the rest of this article, it will be referred to as a group of sets or simply a group. When an agent computes a group of equivalent sets, this means that it is indifferent regarding all the sets of coalitions in this group (for instance, it computes a group with those schedules that it prefers to others and that it cannot classify).

Context: a set of unspecified parameters which must be stable during a negotiation step. For instance, it may concern a date or any external parameter.

Utility function: the utility function may be ordinal or cardinal. If it is cardinal, it associates a utility with a set of coalitions within a given context. If it is ordinal, it compares two sets in a given context. In this case, measuring the utility of a set means comparing it with a reference situation which will be the same one throughout the negotiation.

Reference situation: so that the agents know if they have to accept a set of coalitions as a solution, they need to be able to compare it with what they are sure to obtain during the negotiation. This minimum is the reference situation. If no coalition has yet been formed, the reference situation is the situation where nobody does anything. If there are already coalitions, it is the current situation, with possibly some changes in order to take into account new information (cf. section 4.5.2). To be sure to find a solution after a negotiation, the reference situation needs to be feasible and to be the same for all the agents (a demonstration is proposed in section 4.4.3).

Acceptable set: we consider that a set is acceptable for an agent if it is preferred or equivalent to the current reference situation.

Pareto optimum: a Pareto optimum is a situation where it is not possible to improve the situation of an agent without deteriorating that of at least one other.

4. COORDINATION METHOD

4.1.Presentation

As we do not intend to aggregate the utilities of the agents, we seek a solution which is "objectively good", i.e. which may not be contested by any agent. A logical criterion likely to be accepted by all the agents is that we cannot increase the utility of an individual without deteriorating that of at least one other. If this does not happen, i.e. there is a situation such that we can increase the utility of an individual without deteriorating that of another, there is no reason not to prefer this situation. The solution must thus be a Pareto optimum.

Which Pareto optimum should we choose? Now the problem is to compare the utilities of different agents. How should we choose between a schedule which is the first choice of a professor and another which is the first choice of a student? One solution is to avoid making a choice but to find a Pareto optimum. This offers the advantage of reducing computations as agents are not obliged to compute them all. The only constraint is that each agent should find it interests in accepting this solution, therefore in preferring this solution to the initial situation. The first aim of our protocol is

thus to find a Pareto optimum likely to be accepted by all the agents as early as possible.

How is a Pareto optimum obtained? The agent which initiates a negotiation seeks one or more sets of coalitions it prefers (cf. section 5) and chooses an agent to which it sends them (cf. section 4.2.1). Then it seeks the set(s) that it would choose as a second choice and sends them to that agent, and so on, until there are no more sets at least equivalent to the current situation. When an agent receives a group of sets, it evaluates them and sends them to the next agent sorted in decreasing order of preference. When an agent receives a group of sets, if there is at least one set which is preferable or equivalent to the current situation and if all the agents have already taken part in the negotiation, the set of this group that it prefers is a Pareto optimum and may be used as a solution set for the negotiation. For instance, let us consider two agents and seven sets of possible coalitions. Let $E(U_1;U_2)$ be the relative utilities of agents 1 and 2 for the set E . Having E_0 as the initial situation, the seven possible sets are: $E_0(0;0)$; $E_1(0;10)$; $E_2(2;8)$; $E_3(4;8)$; $E_4(4;5)$; $E_5(-2;2)$; $E_6(10;-1)$ (cf. figure 1). Of these seven sets, three are Pareto optima (E_1 , E_3 and E_6).

Agent 1 initiates the negotiation. It sorts all the acceptable sets for it in equivalent groups of sets (cf. figure 2): $G_1(E_6)$; $G_2(E_4;E_3)$; $G_3(E_2)$; $G_4(E_0;E_1)$. E_5 is not sorted as the reference situation (E_0) is better.

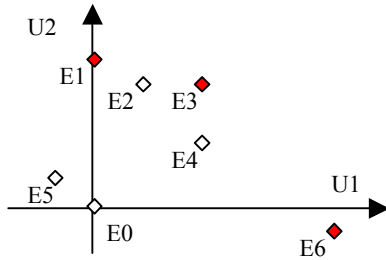


Figure 1.
Describing all possible solutions in a utility space

Groups G_1 , G_2 , G_3 and G_4 are sent in this order to the next agent. Thus, agent 2 starts by receiving G_1 and evaluates it. Set E_6 is unacceptable for the agent because it would bring a less satisfactory situation than the initial situation (figure 3). The agent does not send this set and waits for the rest. It receives G_2 and sorts it into two sets (figure 4) in two groups $G'1(E_3)$ and $G'2(E_4)$. $G'1$ is acceptable.

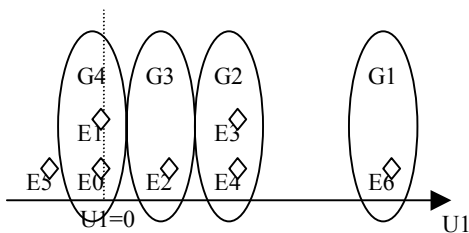


Figure 2. Group of sets of agent 1

As all the other agents have already participated in the negotiation, agent 2 cannot send it. All the sets of $G'1$ can thus be a solution. The agent must choose E_3 , which is Pareto optimal. It sends this set to agent 1 in order to inform it of the result of the negotiation.

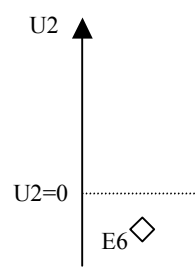


Figure 3. First group of agent 2

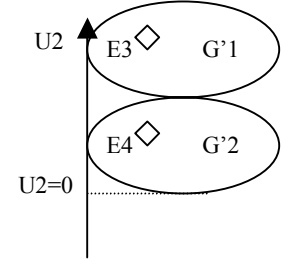


Figure 4. Second group of agent 2

4.2. Algorithm

The negotiation process is based on three phases: initialization of the negotiation and transfer of tasks, negotiation, transmission of the solution. We can distinguish the behavior of the agent which initiates the negotiation from the intermediate and final agents which take part in the negotiation. The order of the agents can differ from one negotiation to another. Each agent can be in any position. However, the order must be stable during a given negotiation.

- Phase 1. Initialization of the negotiation and transfer of tasks. Any agent can initiate the negotiation. This action can result when a new task appears or when an agent modifies its preferences. The initiator agent informs the others that it begins a new negotiation and any agent which wants to begin another negotiation, must wait until the end of the negotiation in progress. To avoid conflicts between two simultaneous requests, each agent sends a confirmation. Each agent asks the other agents to send it their tasks. It then deduces the set of tasks to be performed and associates each one with a coalition. The initiator agent computes all possible sets of coalitions (cf. sections 4.3. and 4.6 regarding complexity), gathers them in a group of sets and sends this group to the agent which must initiate the negotiation.

- Phase 2. Negotiation. When an agent receives a group of sets, it sorts the sets in order of preference into homogeneous new groups of sets. In these groups, all sets are equivalent in terms of agent utility. The agent sorts only these sets that are least equivalent to the reference situation. The others are not considered.

If the agent is not the last agent, it sends its new groups to the next agent in decreasing order of preference. If it is the last agent, and if this agent has created new groups because it has found acceptable sets, it considers that all the sets of the best new group are Pareto optima. It can thus choose one of them randomly and it will be the solution for the negotiation

- Phase 3: Transmission of the solution. Once the last agent has identified a Pareto optimal solution, it sends this set to the other agents which accept it as the solution for the negotiation.

4.3. Using undeveloped coalitions to improve the complexity of the algorithm

As the first agent starts by computing all the possible combinations for all the tasks, this augments the computational complexity of our model and the volume of data sent to the

following agents. A way to improve the computation complexity without decreasing information quality, and thus the result and the properties of the algorithm, is to use and transmit undeveloped coalitions, i.e. the tasks for which all possible coalitions have not yet been computed. If an agent receives an undeveloped coalition in a set and this coalition cannot affect its utility (if it joins it or not), it leaves it aside and does not compute it. If it can, it computes all the possible combinations for the corresponding task. Considering our assumption, the result of this computation is the same whatever the agent which does it. For instance, in drawing up schedules, a professor agent which begins the negotiation will only develop coalitions related to classes it is likely to give, because they are the only ones which can modify its utility. This method produces better results if the agents are affected only by a few tasks. However, if an agent can be affected by any task, it will be asked to develop all possible sets when it takes part in a negotiation.

In our protocol, only a few changes are necessary in order to use undeveloped coalitions. At the end of the first phase, the initiator agent sends to the agent which will begin the negotiation a group of sets containing one set of undeveloped coalitions. In the second phase, when the agent receives a group, for each set of this group and for each undeveloped coalition of this set the agent checks if the corresponding task can affect its utility. If so, it computes all possible coalitions corresponding to this task, adds all the new sets of coalitions to the sets it must compute and removes the set which contained the undeveloped coalition.

4.4. Formal analysis of our model

The order in which agents negotiate influences the result. The first agent is favored in choosing the solution as it is the first to choose the sets it prefers from among all the possible sets, and to send them to the following agents. The choice of the next negotiator agent has thus a great impact. The first solution is to choose randomly among those which have not yet participated in the negotiation. However, to improve the complexity of the protocol, it is preferable to take the agent which appears the most often in the computed sets. We assume that, since it takes part in many coalitions, this agent will be more interested in the alternatives which will be proposed than will be an agent which is less involved. It will thus sort the sets into several groups (it will possibly reject many if it considers them to be unacceptable). The next negotiator agent will receive smaller groups that it will be able to compute more quickly. The second solution consists in choosing the agents in a predefined order. This makes it possible to favor agents which have priority. This solution is very practical in many real applications, such as in drawing up schedules, where professors have priority over their students.

4.4.1. Why is the solution Pareto optimal?

How can we be sure that the first set received by the last negotiator agent is Pareto optimal, as not all possible sets have been evaluated yet by all the agents? A demonstration is thus necessary.

Proposition 1:

When an agent receives a group of sets, if:

-all other agents have already participated in the negotiation,

-at least one of the received sets is acceptable, i.e. it is at least equivalent to the reference situation,

-none of the sets previously received during the negotiation satisfies the two conditions below,

the acceptable set(s) S that it prefers in the received group is/are Pareto optimal and can be used as a solution for the negotiation.

If S is not Pareto optimal, it would mean that there is a set S' which is preferable for one of the agents that shall be called A . S' is at least equivalent to S for all the other agents. In this case, all the agents which were before A in the negotiation have transmitted S' either in the same group as S (if they are indifferent), or in a previous group (if there is at least one agent which prefers S' to S). If A receives S' before S , it had necessarily sent it (since S is a solution, S is acceptable for all, therefore S' , which is at least equivalent to S , is also acceptable for all). As the groups are computed completely before starting with the next group, A returns S' before computing S . The following agents should thus receive S' before S . Since for them S' is also acceptable, they send it to the following agent, and so on until the last one which will therefore find it acceptable and thus select it as a solution, which is impossible since S has been selected. If A receives S and S' in the same group (all previous agents have considered S equivalent to S'), A should send S' before S as it prefers S' to S . As in the previous case, agents following A should receive S' before S . Since S' is acceptable, they should then send it to the next negotiator agent, until the last agent which should also find it acceptable and should therefore select it as a solution for the negotiation, which is also impossible since S has been selected. Consequently, it is impossible for a set S' to exist such that an agent prefers S' to S and that all the other agents find it at least equivalent to S . Therefore S is Pareto optimal.

4.4.2. Why are agents sure to find a solution?

The first optimum S found is the first set which is received by the last negotiator agent and that this agent considers acceptable. Why is there always an optimum?

Proposition 2:

The protocol always provides at least one solution to the problem.

For each agent, the acceptability criterion is that the set is at least as satisfactory as the reference situation. However this reference situation is the same for all and belongs to the possible sets. Therefore all the agents necessarily find this situation acceptable and will forward it. Thus, there will always be at least one acceptable set which will reach the last negotiator agent. If the reference situation is the first set to arrive, it is an optimum and also the solution for the negotiation. If another acceptable set had arrived first, this one would provide the solution.

4.5. Dynamic restructuring of coalitions

Our protocol provides a solution, i.e. a set of coalitions with the initial conditions (utility functions, a set of tasks and a context). What happens if a change occurs in one of these conditions, for instance if a task is added or removed, or if an agent modifies its utility function? In current protocols ([11][10][1]), all the

calculations must be redone to find a new solution to the problem. It seems a pity not to use the results obtained in the current situation. New information should be added to the previous conditions, it should not completely replace them. A simple means to use earlier calculations is to start from the current solution. Instead of evaluating the different sets compared to an initial situation where no agent does anything, the agents will evaluate the new solutions compared to the current solution. As the solution is at least equivalent to the initial situation for all the agents (since it is Pareto optimal), it is difficult to find a similar or better one. Thus, fewer sets and groups of sets will be forwarded and evaluated. This will accelerate the problem solving process.

The new reference situation must remain feasible and identical for all agents in spite of the new information. Thus it is not the current situation which is used as the reference situation but the modified current situation, in which all the changes have been taken into account. For instance, for an agent which leaves, the reference situation will be the current set of coalitions without this agent. For a removed task, it will be the current set of coalitions minus the coalition corresponding to the task.

4.5.1. Why is the solution Pareto optimal?

The demonstration of the first proposition (cf. section 4.4.2) is still valid: when the last agent receives a group of sets, if it has not yet received an acceptable set and the best set of the received group is acceptable, set S is a solution for the negotiation. Moreover, there is no other set which is at least equivalent to S , for all the agents, and preferable for at least one of them, otherwise the last agent would have received it before receiving S and this set would have been selected as a solution.

4.5.2. Why do the agents always find a solution?

The demonstration of the second proposition (cf. section 4.4.3) is still valid: the reference situation is the same for all, acceptable by all (as it is compared with itself) and also feasible. Thus, there is at least one set (the reference situation) which will be sent by each agent to the next agent and which will be the solution if it is the first to be received by the last agent.

4.6. Behavior models of the agents

The aim of each agent is to build new groups of homogeneous coalitions from a group of sets received from the previous negotiator agent and to classify these new groups in order of preferences. Heuristics can be used to find the best group according to the context and the application. This improves the complexity of the algorithms.

The simplest solution is that the first negotiator agent computes all the possible sets and then each agent makes an exhaustive classification of all the possible sets. The advantage of this solution is that it is simple, but it leads to a higher complexity, especially for the first agent. Other search methods can serve to improve the computation complexity and to distribute the calculations among the various agents.

- *Using undeveloped coalitions.* The proposed method using undeveloped coalitions (presented in section 4.3) reduces the calculations and the volume of the information transferred while preserving the ease of calculation by the agents.

- *Tests of intermediate acceptability.* A complementary solution in order to reduce the number of iterations is to test if an (incompletely developed) set can be potentially preferred to the reference situation. If this is not the case, it will not be necessary to develop it. Thus this branch of the exploration tree can be forgotten. These tests are especially useful during the restructuring of coalitions. The reference situation is then the current situation that is likely to be very satisfactory for the agent. This agent can easily set aside many sets which will not give a better solution, especially if the agent does not prefer changes in its situation.

- *Search limited to the best group.* The aim of an agent is to send to the next negotiator agent groups of sets organized in decreasing order of satisfaction. If the solution is in group G_i , all groups G_j with $j > i$ have been evaluated, classified and probably developed unnecessarily. It would be useful to only evaluate the sets of G_1 , then those of G_2 , and so on. The problem is that agents do not know in advance what will be the degree of satisfaction associated with the best group. However, in order to evaluate only the members of G_1 , it is necessary to know the satisfaction which is associated with them, and therefore to have already evaluated them! Even if it is impossible to compute just the sets of the group, we can try to gradually limit computations to the useful sets. To do so, the agent needs a lower limit, which is the best evaluated set at this moment, and it will only develop the sets which are at least equal to this limit. Each time a set, even an incompletely developed one, is evaluated and is higher than the limit (in at least one of its future developments), it becomes the new limit. On the contrary, when an evaluated set cannot reach the limit, but can nevertheless be acceptable in a weaker group, it is preserved and added to a group which will be used as a starting group to compute the following groups.

- *Limited search using intermediate evaluation.* In the previous case, the order in which the coalitions are developed is of great importance. The faster the best set is reached, the faster it becomes the reference situation and the less the other sets are developed (because the reference situation becomes rarely approachable). It is thus useful to set up an intermediate evaluation procedure of the sets to be developed in order to compute first of all the set which seems most likely to generate sets bringing great satisfaction.

- *Prospective search.* In order better to use the utility function, instead of starting from an empty set and developing it, an agent can immediately use its knowledge of its utility function and the tasks to be achieved in order to deduce the best sets. If the number of possible sets is high, this solution can be advantageous since the complexity does not depend here on the number of possible sets but on the type of utility function of the agent. This method can give far more effective results but the procedure for each type of utility function needs to be rewritten.

5. IMPLEMENTATION RESULTS

To illustrate our model, we have implemented a teaching scheduling application system using the utility function of the professors and the students. Since the utility function of the agents is completely free, many parameters have been introduced here so that professors and students can give precise indications of their preferences. The utility function returns a complete result.

However, this result is related to the reference situation, the utility of which (and it is absolute) is computed at the beginning of each negotiation. The utility function has different variables: for each day the time of the first class, the time of the last class, the number of hours per day, the number of classes not given, the number of compulsory classes not given for each agent, the number of changes compared to the current schedule, the total number of hours per week. For each agent modifiable parameters are available and can be used as a utility function. Given the number of parameters, three profiles have been defined to simplify the choices: morning, afternoon and grouped (prefers to group its classes on a minimum number of days). These profiles correspond to values of arbitrary parameters used for the tests.

5.1.Evaluation results

How should such a protocol be evaluated? We cannot check if the utility function is maximal, as we assume that the multi-agent system has several utility functions that are incomparable. We have checked that during the tests we always obtain a result and that this result is a Pareto optimum (as proved in sections 4.4.3 and 4.4.2). We have analyzed the performance of the protocol by observing several parameters: the number of messages exchanged, the size of these messages (the number of coalition sets they contain) and number of coalition sets that have been evaluated. The number of messages exchanged between the agents is independent of the search strategy. However, their size depends on the use of undeveloped coalitions. As for the number of evaluated sets it depends very much on the search method used. The basic method systematically evaluates all the possible sets whereas the heuristics proposed seeks to reduce the number of these sets in order to obtain the result more quickly. Four of these heuristics have been implemented. The heuristics which has given the best results and which has been used in the experiments described consists in seeking only the best group by doing intermediate tests as soon as possible in order to identify the value of the best group.

In the following, we will analyze these factors on a simple example using 4 agents (2 professors, 2 groups of students) and 2 classes. Each group does the two classes, i.e. we have four tasks in the system). Several experiments have been done with more agents (for more detail see [3]). In this example, we consider the schedule for two days, with eight possible times per day. We vary the profile of each agent (morning, afternoon, grouped) to obtain the average, maximum and minimum of the results. Students have two classes, each one must be placed in one of 16 time slots (or not to be placed if it is not given, so we have $16+1$ cases to consider) with only one possible professor for each class. Thus, there are 17^4 , i.e. 83,521 possible schedules. The order in which the negotiations proceed is student 1, student 2, professor 1 and professor 2. The last agent does not send any message as it just waits to receive a set which is appropriate for it and then sends it to the other agents as the solution for the negotiation.

The number of messages sent depends very much on the precision of the utility function of the agents. If agents have very precise preferences, they will distribute the sets among many small-size groups and will send many messages. The utilities of the agents considered here have one hundred levels. There is thus a maximum of 100 messages sent by the first agent (the second can thus send a maximum of 10,000 because it can divide each group

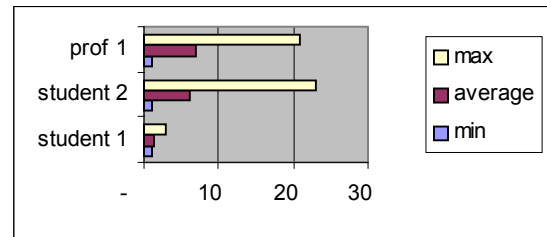


Figure 5. Number of messages exchanged during a negotiation with 4 agents and 4 tasks

received into 100 other groups). In this example, the number of messages sent is summarized in figure 5. The number of messages sent varies considerably according to the incompatibility of the preferences of the agents. For instance, a morning profile student will seek morning classes in priority. If the professor has an afternoon profile, it will consider these schedules unacceptable. Consequently, the student will have to send other propositions which it finds less appropriate. On the contrary, if all agents accept the first propositions, only one message per agent is necessary (minimal case).

The total size of the messages sent (figure 6) makes it possible to measure network obstruction. This size, measured with the number of sets, must be compared with the 83,521 possible sets. The agent which sends more sets and messages is student 2 for two reasons: (1) student 1 sends it the sets corresponding to its

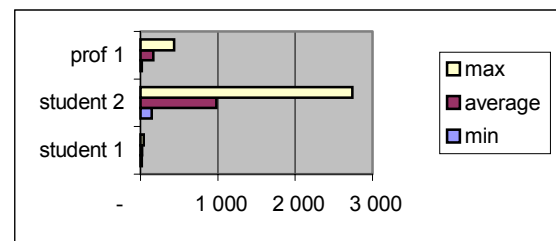


Figure 6. Total size of messages sent (measured by number of sets) during a negotiation with 4 agents and 4 tasks

two classes according to its preferences; (2) student 2 computes all the possible combinations of its own classes in each of these sets. It then sends these combinations in decreasing order of preference to professor 1 until there are no more acceptable sets to send and if no solution has been found. At this moment, student 1 sends it a second message and the negotiation continues. The purpose of dynamic restructuring of the coalitions is to give a result that is as satisfactory as our basic protocol but faster, which is possible because we use information drawn from the preceding negotiation by taking the previous solution as a new reference situation. The result will not necessarily (and probably will not) be the same, if the initial protocol has been applied, but the result is always a Pareto optimum.

We can study the effect of adding new classes to the previous situation in terms of the number of sets evaluated and transmitted. We gradually added 4 classes to students 1 and 2 (starting with the first). The first negotiation (4 classes) used our basic protocol, whereas the others four are restructurings of the previous situation. The number of sets sent and evaluated must be

compared with the total number of possible sets, which is always higher. It varies between 80,000 for 4 classes to $7 \cdot 10^9$ for 8 classes. The average size of the messages sent during these additions is indicated in figure 7.

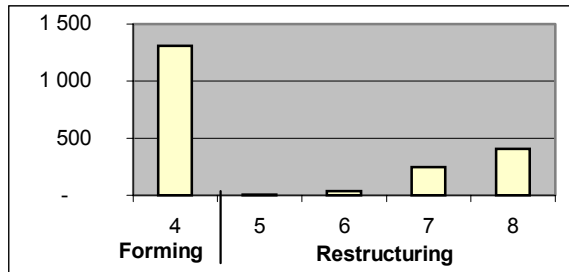


Figure 7. Size of messages sent (measured by number of sets) during a negotiation with 4 agents and classes varying from 4 to 8

The number of sets evaluated (and not sent) compared to the total number of possible sets is given in figure 8. During the formation of the coalitions corresponding to the four initial tasks, the total number of sets sent was on average 1,308 and the total number of evaluated sets corresponds to 5.5% of the total number of sets of possible coalitions (83,521). During the dynamic restructuring of coalitions, due to the addition of the 6th class, the total number of evaluated sets is 40 and the total number of evaluated sets corresponds to 0.00085% of the total number of possible coalitions. The number of sets sent and the number of sets evaluated is related not to the total number of tasks carried out but to two parameters: the number of tasks the agents fail to perform (because of incompatible preferences) and the number of new tasks. The effect is cumulative, which explains why the number of sets sent gradually increases. For instance, if the 6th class has not been assigned, this affects the number of sets sent after the addition of the 7th and the 8th classes because agent 1 tries again each time to assign the 6th class (which is not useless, as it may happen that in future negotiations a new class may modify the utility of the agents). The number of sets evaluated (figure 8) is here very low compared to the number of possible sets (on average 1% of the basic protocol and between $4 \cdot 10^{-4}\%$ and $6 \cdot 10^{-6}\%$ for the restructurings). However, the use of the basic protocol would have led to the computation and evaluation of all the possible sets before sending the groups of acceptable sets. The choice of a good heuristics to search for the best group is thus fundamental so that the search time is acceptable.

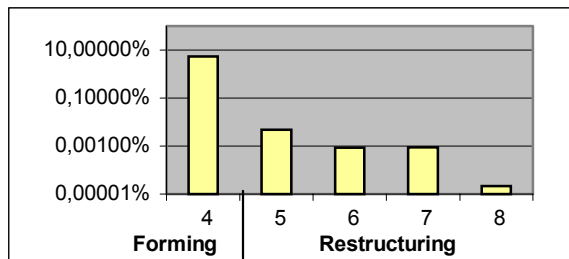


Figure 8: Evaluated sets of coalitions for 4 agents and classes varying from 4 to 8 (logarithmic scale)

6% for the restructurings). However, the use of the basic protocol would have led to the computation and evaluation of all the possible sets before sending the groups of acceptable sets. The choice of a good heuristics to search for the best group is thus fundamental so that the search time is acceptable.

6. RELATED WORK

Coalition formation protocols have largely been inspired by work in game theory (cf. [4]), which has provided the concepts used in MAS for the analysis of this problem (typology of the problems, solutions, equilibriums, utility functions). Through power indices, it is possible to calculate the real influence of an agent in a coalition. Game theory provides methods of calculation to define the best coalitions in various types of problems. In particular, its application to multi-agent systems has been studied by Sandholm [9]. The limits of its use are related both to the underlying assumptions (the agents are generally considered as perfectly rational) and to the aim (game theory focuses generally on the value of the optimal solution and not on the most efficient method to reach that solution, never mind the most efficient distributed method).

Current protocols in multi-agent systems are based on the following decomposition of the problem: generation of the coalitions, resolution of the optimization problem in each coalition and distribution of the created value between the agents. [11] proposes a simple and effective protocol. This protocol can be applied in very general cases (recovery of coalitions, scheduling) and makes it possible to find the best solution. However, it supposes that the agents have a common utility function. The protocol also implies that the value of the set of possible coalitions is calculated at least once. This gives a high complexity. [10] deals with this problem by proposing a method with a limited complexity while searching for a minimum result (with respect to the optimum result). [11] presents an analysis of the problems of having limits in calculation capacity and proposes a terminology adapted to this type of problem.

In more recent work, [12] proposed an algorithm based on the principle of electing a leader for coalition formation. This algorithm has been applied to electronic commerce processes. This approach is similar to the one proposed in [1]. Lerman *et al.* have proposed an alternative, physics-motivated mechanism for coalition formation that treats agents as randomly moving, locally interacting entities [6]. They consider that a new coalition may form when two agents meet randomly, and it may grow when a single agent randomly meets the coalition. The aim of this work was to define a mathematical model, formalized as a series of differential equations. These equations have steady state solutions that describe the equilibrium distribution of coalitions, but the authors have not given any details of the autonomous agent behaviors and how they concretely use this mathematical model. No algorithmic specifications have been proposed and the convergence of this model has not been addressed.

Zlotkin and Rosenschein have proposed a mechanism for coalition formation that uses cryptography techniques for sub-additive task-oriented domains. This mechanism is based on a Shapley value. A Shapley value for an agent is a weighted average of all the utilities of the agent which contributes to all possible coalitions. The weight of each coalition is the probability that this coalition will be formed in a random process that starts with the first agent, and in which this coalition grows by one agent at a time such that each agent that joins the coalition is credited with its contribution to the coalition. The Shapley value is the expected utility that each agent will have from such a random process [14][15]. However, this mechanism can only be applied to small-

sized multi-agent systems because of its combinatorial complexity due to the calculation of all possible coalitions.

[1] and [13] are interested in problems where the agents have their own utility function and where an aggregation is necessary only within a coalition ([1]) or in an alliance ([13]). [1] uses the Choquet integral to carry out multi-criteria aggregations among agents which can be either cooperative or competitive. Moreover, the protocol does not suppose that all the agents know each other. The protocol is limited however if the coalitions are disjointed. [13] studies the case of not disjointed coalitions which are formed gradually through alliances and progressive adaptation of the preferences of the agents (whose interest it is to adapt so as not to be excluded from the coalitions).

It is difficult to compare our protocol with current protocols since it does not have the same objectives. In current protocols, utility functions of the agents are systematically aggregated or adapted. On the contrary, the utilities here are neither aggregate nor transmitted. The results cannot thus be compared because they relate to different problems. However, if all the agents have same utility function at the beginning, the protocol suggested should obtain the same result as that of [11] or [1].

7. CONCLUSION AND PERSPECTIVES

For the problem considered (formation and restructuring of coalitions without aggregation of agent preferences), we have shown that the protocol makes it possible to obtain a solution which is a Pareto optimum. Moreover, the tests have shown that the average complexity remained low compared to the total number of possible cases. In spite of these encouraging results, many improvements are still possible and are currently being addressed.

Regarding the protocol, a logical extension would be to send sets with constraints on the coalitions instead of sending several independent sets of coalitions. For instance, in our application of drawing up schedules, instead of transmitting three sets of coalitions with the three alternatives time 1, time 2, time 3, one agent could send: "time ranging between 1 and 3". That would reduce the number of sets of coalitions to be computed and would enable the agent which receives them to make an intelligent search instead of having to evaluate all the sets without seeking links between them.

In short, the protocol proposed is adapted to problems requiring coordination through the formation of coalitions and where it is not desirable, or possible, to aggregate the preferences of the agents. The protocol provides optimal Pareto-type solutions. If changes occur in the multi-agent system, it allows agents to compute new solution, which is always optimal, dynamically and quickly, on the basis of the current solution.

REFERENCES

- [1] Aknine, S., Pinson, S., Shakun, M.F., 2000, Coalition Formation Methods for Multi-Agent Coordination Problems, *Group Decision and Negotiation*.
- [2] Binmore K., 1999, Jeux et théorie des jeux, De Boeck, Bruxelles.
- [3] Caillou, P., 2000, Pareto Optimality Method for Coalition Formation and Dynamic Restructuring of Agent Coalitions, Master Thesis, Université Paris-9.
- [4] Kahan J.P., Rapoport A., 1984, Theories of coalition formation, Hillsdale, LEA, 1984.
- [5] Ketchpel S., 1994, Forming coalitions in the face of uncertain rewards, in *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, 414-419, Seattle.
- [6] Lerman, K., Shehory, O., 2000, Coalition Formation for Large-Scale Electronic Markets, *ICMAS*.
- [7] Ossowski S., 1999, Co-ordination in Artificial Agent Societies, Springer, Berlin..
- [8] Sandholm T.W., 1996, Negotiation among Self-Interested Computationally Limited Agents, PhD thesis, University of Massachusetts, Amherst.
- [9] Sandholm T.W., Larson K., Andersson M., Shehory O., Tohmé F., 1999, Coalition Structure Generation with Worst Case Guarantees, in *Artificial Intelligence*, 111: 209-238.
- [10] Sandholm T.W., Lesser V.R., 1997, Coalitions among computationally bounded agents, in *Artificial Intelligence*, 94: 99-137
- [11] Shehory O, Kraus S., *Methods for task allocation via agent coalition formation*, Artificial Intelligence 101 (1998) 165-200
- [12] Tsvetovat, M., Sycara, K., Chen, Y., Ying, J., 2000, Customer Coalitions in the Electronic Marketplace, Agents 2000 Conference, Barcelona, Spain.
- [13] Vauvert G., El Fallah – Seghrouchni A., 2001, Coalition Formation among Strong Autonomous and Weak Rational Agents, MAAMAW'2001, Annecy.
- [14] Zlotkin, G., Rosenschein, J., 1994, Coalition, Cryptography and Stability: Mechanisms for Coalition Formation Task Oriented Domains, in *AAAI'94*, Seattle.
- [15] Zlotkin, G., Rosenschein, J., 1996, Mechanisms for Automated Negotiation in State Oriented Domains, in *Journal of Artificial Intelligence Research*, 5.