

Coalition Formation among Strong Autonomous and Weak Rational Agents

Guillaume Vauvert and Amal El Fallah – Seghrouchni

Laboratoire d'Informatique de Paris Nord – UPRES-A 7030

Institut Galilée – Université Paris 13

99, av. J-B Clément – 93430 Villetaneuse – France

{guillaume.vauvert, elfallah}@lipn.univ-paris13.fr

Abstract. Since no optimal structure exists, organizations have to be flexible to dynamically react towards environment changes. In an economic context, agents are strongly autonomous and weakly rational and have to deal with cooperation and competition, as in task allocation domain. This paper proposes an open, egalitarian and distributed protocol based on the exchange of preferences computed using qualitative and quantitative criteria: agents will agree on coalitions to form in order to fulfill tasks. We prove that our protocol converges to reach a consensus. Experimentation shows that the most rigid strategy is not optimal and surprisingly that higher competition leads to easier consensus.

1 Introduction

Multi-Agent Systems (MAS) are becoming more and more important for three main reasons pointed out in [12, 23]: a growing communication infrastructure over which separately designed agents can interact; applications for computer support for negotiation at the operative decision making level; and industrial trend towards virtual enterprises.

When different agents act in the same environment, they need means of coordination and cooperation. Cooperation becomes essential when: 1) agents cannot perform tasks by themselves; 2) other agents are more efficient in performing these tasks; and 3) working on the task collaboratively will increase benefits or reduce its costs [32].

To increase the efficiency of task achievement at the system level (more tasks can be satisfied) and at the agent level (increase agent's ability to satisfy their goals and maximize their own personal payoff [29]), agents may work jointly [39, 25, 31, 11]. A coalition is a group of agents whom have decided to cooperate in order to carry out a common task [32]. A coalition can work on a single task at each time [32, 28], but sometimes agents may be members of more than one coalition [32].

Each real-world organization has his own structure which is influenced by many factors (efficiency of the conception, size and age, technical system, environment, power [16]). All organizational theories agree that no optimal structure exists, thus organizations have to modify dynamically their structure to follow

environment changes (see 6.1). Coalition formation takes into consideration requirements and constraints arising from the dynamic nature of the environment [32], even if new informations are generally not integrated during all the process, but at some given times. After [36], most of theories criticize hierarchical and centralized structures [6, 5], and a new form of organization without centralized authority based on decentralized responsibilities has been propounded in [5]. Each team is technically and economically autonomous and no power relation exists between teams. This structure looks like a coalition (use of organizational theories is justified at section 6.1). In MAS, coalition formation allows to coordinate agents when information is distributed and changes dynamically [32]. Coalition formation addresses three problems [23], which are usually considered independent even if they are not: 1) coalition structure generation (partitioning or covering the agents); 2) solving the optimization problem (solving their joint problem and receiving eventually a benefit depending on used resources and spent time) and 3) dividing the value of the solution among agents (decided by agents themselves or imposed before beginning the process, addressed by game theory [21]).

The problem of distributed task allocation has been tackled in the Distributed Problem Solving (DPS) context [35]: an agent that attempts to satisfy a task may divide it into several sub-tasks using sub-contracts. In this case, a task is allocated to a single agent which is responsible of its performance (a distribution is necessary). Efficiency is evaluated through simulation.

Many works in game theory [21, 19, 13, 14] address the problem of coalition formation, but often concentrate on the distribution of the benefits, the stability and the fairness. Algorithms are usually centralized, with exponential complexity and not limited in communication and time and coalition are statically evaluated. Stability was widely studied in [10].

Set Partitioning Problem and Set Covering Problem have been studied in operational research, combinatorial algorithms and graph theory [8, 2, 3]. They have been shown to be NP-hard problems [7]. As argued in [28], these approaches provide no appropriate solution to the problem of coalition formation among agents, due to three main deficiencies: exact and optimal solutions have exponential complexity to be found; approximated solutions have polynomial complexity but sub-groups studied are artificially limited in size ([32] uses this technique to reduce complexity); and solutions are centralized. Furthermore, those works don't take into account agents' autonomy (this aspect will be developed in 6.1). Distributed Artificial Intelligence (DAI) uses game-theoretical concepts, but solutions are distributed. In this case, complexity, task allocation and communication are efficient. However, some underlying assumptions such as super-additive environment ([27, 39, 11]) do not hold in real-world MAS.

By the contrary, MAS deals with interactions among self-motivated, rational and autonomous agents.

This article is organized as follows: section 2 presents the main contributions related to coalition formation and classifies them according to the most significant criteria. It also put forward our claims. Section 3 formalizes the concepts we

define in order to solve the consensus problem. Section 4 proposes a distributed algorithm to be executed by agents during the consensus process. It goes on to prove the convergence of the proposed algorithm. Section 5 discusses experimentation and provides some of our most significant results. Finally, section 6 debates on issues of weak rationality and strong autonomy of agents.

2 Related work

Many coalition formation approaches exist, but address different problems in different domains. Sarit Krauss has proposed a classification of works in coalition formation [12]. We will extend this classification to emphasize our criteria.

2.1 Domain

The first set of criteria is about the domain. Distributed authority, communication and negotiation are always considered:

- Individual goals [27] *vs* Common goal [32, 28, 28] (social welfare maximizing).
- Self-interested [27, 29] *vs* Altruistic.
- Only pure software agents *vs* with interface software and people agents.
- Known rationality: group rationality [32], personal rationality [27, 29], coalitional rationality [28].
- Bounded rationality [24, 25].
- Positive externalities (cooperation) [27] *vs* Negative (competition).
- Number of agents: a dozen [32], a hundred [38], thousands.
- Size of solution space (number of possible coalitions): too many to be enumerated and evaluated (costly and/or limited time) [29, 23] *vs* Small space.
- Defined protocols agreed (regulations should be agreed in advance and are incorporated into all of the agents [29], since each agent chooses its own strategy [29]) *vs* non pre-defined protocols.
- Static *vs* evolutionary evaluation of incomes.
- Common *vs* individual evaluation of incomes.
- Costly [25] *vs* costless computation.
- Independent tasks [28].
- Satisfy the more tasks as possible [28] *vs* satisfy all tasks.
- Enough competences and agents to solve problem.
- Dynamicity: complete (agents appear/disappear and task may arrive constantly) *vs* partial (not modified during coalition formation process) [32, 31].
- Transferable resources between agents (more beneficial coalition [28, 29]) *vs* no transferability.
- Monetary system for side-payment [29].
- Set partitioning [29] *vs* set covering [30].
- CFG (value of each coalition is independent of nonmembers' actions) [29, 23, 11, 24, 39] *vs* non-CFG.
- Super-additive [11, 27, 39], sub-additive [24, 39], no assumption on additivity (most of cases).

Self-interested agents can deal with selfish goals (e-commerce) and altruistic agents can deal with common goals (problem solving in DPS), but there are other possibilities. Agents are altruistic if they are designed to collaborate, since a common goal is the aim of the system. Self-interest and altruism affect the design of agents, since individual and common goals concern the type of problem. The type of goal is given by the problem, since the type of agent is defined by environment (DPS, e-commerce), or resolution choice (DAI, MAS, ...). For example, a task allocation in DPS can be solved using altruistic agents since goals are individual. An another example: to solve a problem in MAS, agents have to try to satisfy a common goal even if they are self-interested. Generally, if the system's goal is to reach a common goal, benefits are mesearred from the system viewpoint.

Strong rationalities have been used to enable efficient protocols:

- Personal rationality [9, 14, 21]: an agent will join a coalition only if the payoff he will receive is greater than what he can obtain by staying outside (eventually getting a payoff that compensates him for the loss of resources or non fulfillment of some of its tasks).
- Coalitionally rational [27]: each coalition will add new members only when its new value is greater than the value of the original coalition.
- Group rationality [9, 21]: agents are group rational if forming a coalition always increases the global benefit.

In super-additive environment, grand coalition is optimal; thus the only problem that still remains is how the payoff should be distributed among its members.

In [24], bounded rational value of a coalition is determined by three parameters: 1) as usual, the domain problem (task and resources); 2) the execution architecture (limited and costly computation) and especially 3) the possibility for agents to design their protocols. Effects of computational limitations on coalition structure formation and stability have been studied in [25] and [24]. The third parameter is ambitious but we think that it is a necessary condition to design autonomous systems (6.2).

2.2 Quality of solutions

Solutions provided by these protocols are different since each protocol have its own properties. The following properties are important to choose the adapted protocol:

- Quality of solution (optimality) if a measure is available.
- Complexity in time and space to reach final state.
- Anytime, design-to-time [24].
- Certainty to reach a final state.
- Stability (studied in [21]).
- Limited number of agents per coalition [32, 30].

2.3 Our claim

In an economic context, agents have individual goals (to increase their incomes) and they are self-interested; agents might be pure software agents or interface agents for human, and then no strategy is assumed and rationality is bounded. The defined protocol is known and agreed by agents, but they are completely autonomous: protocol takes into account possibilities for agents to try to cheat. The problem of task allocation binds agents to cooperate in order to fulfill tasks (each agent is able to fulfill sub-tasks). We assume that all tasks can and must be fulfilled. A task might be dependent of an another (precedence order, income decrease, same/different agent for some sub-tasks) and coalitional value may depend on non-member actions: this may be taken into account by a modification of solution space and of sub-tasks incomes (but no experimentation have been made upon). Resources may be not transferable, but if they are, agents may exchange resources outside the protocol without modifying it.

A monetary system is used for experimentations to simplify computation, but since the protocol is only based on preferences exchange, it is not necessary (agents need only criteria to compute their preferences).

The solution space might not be too large; but if it is, each agent might use heuristics to quickly evaluate the best solutions.

The number of agents may be large (around 25), and experimentations show that the number of turns decreases when the number of agents increases (time however increases because each turn spend more time).

Evaluations of incomes are individual and may evolve during the process. Computation and communication time might be taken into account, by decreasing sub-task income as time elapses, but strategies and experimentations don't take that into consideration.

Experimentations assume that agent may fulfill different sub-tasks in different coalitions, but the protocol run with a partition of agents: solution space has simply to be reduced.

Optimality of the chosen task distribution has no sense here: it depends on agents' strategies. However, chosen solution is legitimate, because no agent is favored.

In this defined context, we propose a protocol that takes into account strong autonomy and weak rationality (6.1) to reach a consensus about a sub-task distribution.

3 Coalition Formation

Each agent likes some solutions and dislikes others. To reach a consensus, agents have to exchange information to possibly evolve their preferences. Argumentation should be used, but it needs a complex process, it binds agents to have a common communication language and to know the rationality of others. Heterogeneous agents should prefer to exchange basic information that don't need such a formal process. Thus, at each turn, agents send their preferences to others and consider other's preferences to compute their next preferences. Because agents whom don't make concessions are more likelihood to be ejected from the final

solution (see 5), agents may be flexible. If they aren't, they may form alliances; if no alliance is formed, agents choose two agents whom are obliged to ally. Finally, alliance formation leads to facilitate a consensus to be reached. This algorithm is more broadly beared out and described in [37].

3.1 Case study

Let us now present the concepts of the coalition formation problem and highlight their meaning within an application: airlines choose to cooperate in order to provide their passengers with a unified reservation system. The problem is that for each travel, several airlines are in competition on some stages.

3.2 Formalization

Definition 1 (Coalition Formation Problem (CFP)). A CFP is defined as a tuple $\langle \mathcal{A}, \mathcal{T}, \mathcal{S}, \mathcal{C}, \mathcal{P} \rangle$, where:

\mathcal{A} : the set of agents candidate to the execution of sub-tasks;

\mathcal{T} : the set of tasks to be accomplished;

\mathcal{S} : the set of sub-tasks to be carried out;

\mathcal{C} : the set of competences necessary to perform the sub-tasks;

\mathcal{P} : the set of incomes.

Where:

An **agent** $a \in \mathcal{A}$ is defined by: $a = \langle C, \text{strategy} \rangle$, $C \subset \mathcal{C}$, and strategies contain preferences computation and some criteria used to form alliances (see 3.3).

A **task** $t \in \mathcal{T}$ is defined by the set of sub-tasks it contains: $t = \langle S \rangle$, $S \subset \mathcal{S}$.

A **sub-task** $s \in \mathcal{S}$ is defined by $s = \langle C, p \rangle$, $C \subset \mathcal{C}$, $p \in \mathcal{P}$, where c is the set of competences which an agent must have to be able to carry out the sub-task, and p the associated profit. This profit will be used by agents to compute their preferences.

A **competence** $c \in \mathcal{C}$ is a single item which represents what is required for a sub-task to be carried out by an agent. A sub-task requires one or more competence.

A **profit** $p \in \mathcal{P}$ is used as an income, but only to simplify the internal computations of agents: $\mathcal{P} = [0, \text{MaxProfit}]$, $\text{MaxProfit} \in \mathbb{R}$. However, several types of decision making should be used to compute preferences.

Example 1. $\mathcal{A} = \{EUropeanAirlines, AMericanAirlines, WOrldAirlines, \dots\}$

A task is a flight between two cities which puts into others: $\mathcal{T} = \{\text{New York-MAdrid (via PArise and LYon)}, \text{Los Angeles-MOscow (via New York and PArise)} \text{ and } \text{BERlin-JOhannesburg (via PArise)}\}$.

$\mathcal{S} = \{\text{New_York} \rightarrow \text{PArise}, \text{LYon} \rightarrow \text{MAdrid}, \text{PArise} \rightarrow \text{MOscow}, \dots\}$. We can now define the task $NY - MA$ by $NY - MA = \langle \{\text{NY} \rightarrow \text{PA}, \text{PA} \rightarrow \text{LY}, \text{LY} \rightarrow \text{MA}\}, \dots \rangle$.

Each flight needs competences: authorization to do a national stage, passengers capacity, range of action.

$\mathcal{C} = \{\text{autX}, \text{WC}, \text{MC}, \text{LC}, \text{VSR}, \text{SR}, \text{MR}, \text{LR}\}$

$EUA = \langle \{\text{autFR}, \text{autEU}, \text{autRU}, \text{WC}, \text{MC}, \text{SR}, \text{MR}\} \rangle$

We choose $\mathcal{P} = [0, 10000]$ and for example, we have $NY - MA = \langle \{\text{NY} \rightarrow \text{PA}, \text{PA} \rightarrow \text{LY}, \text{LY} \rightarrow \text{MA}\}, 8000 \rangle$.

To solve this problem, agents exchange their preferences about possible solutions. If no consensus is reached, they may form an alliance. So, agents need to represent solutions, preferences, alliances and coalitions.

To reach a consensus, agents have to modify their opinion. Among all the possibilities (pressure, corruption...), we choose the preferences exchange. First, it appears more rational to us, since it leads to a more legitimate solution. Also it may be used by heterogeneous agents, while high level languages need complex symbolic process.

Definition 2 (Solution). *A solution is an assignment of each sub-task to an agent which is able to perform it. A solution $\sigma \in \Sigma$ is an application $\mathcal{S} \rightarrow \mathcal{A}$ such that $\forall s \in \mathcal{S}, a = \sigma(s) \Rightarrow s.C \subset a.C$.*

Definition 3 (Preference). *A preference is represented by distances (not in mathematical meaning) $\delta \in \Delta$ between solutions, where $\delta : \Sigma \times \Sigma \rightarrow [-1, 1]$ is an antisymmetrical application. So, $\delta(\sigma_1, \sigma_2) = d$ is interpreted by “ σ_2 is preferred to σ_1 with a distance d if $d \geq 0$ and σ_1 is preferred to σ_2 with a distance $-d$ if $d < 0$ ”. A null distance means that agent has no particular preference.*

Example 2. $\sigma_{15} = [\text{NY} \rightarrow \text{PA}_2 \hookrightarrow \text{WOA}, \text{LY} \rightarrow \text{MA} \hookrightarrow \text{BUA}, \text{PA} \rightarrow \text{MO} \hookrightarrow \text{EUA}, \dots]$. Let $S_1 = \{\sigma_0, \sigma_2, \sigma_4\}$ the set of solutions which provide outcomes and $S_2 = \{\sigma_1, \sigma_3, \sigma_5\}$ the set of solutions which provide none. $\delta(\sigma, \sigma') = 0$ if σ and σ' are in the same set, and $\delta(\sigma, \sigma') = 1$ otherwise.

Definition 4 (Sight). *A sight $v_t \in V$ is an application $\mathcal{A} \rightarrow \Delta$ and $t \in \mathbb{N}$ the turn number.*

Definition 5 (History). *A history $h \in H$ is a sequence of sights $(v_t)_t$. A history $h = (v_t)_{1 \leq t \leq T}$ represents all preferences exchanged from the first turn to the last one.*

An alliance is a set of agents and behave like a single one. A member have a representative role: he communicates with other representatives (sending alliance preferences).

Definition 6 (Alliance). *An alliance $\lambda \in \Lambda$ is defined by $\lambda = \langle A, a_{rep} \rangle$, where $A \subset \mathcal{A}$ and $a_{rep} \in \mathcal{A}$ an alliance member with a representative role, with the constraint that an agent can belong to only one alliance.*

Definition 7 (Coalition). *A coalition $\Omega(\sigma, t) \subset \mathcal{A}$ associated to the task $t \in \mathcal{T}$ in the solution $\sigma \in \Sigma$ is defined by: $\Omega(\sigma, t) = \{a \in \mathcal{A} / \exists s \in \mathcal{S}, s \in t.S, \sigma(s) \ni a\} = \bigcup_{s \in t.S} \sigma(s)$. A coalition contains all the agents which take part in a task.*

The agent with the representative role takes informations from all alliance members and computes the alliance preference. As suggested in [1], it is difficult to find a function which behave accordingly to our intuition of preference aggregation.

Definition 8 (Alliance Preferences Computation). *An alliance preferences computation APC is an application $\Lambda \rightarrow \Delta$. This application is known only by alliance members; other agents only known result of computation.*

Example 3. Let $\lambda \in \Lambda$ an alliance, $\lambda = \{A, a_{rep}\}$, $A \subset \mathcal{A}$. $APC(\lambda) = \delta$, where δ is defined by: $\forall(\sigma_1, \sigma_2) \in \Sigma^2$, $\delta(\sigma_1, \sigma_2) = \sum_{a \in A} a.\delta(\sigma_1, \sigma_2)$. This example use only members preferences to compute alliance preference.

Definition 9 (Initial Preferences Computation). *An IPC is an element $\delta \in \Delta$, that should be computed using incomes or another criterion.*

Definition 10 (Dependent Preferences Computation). *A DPC is a function $H \rightarrow \Delta$, $h \mapsto \delta$.*

Example 4. Let $\delta = IPC$, $\forall(\sigma_1, \sigma_2) \in \Sigma^2$, $\delta(\sigma_1, \sigma_2) = profit(\sigma_2) - profit(\sigma_1)$. δ is an antisymmetrical application. Let $\delta = DPC(h)$, $h = (v_t)_{t \in \mathbb{N}}$. $\forall(\sigma_1, \sigma_2) \in \Sigma^2$, $\delta(\sigma_1, \sigma_2) = [\sum_{a \in \mathcal{A}} (v_T(a))(\sigma_1, \sigma_2)] / |\mathcal{A}|$. δ is an antisymmetrical application.

3.3 Agent's strategy

The agent's strategy depends on his preferences computation : Initial Preferences Computation (computation of the first preferences without knowing those of the others) and Dependent Preferences Computation (computation of preferences of next turns).

Definition 11 (Strategy). *Representative agent has particular procedures that define his strategy:*

- **Releasing Switch-over Proposal Criterion:** *criterion used to decide when to propose to release to switch-over mode. A RSPC is an application $H \rightarrow \{False, True\}$.*
- **Releasing Switch-over Acceptance Criterion:** *criterion which decides to accept or not to switch to release mode. A RSAC is an application $H \rightarrow \{False, True\}$.*
- **Alliance Formation Proposal Criterion:** *gives a list of agents to which to propose to form an alliance. An AFPC is an application $H, A \rightarrow \{False, True\}$.*
- **Alliance Formation Acceptance Criterion:** *allows to answer to alliance formation propositions. An AFAC is an application $H, A \rightarrow \{False, True\}$.*

Example 5. Let $h = (v_t)_{1 \leq t \leq T}$. $RSPC(h) = False$ if $T \leq 2$ and $RSPC(h) = (v_T = v_{T-1}) \vee (v_{T-1} = v_{T-2}) \vee (v_T = v_{T-2})$ otherwise. To reduce computation complexity, only loops of length 3 or less are detected and to simplify computations, $RSAC = RSPC$.

Let $d : \Delta \times \Delta \rightarrow \mathbb{R}$ a distance between agents preferences, for example: $\forall(\delta_1, \delta_2) \in \Delta^2$, $d(\delta_1, \delta_2) = \sum_{(\sigma_1, \sigma_2) \in \Sigma^2} |\delta_1(\sigma_1, \sigma_2) - \delta_2(\sigma_1, \sigma_2)|$. For an agent a , $AFPC(h)$ is the set of agents which preferences are enough near to him using a threshold. We can use the same application to compute $AFAC$ but using a greater threshold.

4 The algorithm of consensus protocol

Each agent may play several roles within the system:

- Candidate role: it exists only at the beginning of the process, when agent receives tasks to fulfill and decides to take part in or not.
- Member role: he receives and sends his preferences within his alliance or the system, proposes/accepts the switch-over releasing mode, proposes/accepts the alliance formation.
- Representative role: he receives the preferences from other members of the system and broadcasts it to internal ones and reciprocally.
- Organizer role: he sends datas and manages inscriptions and turns.
- Supervisor role: he is responsible for checking respect of the protocol, in particular that agents send the same information to all others.

Agents may try to refuse the protocol in order to earn more money. The proposed protocol prevents agents to use information that others don't have, using a parallel diffusion (crypting messages: see [37]), but agents may cheat by sending different preferences to each agent. To avoid that, supervisors are allowed to ask agents what preferences have been sent and received.

At the beginning, each agent is a candidate. He asks the organizer for informations, and if he chooses to participate in the sub-tasks distribution, he becomes a member of an alliance of which he is also the representative (because he is alone in his alliance). During the process, an agent may lose or win a representative role (when a new alliance is created). Supervisors verify that the agents send same preferences to all others; if it is not the case, the culprit has to pay penalty to the supervisor. The influence of penalty amount on cheat has not been experimented (6.2).

4.1 Representative algorithms

The representative's algorithm plays a leading role. Each representative has a list of interlocutors $InterList \subset \mathcal{A}$ initialized with the list of the candidates. The algorithms 1 and 2 are carried out by each representative in a distributed way.

Algorithm 1 Representative: Switch-over mode

```

broadcast("Propose to form an alliance",  $AFPC(h)$ )
for  $a$  such that receive("Propose to form an alliance",  $a$ ) do
  if  $a \in AFAC(h)$  then send("Accept to form an alliance",  $a$ )
end for
if no formed alliance then
  system selects entities with nearest preferences
  system force them to form an alliance
end if
Bring up to date  $InterList$ 
Return.
```

Algorithm 2 Representative: Main

```

receive("IndPref",AllianceMembers)
 $h \leftarrow \text{ParallelDiff}(\text{IndPref}, \text{InterList})$ 
send("IndPref",AllianceMembers)
while the consensus isn't reached do
  if  $RSPC(h)$  then send("proposition to switch-over mode",)
  if receive("proposition to switch-over mode",) then
    if  $RSAC$  then send("proposition to switch-over mode",)
  end if
  if  $\forall a \in \text{InterList}, \text{receive}$ ("proposition to switch-over mode",a) then
    call switch-over mode {switch-over mode accepted}
  end if
  receive("DepPref",AllianceMembers)
   $h \leftarrow \text{ParallelDiff}(\text{DepPref}, \text{InterList})$ 
  send("DepPref",AllianceMembers)
end while

```

The members of an alliance have to compute their preferences, send them to their representative and receive the other preferences.

4.2 Validation and complexity of the algorithm

Termination. Without any assumption on the criteria of switch-over mode releasing, we are not able to guarantee that the process terminates. However, if we assume that criterion checks the existence of a loop, we can prove that it ends. In fact, if the same situation occurs twice, then an alliance is necessarily formed. If we assume that the number of situations is finite, then necessarily this case will happen. In the worst case, there will be only formations of forced alliances, what will lead to a great alliance. In fact, the number of situations is not finite because preferences use real numbers. To escape this problem, we consider that two sights are equal if all their preferences are rather close w.r.t. the given distances as introduced in Example 5.

Definition 12 (Pseudo-equality between preferences). *Let ε a small real and let δ and δ' two preferences.*

We shall say that δ and δ' are pseudo-equal ($\delta \simeq \delta'$) if $\forall \sigma \in \Sigma, |\delta(\sigma) - \delta'(\sigma)| < \varepsilon$.

Definition 13 (Pseudo-equality between sights). *Let ε a small real and $v_t =$ and $v_{t'}$ two sights.*

We shall say that v_t and $v_{t'}$ are pseudo-equal ($v_t \simeq v_{t'}$) if $\forall a \in \mathcal{A}, |v_t(a) - v_{t'}(a)| < \varepsilon$.

Definition 14 (A cycle-like in a history). *We say that a history $h = (v_t)_{1 \leq t \leq T}$ contains a cycle-like if $\exists (\tau_1, \tau_2) \in \{1, \dots, T\}^2, \tau_1 \neq \tau_2$ such that $v_{\tau_1} \simeq v_{\tau_2}$.*

Definition 15 (A CFP detects cycle-like). *A CFP $\langle \mathcal{A}, \mathcal{T}, \mathcal{S}, \mathcal{C}, \mathcal{G} \rangle$ detects cycle-like if $[h \text{ contains a cycle-like} \Rightarrow (\exists a_0 \in A \text{ such that } a_0.RSPC(h) = \text{True})]$*

$\wedge (\forall a \in A, a.RSAC(h) = True)]$. In other words, a CFP detects cycles-like if at least one agent detects it and all then accept to change mode.

Lemma 1. *If a CFP detects cycles-like and there is a cycle-like, then an alliance will form.*

Proof. If a CFP detects cycles-like and there is a cycle-like, then at least one agent will propose to change mode and all other will accept. Agents may then form alliances. If they don't, two agent will be compelled to form an alliance. \square

Theorem 1. *If a CFP detects cycle-like, then the program terminates.*

Proof. As number n of agents and number k of solutions aren't infinite, the number of sights not pseudo-equal is finite. A preference between two solutions may have $2/\varepsilon$ different values; then a preference may have $[k(k-1)/2]^{2/\varepsilon}$ different values. The number of sights is thus $n[k(k-1)/2]^{2/\varepsilon}$. In a finite number of turns, the history will necessarily contain two identical sights; thus there will have been a cycle-like and one alliance will be formed (see Lemma 1 above). This alliance formation can only occur $n-1$ times, because agents cannot leave an alliance, what leads at worst to a grand alliance (which contains all agents). Finally, after $n(n-1)[k(k-1)/2]^{2/\varepsilon}$ turns at worst, there is consensus. \square

Complexity. Complexity depends in particular on the number of possible solutions which is directly related to the problem datas. Let us suppose that our system contains n agents. There's no sense to calculate the average number of agents able to solve a sub-task, because that ignores either very widespread competences, and especially competences which are linked. Thus let us suppose that each agent can process a portion of $1/m$ of the tasks, thus a task has on average n/m agents that are able to solve it, what then gives $k = (n/m)^{|\mathcal{S}|}$ solutions, where $|\mathcal{S}|$ is the number of sub-tasks. When the number of agents grows, the number of solutions becomes too large to be processed; it is then necessary to limit space to be considered, but there is then loss of information. In the most general case, our algorithm does not make it possible to change class of complexity, but a calculation on average shows that without alliance formation the more agents have competences which overlap, the more there is competition and the more the consensus is difficult to reach. Experimentations show that with alliance formation, the number of turns is bounded (begin to increase quickly and then decrease slowly).

5 Experimentation

Many parameters influence the process, but three of them have more influence: agents strategies, competences repartition (more or less competition) and the number of agents.

To measure the influence of the first parameter, the number of agents is fixed (7). The preference of the turn t is computed by adding the independent preference (from income of the solution) with weight $w(t)$ and the average preferences of others agents with the weight $w(t)$, where $w(t) = Sup(0, -\alpha t + 1)$. This weight

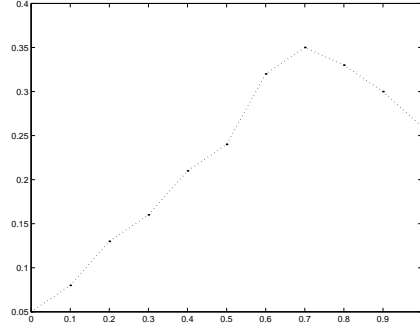


Fig. 1. Income function of strategy for one global strategy

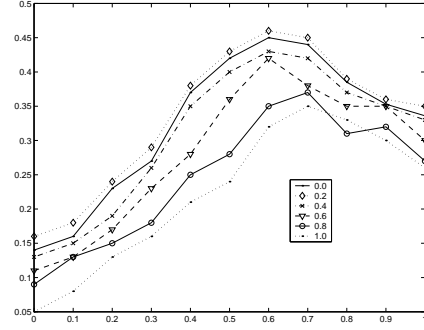


Fig. 2. Income function of strategy for 6 strategies

simulates a more or less flexible strategy. The goal of this experimentation is to find the best average strategy according to other strategies. In Fig. 1, only the strategy of a particular agent varies : α varies from 0.0 (flexible strategy) to 1.0 (rigid strategy) by step 0.1, since all other agents use the same value $\alpha = 1.0$. Results are the average of a large amount of experimentations (350). As expected, agent's income begin to increase when agent's strategy become more and more rigid. But, around 0.7, agent's income decreases: to be too rigid should lead an agent to be excluded from chosen solution, he will so earn less income. Fig. 2 exhibits that this result is true for all other agents' strategies, since optimal value is around 0.6. That should lead agent to choose flexible strategies.

Fig. 3 shows that when more agents are rigid, consensus is hardly reached. If

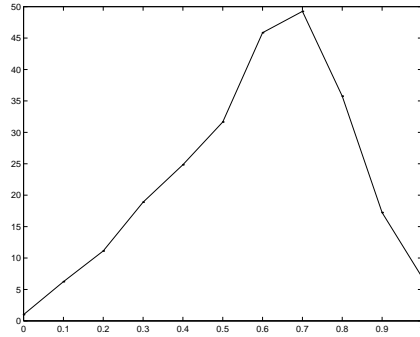


Fig. 3. Number of turn in function of strategy

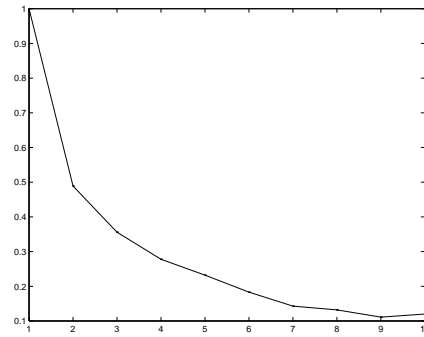


Fig. 4. Income in function of number of agents per sub-task

agents are too rigid, jamming detection leads to form an alliance and consequently to reach a consensus more quickly, even if the last is not desired.

More the agents have competences, more they have to compete with others. We studied the influence of the number of agents per sub-task (competition level) on the incomes (Fig.4) and on the number of turns (Fig.5). As expected, when competition increases, incomes decrease and consensus become more difficult to reach.

As the number of agents increases (Fig.6), there are more and more agents able

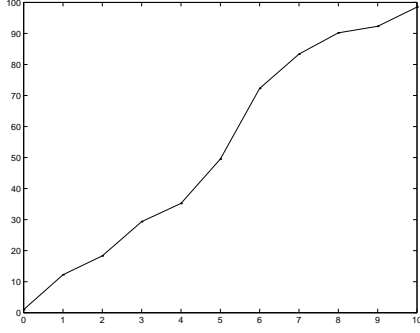


Fig.5. Number of turns in function of number of agents per sub-task

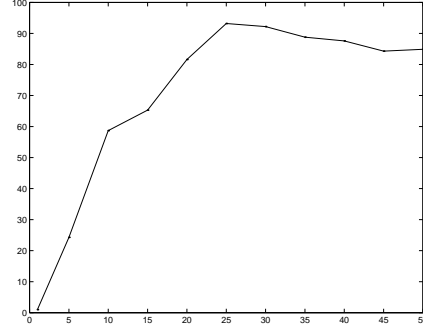


Fig.6. Number of turns in function of number of agents

to fulfill sub-tasks and competition increases. But if the number of agents is greater than 25 (this value depends on other parameters), then reaching a consensus is easier, because the formed coalition contains enough agents to fulfill all the tasks: usually, one coalition fulfill all tasks.

Finally, this protocol provides some surprises: to be extremely rigid is not optimal and high competition leads to a faster consensus.

6 Discussion

Agents we consider are autonomous and rational. This has to be taken into account when designing protocols, but definitions of autonomy and rationality depend on the different viewpoints.

6.1 Autonomous and rational agents

In most works, an agent is autonomous if he has his own goals; these goals are known and are used to design protocols that satisfy the agents as best as possible. In [29], protocols (called regulations) are incorporated into all of the agents, but each agent chooses its strategy for the interaction individually and joins a coalition only if it increases its personal payoff: he seems to be autonomous because he can choose his strategy, but this autonomy is bounded because he must only join beneficial coalitions.

In real-world systems, an agent can be a human interface, which complexity

prevents him to be modelised by a goal to be reached. Even if an agent is not a human interface, a definition of strong autonomy leads to consider that agents goals and behavior cannot be known. We say that an agent is strongly autonomous if he can decide and acts as he wants, even if we reasonably assume that in general he acts according to his approximatively known goals. For example, we consider that an agent in an economic context acts generally to maximize his benefits, but that for some reasons (strategical, technical or unknown), he can prefer to loss money (at least in limited time).

We have also a particular point of view about the notion of rationality. Usually, an agent is said to be rational if he behaves to maximize a criterion (utilities, incomes, *etc*). Since [34] and [22], bounded rationality replaces economical rationality, because an agent doesn't spend all his time to optimize his criterion, because of his limited resources: he will stop when he will be satisfied. Bounded rationality has been used to design agents and protocols that take into account bounded resources [24, 25, 4]; but in fact motivations might be so complex that it is impossible to predict agents' behaviors. Thus, we will consider a weak rationality: an agent is rational if he is inclined to act according to his assumed motivations. His behavior is thus estimated (with more or less uncertainty) but never predicted.

The legitimacy of the solution is also important: a less legitimate solution is to randomly choose the coalition structure among all possible coalition structures, but this solution is not good. To legitimate the solution, many works choose an external criteria of satisfaction based on an assumed rationality [32, 27–29]. The solution may then be accepted, because it satisfies globally as best as possible the agents. But first agents should prefer to negotiate to hope to won more, taking the risk to lose more; then this solution needs to know agents criteria.

Differences on autonomy exist in organizational theories [26]: in one hand, management theories, in the other, sociological theories.

Organizational theory works address the problem of the diversity of real-world organization structures. Usually [26], these works are classified in three categories: management theories, sociological theories and psychological theories. The goal of management theories is to analyze, to distribute and to coordinate activities ensuring that activities are processed with the minimum of money and/or time. The objective is to advise the leader to design an efficient and durable structure. Sociological theories study why structures are different, why they are more or less stable and more or less efficient: relations between structure and efficiency are not simple (two organizations with same or opposite structures might be equally efficient or really not). Economical theories, psychological theories and labour domain theories are not sufficient to explain complexity of phenomena. Sociological theories try to find parameters that determine objectively the structures. Each organization has to dynamically modify its structure in order to adapt itself to modifications (environment, technology, strategy, size, *etc*), but this adaptation is difficult because of defaults of bureaucracy and organization and members goals [20]. Psychological theories study which organization characteristics influence the labour in firms, but they are too human-dependent

to be used in virtual organizations.

Management theories try to predict the behavior of organizations and to advise managers since sociological theories are explanatory. The first use simple and unrealistic model of human behavior, but results are important. The second are precise and realistic but they are not completely usefull because of the complexity of the models. Management theories deal with weak autonomy and strong rationality, while sociological theories deal with strong autonomy and weak rationality.

The question to be answered is: “When should strongly autonomous and weakly rational agent be used, and when they should not ?” Such agents have to be used when they represent interfaces for people, but also when the MAS to be designed deals with heterogeneous agents designed by various people, at different times, in large scale systems. In other cases, when a small number of agents is designed by a single person, when the system has not to evolve, agents may be weakly autonomous and strongly rational (we don’t talk about human system simulation). In fact, this conception of an agent is rather close to Distributed Problem Solving than Multi Agent Systems.

6.2 Protocols

Considering weak autonomous and strong rational agents allow the design of protocols as a decentralized algorithm which computes a solution that maximizes personal but known criteria. Impossibility of maximizing all criteria leads to consider a global and a coalitional rationality (see 2.1) or an average maximization. The Shapley Value is an example of average maximization: it is a specific payoff division among agents that motivates individual agents to stay in the coalition structure space (Shapley Value is guaranteed in [23]). Under our assumptions, a protocol has to allow agents to reach a consensus, by influencing other agents according to his preferences (weak rationality), taking into account any agent’s act (strong autonomy). For example, agents can try to cheat to earn more money, even though honesty they are assumed to calculate Shapley Value [27]. This paper assumes that agents are strongly autonomous, but don’t say how to design them. We just assume that they are, for example considering interface agents. Our work addresses the problem of the design of protocols that take into account this autonomy, not the problem of agent design. In fact, pre-compiled highly-structured “social laws” are used to coordinate agent activity [33, 17], but agents are designed to follow the social laws since they were designed to do so and not because they benefit individually from following these laws: the designers should agree in advance which regulations the agents will use.

A good example of strong autonomy is given in [23]. Agents are assumed to be insincere, and the goal is then to motivate self-interested agents to follow the desired search method in order to reach a socially desirable outcome, as a consensus. For example, enforcement mechanisms will motivate the agents to search exactly what they are assigned. To motivate agents to follow the protocol, [23] uses penalties: agents whom find a cheater are rewarded by the penalty paid by the cheaters. Agents have also to be motivated to search cheater. But if the penalty is high enough, the supervisor is motivated to search them and then

agents are not motivated to cheat; supervisors will then neither be motivated to search. There is no Nash equilibrium (there is Nash equilibrium when each agents strategy is a best answer to the strategies of the others [15, 18]).

We assume that agents are strongly autonomous. They choose their behavior knowing the protocol, but this protocol is not chosen by agents. Autonomy at the system level should allow agents to design themselves the protocol they will use to solve their problem.

References

1. K. Arrow. *The Origins of the Impossibility Theorem*, chapter 1, pages 1–4. Elsevier Science Publishers B. V., Amsterdam, 1991.
2. E. Bala and M. Padberg. On the set covering problem : an algorithm for set partitioning. In *Operation Research*, volume 20, pages 1152–1161. 1972.
3. N. Christofides and S. Korman. A computational survey of methods for the set covering problem. In *Mathematics of Operations Research*, volume 21 (5), pages 591–599. 1975.
4. A. DeVany. Information, bounded rationality, and the complexity of economic organization. *Taiwan Journal Political Economy*, 1, 1996.
5. H. Dubreuil. *La Fin des Monstres, Idée d’une Organisation Contraire à la Centralization et à l’Étatisme*. Grasset, 1938.
6. M. P. Follet. *Dynamic Administration*. Pitman, 1924.
7. M. R. Garey and D. S. Johnson. *Computers and Intractability : a Guide to the Theory of NP-Completeness*. W. H. Freedman and Company, New York, 1979.
8. R. S. Garfinkel and G. L. Nemhouser. The set-partitioning problem : Set covering with equality constraints. In *Operation Research*, volume 17, pages 848–856.
9. J. C. Harsanyi. *Rational Behavior and Bargaining Equilibrium in Games and Social Situations*. Cambridge University Press, 1977.
10. J. P. Kahan and A. Rapoport. *Theories of Coalition Formation*. Lawrence Erlbaum Associates Publishers, Hillsdale, NJ, 1984.
11. S. P. Kepchel. Forming coalitions in the face of uncertain rewards. In *Proc. of AAAI94*, pages 414–419, Seattle, Washington, 1994.
12. S. Kraus. Negotiation and cooperation in multi-agent environments. *Artificial Intelligence*, 94(1-2):79–98, 1997.
13. S. Kraus and J. Wilkenfeld. Negotiations over time in a multi agent environment. In *Proc. of IJCAI-91*, pages 56–61, Australia, 1991.
14. R. D. Luce and H. Raiffa. *Games and Decision*. John Wiley and Sons, Inc, 1957.
15. A. Mas-Colell, M. Whinston, and J. R. Green. *Microeconomic Theory*. Oxford, 1995.
16. H. Mintzberg. *The structuring of organizations : a synthesis of the research*. 1981.
17. Y. Moses and M. Tennenholtz. Off-line reasoning for on-line efficiency. In *IJCAI 93*, pages 490–495, 1993.
18. J. Nash. Equilibrium points in n-person games. In *Proc. National Academy of Sciences*, volume 36, pages 48–49, 1950.
19. J. Von Neumann and O. Morgenstern. *Theory of Games and Economic Behavior*. Princeton University Press, Princeton, N. J., 1947.
20. C. B. Perrow. *Organizational Analysis : a Sociological View*. Brooks-Cole and Tavistock, 1970.
21. A. Rapoport. N-person game theory. Technical report, University of Michigan, 1970.

22. S. Russell. *Artificial Intelligence : a Modern Approach*. Prentice Hall Engineering, Science & Math, 1994.
23. T. Sandholm, K. Larson, M. Andersson, O. Shehory, and F. Tohme. Coalition structure generation with worst case guarantees. *Artificial Intelligence*, 1999.
24. T. Sandholm and V. Lesser. Coalitions among computationally bounded agents. *Artificial Intelligence*, Special issue on Economic Principles of Multiagent Systems(94(1)):99–137, 1997.
25. T. W. Sandholm and V. R. Lesser. Coalition formation among bounded rational agents. In *Proc. of IJCAI-95*, pages 662–669, Montréal, 1995.
26. J-C. Scheid. *Les Grands Auteurs en Organisation*. Dunod, 1990.
27. O. Shehory and S. Kraus. Coalition formation among autonomous agents: Strategies and complexity. *Lecture Notes in Artificial Intelligence*, From Reaction to Cognition, C. Castelfranchi and J. P. Muller (Eds.)(957), 1995.
28. O. Shehory and S. Kraus. Task allocation via coalition formation among autonomous agents. In *Proc. of IJCAI-95*, pages 655–661, Montreal, August 1995.
29. O. Shehory and S. Kraus. A kernel-oriented model for autonomous-agent coalition-formation in general environments : Implementation and results. In *Proc. of AAAI-96*, pages 134–140, Portland, Oregon, August 1996.
30. O. Shehory and S. Kraus. Methods for task allocation via agent coalition formation. *Artificial Intelligence*, 101(1-2):165–200, 1998.
31. O. Shehory and S. Kraus. Formation of overlapping coalitions for precedence-ordered task-execution among autonomous agents. In *Proc. of ICMAS-96*, pages 330–337, Kyoto, Japan, 1996.
32. O. M. Shehory, K. Sycara, and S. Jha. Multi-agent coordination through coalition formation. In M. Singh A. Rao and M. Wooldridge, editors, *Lecture Notes in Artificial Intelligence*, volume Intelligent Agents IV - 1365, pages 143–154. Springer, 1997.
33. Y. Shoham and M. Tennenholtz. On the synthesis of useful social laws for artificial agent societies. In *Proc. of AAAI 92*, pages 276–281, 1992.
34. H. A. Simon. *Theories of Bounded Rationality*. Decision and Organisation. 1972.
35. R. G. Smith. The contract net protocol : High-level communication and control in a distributed problem solver. In *IEEE Transaction on Computers*, volume 29 (12), pages 1104–1113, 1980.
36. F. W. Taylor. *Principles of scientific management*. 1911.
37. G. Vauvert and A El-Fallah Seghrouchni. Coalition formation for egoistic agents. In *Proc. of MAMA '2000*, Wollongong (Australia), December 2000.
38. M. P. Wellman. A market-oriented programming environment and its application to distributed multicommodity flow problems. *Journal of Artificial Intelligence Research*, 1:1–23, 1993.
39. G. Zlotkin and J. S. Rosenschein. Coalition, cryptography, and stability : Mechanisms for coalition formation in task oriented domains. In *Proc. of AAAI94*, pages 432–437, Seattle, Washington, 1994.

Annexe: the Algorithm of Parallel Diffusion

In [39], agents broadcast their encoded informations, then when they have received encoded informations from all the others, they diffuse the key. Using this diffusion, agent can cheat: in a system with three agents a , b and c , a sends its preferences to b and c , b to a and c , but c only to a ; a has received all the preferences, it sends its key to b and c , which makes it possible c to send to b another preference.

These authors supposed that agents did not cheat too much, and improve this mechanism to prevent any fraud sending and receiving information in the same process. Agents send their encoded preferences with a private key, then when each one has receipted all the preferences, they diffuse an acknowledgment of delivery. Each agent diffuses his key only when he has received all acknowledgments. This diffusion solves the problem, because if one carries out the same scenario here, the cheating agent will not be able to change its encoded message any more. It could of course send a wrong key, but the fraud would be visible.

Algorithm 3 Parallel Diffusion of a data θ in \mathcal{A}

```

for all  $a \in \mathcal{A}$  do
   $\theta^* \leftarrow \text{Encrypt}(\theta, \text{key})$ 
   $\mathcal{B} \leftarrow \mathcal{A} \setminus \{a\}$ 
   $a.\text{broadcast}(\theta^*, \mathcal{B})$ 
   $a.\text{receipt}(\theta^*, \mathcal{B})$ 
   $a.\text{broadcast}(\text{Ack}, \mathcal{B})$ 
   $a.\text{receipt}(\text{Ack}, \mathcal{B})$ 
   $a.\text{broadcast}(\text{key}, \mathcal{B})$ 
end for

```
