A DISTRIBUTED ALGORITHM FOR COALITION FORMATION AMONG E-COMMERCE AGENTS

GUILLAUME VAUVERT AND AMAL EL FALLAH – SEGHROUCHNI
Laboratoire d'Informatique de Paris Nord – UPRES-A 7030 – Institut Galilee
Universite Paris 13 – 99, av. J-B Clement – 93430 Villetaneuse – France
{guillaume.vauvert, elfallah}@lipn.univ-paris13.fr

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 repartition domain. This paper proposes an open and distributed protocol based on exchange of preferences computed using qualitative and quantitative criteria: agents will agree on coalitions to form in order to fulfill tasks. We are going to prove that our protocol converges to reach consensus. Experimentation shows that the most rigid strategy is not optimal and that higher competition leads to easier consensus.

1 Introduction

In economic context, rational agents are lead to cooperate in several situations: 1) agents cannot perform tasks by themselves; 2) other agents are more efficient in performing tasks; and 3) working on the task collaboratively will increase benefits or reduce its costs[1]. To increase the efficiency of task achievement, agents may work jointly[2,3,4,5] and may form coalitions viewed as groups of agents whom have decided to cooperate in order to carry out a common task[1].
Suitable to dynamic environments, coalition formation is usually studied from three perspectives, considered independently even if they are not: 1) coalition structure generation (partitioning or covering the set of 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[6]).

Many coalition formation approaches exist, but address different problems in different domains. Sarit Krauss has proposed a classification of works in coalition formation[7] that we are going to extend to emphasize our criteria. The set of main criteria is about the domain. Distributed authority, communication and negotiation are always considered:
–Individual goals[8] *vs* Common goal[1,9,9] (social welfare maximizing).
–Self-interested[8,10] *vs* Altruistic.

−Only pure software agents *vs* pure software with people agents.
−Known rationality (group[1,11,6], personal[8,10], coalition[9,8]) *vs* unknown.
−Bounded rationality[12,3].
−Positive externalities (cooperation)[8] *vs* Negative (competition).
−Number of agents: a dozen[1], a hundred, thousands.
−Untractable size of solution space [10,13] *vs* Small space.
−Defined and agreed protocols[10]) *vs* non pre-defined protocols.
−Static *vs* evolutionary evaluation of incomes.
−Common *vs* individual evaluation of incomes.
−Costly[3] *vs* costless computation.
−Independent tasks[9].
−Satisfy the more tasks as possible[9] *vs* satisfy all tasks.
−Enough competences and agents to solve problem.
−Dynamicity: agents appear/disappear and task arrive constantly[1,4].
−Transferable resources (more beneficial coalition,[9,10]) *vs* no transferability.
−Monetary system for side-payment[10].
−Set partitioning[10] *vs* set covering[14].
−CFG[10,13,5,12,2] *vs* non-CFG.
−Super-additive[5,8,2], sub-additive[12,2], no additivity (most of cases).

In e-commerce, self-interested agents deal with selfish goals and in DPS, altruistic agents deal with common goals. 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, ...).

As shown in[12], 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[3,12].

In this paper, we focus on self-interested agents acting in an economic context. They have individual goals (to increase their incomes) and might be pure software agents or interface for human, and then no strategy is assumed and rationality is bounded (autonomy and rationality deeply studied in[15]).

The protocol we propose is assumed to be known and agreed by agents, but they are completely autonomous: protocols take 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 a part of a task). 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 subtasks) and coalitional value may depend on non-member actions: this may be taken into

account by a modification of solution space and of subtasks 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 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.
Experimentations assume that agent may fulfill subtasks in different coalitions, but the protocol run with a partition (reduction of the solution space). The optimality of the solution have no sense here, because it depends on agents viewpoint. However, chosen solution is legitimate, because no agent is favored. In this defined context, we propose a protocol that take into account strong autonomy[15], weak rationality[15] and heterogeneity to reach a consensus about a subtask repartition.
This article is organized as follows: section 2 formalizes the concepts we define in order to solve the consensus problem. Section 3 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 4 discusses experimentation and provides some of our most significant results. Finally, section 5 concludes the paper.

2 Coalition Formation

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 4), agents may be flexible. If they aren't enough, 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[16].

2.1 Formalization

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

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.*
An ***agent*** $a \in \mathcal{A}$ *is defined by:* $a = \langle C, strategy \rangle$*, where* $C \subset \mathcal{C}$*, and strategies contain competences computation (see 2.2).*
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* $s = \langle C, p \rangle, C \subset \mathcal{C}, ps \in \mathcal{P}$*, 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 (used by agents to compute his preferences).*
A ***competence*** $c \in \mathcal{C}$ *is a single item which represents what is required to be carried out by an agent. A sub-task can require more than one competence.*
A ***profit*** $p \in \mathcal{P}$ *is used as an income, but only to simplify agents internal calculations:* $\mathcal{P} \subset \mathbb{R}_+^*$*. However, the type of profits independence implies that any unit could have been used.*

Example 1 *Agents = arlines:* $\mathcal{A} = \{EUropeanAirlines, USAirlines, \ldots\}$*.*
A task = a flight: $\mathcal{T} = \{New\ York\text{-}MAdrid\ (via\ PAris\ and\ LYon),\ \ldots\}$*.*
Each flight: needs competences: authorization to do a national stage, passengers capacity, range of action: $EUA = \langle \{autEU,\ MidC,\ ShrtR\} \rangle$*; provides incomes:* $\mathcal{P} = [0, 10000]$ *and* $NY - M = \langle \{NY \rightarrow P, P \rightarrow L, L \rightarrow M\}, 8000 \rangle$*.*

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 (in the meaning given below)* $\delta \in \Delta$ *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 the solutions are indifferent.*

Example 2 $\sigma_{15} = [NY \rightarrow PA_2 \hookrightarrow WOA, L \rightarrow M \hookrightarrow BUA, P \rightarrow MO \hookrightarrow EUA, \ldots]$*.*
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* $(v_t)_{t\in\mathbb{N}} \in V$ *is an application* $\mathcal{A} \to \Delta$.

Definition 5 (History) *A history* $h \in H$ *is a sequence of sights. A history* $h = (v_t)_{1\leq t\leq T}$ *represents all preferences exchanged between turn* 1 *and* T.

An alliance is a set of agents and behave like a single one. A member have the representative role: he send alliance's preference to outside agents.

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 special role (see 2.3), with the constraint that an agent can belong to only one alliance.*

To compute the alliance's preference, the representative needs a function of agregation, which is difficult to find [17].

Definition 7 (Alliance Preferences Computation) *An alliance preferences computation APC is an application* $\Lambda \to \Delta$. *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.*

2.2 Strategy of a member agent

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

Definition 8 (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.*

Definition 9 (Preferences Computation)

−Independent Preferences Computation: $IPC \in \Delta$.

−Dependent Preferences Computation: $DPC : H \to \Delta$, $h \mapsto \delta$.

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.*

2.3 Strategy of a representative agent

Definition 10 (Criteria)

−Releasing Switch-over Proposal Criterion (criterion used to decide when to propose to release to switch-over mode): $RSPC : H \to \{False, True\}$.

−Releasing Switch-over Acceptance Criterion (criterion which decides to accept or not to switch to release mode): $RSAC : H \to \{False, True\}$.
−Alliance Formation Proposal Criterion (gives a list of agents to which to propose to form an alliance): $AFPC : h \mapsto (\lambda_1, \ldots, \lambda_n)$, where $AFPC(h) = \emptyset$ is allowed and is interpreted by "the agent doesn't want to form an alliance".
−Alliance Formation Acceptance Criterion (allows to answer to alliance formation propositions): $AFAC : H, A \to \{False, True\}$.

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 diminish computation complexity, only loops of length 3 or less are detected and to simplify computations, $RSAC = RSPC$. Let $d : \Delta \times \Delta \to \mathbb{R}$ 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.*

3 The algorithm of consensus protocol

Each agent may play several roles within the system. The organizer sends datas and manages inscriptions and turns. The supervisor prevents agents to send different preferences to each agent (information can not be used before others thanks to a parallal diffusion[16]) by asking agents what preferences they have sent and received (penalty may be paid by culprits). The candidate receives tasks to fulfill and decides to take part in or not: if he does, he becomes an alliance of one member (himself) and the representative of this alliance. The member receives and sends his preferences when asked by the representative. The representative has been defined in section2.3 and his algorithm is given below.

The representative's algorithm plays a leading role. Each representative has a list of interlocutor's $InterList \subset \mathcal{A}$ initialized with the list of the candidates. The following algorithm is carried out by each representative a_i in a distributed way. In switch-over mode, representatives decide which alliances are going to merge (using $AFPC$ and $AFAC$); if no alliance desires to merge, the system choses them.

Termination. In order to be able to guarantee that the process terminates, we have to assume that the criteria of switch-over mode releasing checks the existence of a loop: if the same situation occurs twice (this case will necessarily happen), then an alliance is formed. 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 (preference use real numbers). To escape

Algorithm 1 Representative: Main

receive("$IndPref$",AllianceMembers)
$h \leftarrow ParallelDiff(IndPref, InterList)$
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",)
  if $\forall a \in InterList$, **receive**("proposition to switch-over mode",a)
    then call **switch-over mode**
  receive("$DepPref$",AllianceMembers)
  $h \leftarrow ParallelDiff(DepPref, InterList)$
  send("$DepPref$",AllianceMembers)
end while

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 11 (Pseudo-equality) *Let ϵ a small real, δ and δ' two preferences and $v_t =$ and $v_{t'}$ two sights. We shall say that:*
$-\delta$ and δ' are pseudo-equal ($\delta \simeq \delta'$) if $\forall \sigma \in \Sigma$, $|delta(\sigma) - delta'(\sigma)| < \epsilon$;
$-v_t$ and $v_{t'}$ are pseudo-equal ($v_t \simeq v_{t'}$) if $\forall a \in \mathcal{A}$, $|v_t(a) \simeq v_{t'}(a)| < \epsilon$.

Definition 12 (A cycle-like in a history) *A history $h = (v_t)_{1 \leq t \leq T}$ contains a cycle-like if $\exists (\tau_1, \tau_2) \in [[1, T]]^2$, $\tau_1 \neq \tau_2$ such that $v_{\tau_1} \simeq v_{\tau_2}$.*

Definition 13 (A CFP detects cycle-like) *A CFP $\langle \mathcal{A}, \mathcal{T}, \mathcal{S}, \mathcal{C}, \mathcal{G} \rangle$ detects cycle-like if (h contains a cycle-like $\Rightarrow$ ($\exists a_0 \in A$ such that $a_0.RSPC(h) = True \wedge \forall a \in A$, $a.RSAC(h) = True$)). In other words, a CFP detects cycles-like if at least one agent detects it and all then accept to change mode.*

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

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. As number n of agents and number k of solutions aren't infinite, the number of sights not pseudo-equal is finite ($2nk/\epsilon$). Finally, after $2kn(n-1)/\epsilon$ turns at worst, there is consensus. ∎

Complexity. Complexity depends in particular on the number of possible solutions which is directly related to the problem datas. Let us assume that our system contains n agents and that each of them is able to process a portion of $1/m$ of the tasks; then a task has on average n/m agents able to carry it,

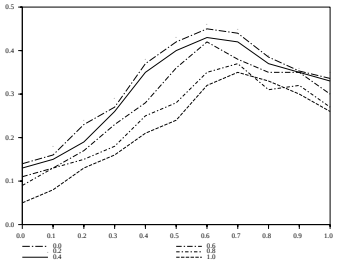

Figure 1. Income / strategy

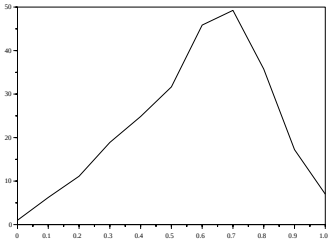

Figure 2. Number of turn / strategy

what gives $k = (n/m)^s$ solutions. In the most general case, our algorithm does not make it possible to change class of complexity, but experimentations show that with alliance formation, the turn number is bounded.

4 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 agent a at turn t: $\delta_a(t) = (1 - w(t)) \times \delta_a(0) + w(t) \times \sum_{b \in A} \delta_b(t-1)/|A|$, where $w(t) = e^{-\alpha t}$. This weight 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, each curve represents the strategy of the population (from 0.0 =rigid to 0.0 =flexible strategy). Results are the average of a large amount of experimentations (350). As expected, agent's income begin to increase, 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. That should lead agent to choose flexible strategies.
Fig.2 shows that when more agents are rigid, consensus is hardly reached. If 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 subtask (competition level) on the incomes (no Figure) and on the number of turns (Fig.3). As expected, when competition increases, incomes decrease and consensus become more difficult to reach.

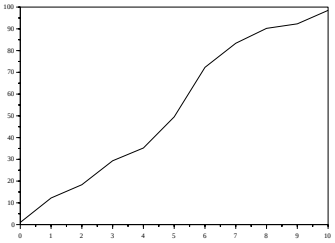

Figure 3. Turn / agents per subtask

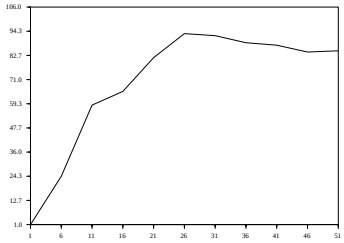

Figure 4. Turn number / $|\mathcal{A}|$

As the number of agents increases (Fig.4), there are more and more agents able to fulfill subtasks 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.

5 Conclusion

From the point of view of deployment of M.A.S. in an economic context, it is necessary to consider weakly rational, strongly autonomous and heterogeneous agents. To arrive, within this framework, to form coalitions, we propounded an open, distributed and egalitarian protocol based on an exchange of preferences computed with quantitative and qualitative criteria since dependent on the specific strategies of the agents. Moreover, we set up releasing procedures thanks to the flexible concept of alliance to avoid the system paralysis. We have shown that, with this protocol, to be extremely rigid is not optimal and high competition leads to a faster consensus.
The next step is to lower complexity by reducing the size of exchanged preferences (using Taboo to find quickly good solutions) and the number of turn (by using models of others).

References

1. O. 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.

2. 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.
3. T. W. Sandholm and V. R. Lesser. Coalition formation among bounded rational agents. In *Proc. of IJCAI-95*, pages 662–669, Montréal, 1995.
4. O. Shehory and S. Krauss. Formation of overlapping coalitions for precedence-ordered task-execution among autonomous agents. In *Proc. of ICMAS-96*, pages 330–337, Kyoto, Japan, 1996.
5. S. P. Kepchel. Forming coalitions in the face of uncertain rewards. In *Proc. of AAAI94*, pages 414–419, Seattle, Washington, 1994.
6. A. Rapoport. N-person game theory. Technical report, Michigan Univ., 1970.
7. S. Kraus. Negotiation and cooperation in multi-agent environments. *Artificial Intelligence*, 94(1-2):79–98, 1997.
8. O. Shehory and S. Kraus. Coalition formation among autonomous agents: Strategies and complexity. *Lecture Notes in A.I.*, From Reaction to Cognition, C. Castelfranchi and J. P. Muller (Eds.)(957), 1995.
9. O. Shehory and S. Kraus. Task allocation via coalition formation among autonomous agents. In *Proc. of IJCAI-95*, Montreal, August 1995.
10. 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, 1996.
11. J. C. Harsanyi. *Rational Behavior and Bargaining Equilibrium in Games and Social Situations*. Cambridge University Press, 1977.
12. 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.
13. T. Sandholm, K. Larson, M. Andersson, O. Shehory, and F. Tohme. Coalition structure generation with worst case guarantees. *Artificial Intelligence*, 1999.
14. O. Shehory and S. Kraus. Methods for task allocation via agent coalition formation. *Artificial Intelligence*, 101(1-2):165–200, 1998.
15. G. Vauvert and A. El Fallah Seghrouchni. Coalition formation among strongly autonomous and weakly rational agents. In *Proceedings of MAAMAW'2001*, Annecy, France, May, 2-4 2001.
16. G. Vauvert and A. El Fallah Seghrouchni. Coalition formation among egoistic agents. In *Proceedings of MAMA'2000*, Wollongong, Australia, December 11-13 2000.
17. K. Arrow. *The Origins of the Impossibility Theorem*, chapter 1. Elsevier Science Publishers B. V., Amsterdam, 1991.