Maximal Clique Based Distributed Group Formation for Autonomous Agent Coalitions

Predrag Tosic*, Gul Agha
Open Systems Laboratory, Department of Computer Science
University of Illinois at Urbana-Champaign, USA
{p-tosic, agha}@cs.uiuc.edu

Abstract

We present herein a fully distributed algorithm for group or coalition formation among autonomous agents. The algorithm is based on a distributed computation of maximal cliques (of up to pre-specified size) in the underlying graph that captures the interconnection topology of the agents. Hence, given the current configuration of the agents, the groups that are formed are characterized by a high degree of connectivity, and therefore high fault tolerance with respect to node and link failures. We also briefly discuss how our basic algorithm can be adapted in various ways so that the formed groups satisfy the requirements ("goodness" criteria) other than mere strong inter-group communication connectivity. We envision various variants of our basic algorithm to prove themselves useful subroutines in many multi-agent system and ad hoc network applications where the agents may repeatedly need to form temporary groups or coalitions in an efficient, fully distributed and online manner.

Keywords: *distributed algorithms, distributed group formation, multi-agent systems, autonomous agents, agent coalitions*

1 Introduction and Motivation

Multi-agent systems (MAS) are characterized, among other properties, by (i) considerable degree of autonomy of individual computing entities or processes (agents) and, at the same time, the fact that (ii) each agent has a local, that is, in general, incomplete and imperfect "picture of the world". Since in MAS there is either no central control, or at best only a limited central control, and the individual agents have to both think and act locally, genuinely distributed algorithms are needed for the agents to effectively coordinate with one another. MAS pose a number of challenges to a distributed algorithm designer; most of these challenges are related to various aspects of the agent coordination problem [1, 18]. In order to be able to effectively coordinate, agents need to be able to *reach consensus (or agreement)* on various matters of common interest. Two particularly prominent and frequently encountered consensus problems are those of *leader election* (e.g., [6, 15]) and *group formation*.

Group formation is an important issue in distributed systems in general, and MAS in particular. Given a collection of computational and communicating agents, the goal in distributed group formation is that these agents, based on their local knowledge only, decide how to effectively split up into several groups, so that each agent knows what group(s) it belongs to.

There are several critical issues that a MAS designer needs to address in the context of (distributed) group formation. First, what is the right notion of a group in a given setting? That is, how is the quality of a particular group configuration measured, so that one can say that one grouping of agents into coalitions is better for that agent and/or for the whole system than another? Second, a distributed group formation mechanism - that is, a distributed algorithm that enables agents to effectively form groups or coalitions - needs to be provided. Third, groups and each agent's knowledge about its group membership need to be maintained and, when needed, appropriately updated. Another important issue is whether the groups are allowed to overlap, i.e., whether an agent is allowed to simultaneously belong to two or more groups. Variants of these basic challenges

*Contact author. Phone numbers: 217-244-1976 (work), 217-390-6515 (cell); fax: 217-333-9386 (for calls/faxes from within the U.S.). Mailing address: Department of Computer Science, 1334 Siebel Center, 201 N. Goodwin, Urbana, IL 61801 USA

are quite common in the MAS literature; indeed, these challenges have arisen in our own recent and current work on parametric models and a scalable simulation of large scale ($10^3 - 10^4$ agents) ensembles of autonomous unmanned vehicles on a multi-task mission [5, 16, 17].

Herein, however, we restrict our attention to the second issue mentioned above. We propose a particular mechanism (distributed algorithm) for an effective coalition formation that ensembles of autonomous agents can use as one of their basic coordination subroutines. A need for a distributed, online and efficient group formation may arise due to a number of different factors, such as the geographical dispersion of the agents, heterogeneity of tasks and their resource requirements, heterogeneity of agents' capabilities, and so on. While for small- and medium-scale systems of robots or unmanned vehicles a fully or partially centralized approach to group formation and maintenance may be feasible or even optimal, large scale systems (with the number of agents of orders of magnitude as in [5] or higher) seem to require a fully distributed approach. That is, the agents need to be able to self-organize into coalitions, and quickly reach a consensus on who is forming a coalition with whom. The algorithm we present herein is an attempt to address these challenges shared by many large-scale MAS applications.

We remark that the group formation and the leader election problems are often inter-related. In particular, at least three general approaches to the combined problem of both forming groups and electing group leaders can be identified. One approach is, that groups are formed by an authority from "the outside", and then the agents within each thus formed group are to reach consensus on who is to be the group leader. That is, only the leader election part is distributed. Distributed leader election (once the group structure is already in place) has been extensively studied in various distributed system models (e.g., [6]).

Another approach is to first select leaders (possibly by appointing them from the outside), and then let these leaders agree with one another on how to assign the rest of the agents to groups "belonging" to different leaders. For very large scale systems and dense interconnection topologies, this is likely the most feasible approach due to its scalability.

The third possibility of how to attack the joint group-formation-and-leader-election problem is that the agents in the ensemble self-organize and form groups first, and then, as in the first scenario, agents within each group decide on their leaders. That is, group formation precedes leader election, and both are done in a genuinely distributed manner. We argue that this approach is scalable for large scale MAS provided that the interconnection topology of the ad hoc network the agents are forming (i) is relatively sparse, and (ii) does not change too rapidly. While our algorithm in Section 4 is generic, in designing it we were admittedly motivated by the autonomous unmanned vehicle applications - more specifically, by the micro unmanned aerial vehicles deployed over a sizable geographical area (see [5, 17]).

2 Group Formation in Multi-Agent Systems

Large ensembles of autonomous agents provide an important class of examples where the agents' capability to coordinate and, in particular, to self-organize into groups or coalitions, is often of utmost importance for such systems to be able to accomplish their tasks, or, in some cases, even to merely survive. One can distinguish between two general, broad classes of such autonomous agents. One is the class of agents employed in *distributed problem solving*. The agents encountered in distributed problem solving (DPS) typically share their goal(s). For instance, DPS agents most often have a joint utility function that they all wish to maximize as a team, without any regard to (or sometimes even a notion of) individual payoffs. This joint utility or, more generally, the goal or set of goals assigned to DPS agents, is usually provided by their designer. However, it may not be always feasible - or even possible - that the designer explicitly provide, for instance, how are the agents to divide-and-conquer their tasks and resources, how are they to form groups and elect leaders of those groups, etc. Due to scalability, incomplete *a priori* knowledge of the environments these agents may encounter, and possibly other considerations, instead of "hard-wiring" into his DPS agents explicitly how are the agents to be coordinated, the system designer may choose only to enable the agents with the basic coordination primitives, and leave to them to self-organize and coordinate as the situation may demand. Hence, in many situations the DPS agents may be required to be able to effectively form groups or coalitions in a fully distributed manner.

The second basic type of agents, the *self-interested agents*, are a kind of agents that do not share their goals (and, indeed, need not share their designer). In contrast to the DPS agents, each self-interested agent has its own agenda (e.g., an individual utility function it is striving to maximize), and no altruistic incentives to cooperate with other agents. Even such self-interested, goal-driven or individual-utility-driven agents, while in

essence selfish, may nonetheless still need to cooperatively coordinate and collaborate with each other in many situations.

One class of examples are those agents (such as, e.g., autonomous unmanned vehicles) that, if they do not coordinate in order to resolve possible conflicts, they risk mutual annihilation. Another class of examples are the agents with bounded resources: individually, an agent may lack resources to accomplish any of its desired tasks - yet, if this agent forms a coalition with one or more other agents, the combined resources and joint effort of all agents in such a coalition may provide utility benefits to everyone involved.

For these reasons, group formation and coalition formation are of considerable interest for many different kinds of autonomous agents and multi-agent systems, and, among other, even in those multi-agent systems where the agents do not share a global utility function, and where each agent generally acts selfishly. In particular, efficient fully distributed algorithms for effective group formation are needed. Such algorithms should use only a few communication rounds among the agents, place a very modest computational burden on each agent, and ensure that a distributed consensus among the agents - that is, in the end, who is forming a group with whom - is effectively, reliably and efficiently reached.

We propose herein one such class of distributed algorithms. We describe in some detail the generic, distributed max-clique-based group formation algorithm in Section 4. Various variants of this basic max-clique-based group formation algorithm can be designed to suit the needs of various types of agents, such as, e.g., the classical DPS agents, the self-interested agents, and the resource-bounded agents.

First, we discuss a generic distributed group formation algorithm based on the idea that an agent (node) would prefer to join a group with those agents that it can communicate with directly, and, moreover, where every member of such a potential group can communicate with any other member directly. That is, the preferable groups (coalitions) are actually (maximal) cliques. It is well-known that finding a maximal clique in an arbitrary graph is **NP**-complete in the centralized setting [3,4]. This implies the computational hardness that, in general, each node faces when trying to determine maximal clique(s) it belongs to. However, if the degree of a node (that is, its number of neighbors in the graph) is small (in particular, if it is $O(1)$), then finding all maximal cliques this node belongs to is computationally feasible. If one cannot guarantee that (or does not know if) all the nodes in a given underlying MAS interconnection topology are of small degree, then one has to impose additional constraints in order to ensure that the agents ("nodes") are not attempting to solve an infeasible problem. In particular, we shall additionally require herein that the cliques to be considered - that is, the possible coalitions to be formed - be of up to a certain pre-specified maximum size. Once such coalitions are formed, being cliques, they can be expected to be relatively robust with respect to the subsequent node or link failures in the system.

Second, we outline how this basic maximal clique based algorithm can be fine-tuned so that the formed coalitions are of good quality with respect to criteria other than the mere robustness with respect to link or node failures. In particular, we indicate how, in multi-agent, bounded resource, multi-task environments (as in, e.g., [17]), the maximal clique algorithm can be adjusted so that each agent strives to join a group such that the joint resources of all the agents in the group match this particular agent's needs in additional resources. Such a choice of the group (coalition) would enable the agent to now be able to complete some of the tasks that this agent would not be able to complete with its own resources alone.

A variety of coalition formation mechanisms and protocols have been proposed in the MAS literature both in the context of DPS agents that are all sharing the same goal (as, e.g., in [13]) and in the context of self-interested agents where each agent has its own individual agenda (as, e.g., in [12, 22]). In particular, the problem of distributed task or resource allocation, and how is this task allocation coupled to what coalition structures are (most) desirable in a given scenario [13], are the issues of central importance in our own work on MAS in bounded resource multi-task environments [5, 16, 17]. These considerations have in part also motivated our work, and especially the extensions of the basic max clique based group formation algorithm, where the cost functions or coalition quality metrics, other than each coalition's interconnectivity alone, are used to determine which coalition(s) is (are) most preferred by which agent[1]. However, while in [13] all agents share all of their goals, we assume herein that each agent is self-interested and, in particular, each agent therefore has its own preference ordering on the coalitions that it may consider joining.

Another body of MAS literature related to our work on modeling and simulation of large scale ensembles of

[1]In the present work, due to space constraints, this issue of how to generalize our algorithm to more general and specifically more resource-oriented coalition cost functions will be only briefly touched upon in Section 5. However, such generalizations, and their implementation in our UAV simulator [5], are a high priority agenda in our future work.

UAVs [5, 16, 17] casts the distributed resource allocation problems into the distributed constraint satisfaction (DCS) terms [7, 8]. The importance of DPS in MAS in general is discussed in [21]. However, discussing DCS based approaches to distributed resource or task allocation and coalition formation is beyond the scope of this paper.

3 Problem Statement and Main Assumptions

Our goal is to design a generic, fully distributed, scalable and efficient algorithm for ensembles of autonomous agents to use as a subroutine (or a coordination strategy) with a purpose of efficiently forming (temporary) groups or coalitions. The proposed algorithm is a graph algorithm. Each agent is a node in the graph. As for the edges, the necessary requirement for an edge between two nodes to exist is that the two nodes be able to directly communicate with one another at the time our distributed group formation subroutine is called[2].

The basic idea is to efficiently partition this graph into (preferably, maximal) cliques of nodes. These maximal cliques may also need to satisfy some additional criteria in order to form temporary coalitions. These coalitions are then maintained until they are no longer useful or meaningful. For instance, the coalitions should be transformed (or else simply dissolved) when the interconnection topology of the underlying graph considerably changes, either due to the agents' mobility, or because many old links have died out and perhaps many new, different links have formed, and the like. Another possible reason to abandon the existing coalition structure is when the agents determine that the coalitions have accomplished the set of tasks that these coalitions were formed to address. Thus, in an actual MAS application, the proposed group formation algorithm may need to be invoked a number of times as a coordination subroutine.

The algorithm is sketched in the next section. For this algorithm to work, the following basic assumptions need to hold:

- agents communicate with one another by exchanging messages either via local broadcasts, or in a peer-to-peer fashion;

- communication bandwidth availability is assumed not to be an issue;

- each agent has a sufficient local memory to be able to store all the information that it receives from other agents; this information is cf. made of the lists of neighboring nodes and of the coalitions proposed to this agent - see *Section 4;*

- communication is reliable during the group formation, but, once the groups are formed, this need no longer hold[3];

- each agent has (or else can efficiently obtain) a reliable knowledge of what other agents are within its communication range;

- each agent, or node, has a unique global identifier (heretofore referred to as 'UID'), and it knows its UID;

- there is a total ordering, $\prec$, on the set of UIDs, and each agent knows this ordering $\prec$;

- communication ranges of different agents are identical - in particular, if agent A can communicate messages to agent B, then also B can communicate messages to A.

On the other hand, an agent need not a priori know UIDs of other agents, or, indeed, how many other agents are present in the system.

In addition to its globally unique identifier (UID), which we assume is a positive integer, each agent has two local flags that it uses in communication with other agents. One of the flags is the binary "'decision flag", which indicates whether or not this agent has already joined some group (coalition). Namely, $decision\_flag \in \{0, 1\}$, and the value of this flag is 0 as long as the agent still has not irrevocably committed to what coalition it is joining. Once the agent makes this commitment, it updates the decision flag to 1 and broadcasts this updated flag value to all its neighbors (see below). That is, as long as the decision flag is 0, the agent's proposals of what group it would like to form or join are only tentative. Once the decision flag becomes 1, however, this indicates that the agent has made a committed choice of which coalition to join - and this selection is final[4].

The second flag is the "choice flag", which is used to indicate to other agents, how "happy" the agent is with its current tentative choice or proposal of the group to be formed. That is, the choice flag indicates the level of agent's urgency that its proposal for a particular coalition to be formed be accepted by the neighbors in the interconnection topology to whom this proposal is sent.

[2]We point out that this definition of the graph edges can be made tighter by imposing additional requirements, such as, e.g., that the two agents to be connected by a link also be compatible, for instance, in terms of their capabilities, or that they each provide some resource(s) that the other agent needs, or the like.

[3]As this requirement is still quite restrictive, and considerably limits the robustness of our algorithm, we will try to relax this assumption in our future work.

[4]...at least for this particular invocation of the group formation subroutine.

It takes values $choice\_flag \in \{0, 1, 2, 3\}$. When an agent is sending just its neighborhood list (at round one of the algorithm - see next section), the value of this flag is 3. Else, if the current choice of a coalition C_i proposed by agent i has equally preferable alternatives[5], then i sets $choice\_flag(i) \leftarrow 2$. If i has other available choices of groups it would like to join, yet each of these alternative choices is strictly less preferable to i than the current proposal (e.g., if each other possible group is a maximal clique of strictly smaller size than the maximal clique $C_i^{current}$ that is the agent i's current proposal), then $choice\_flag(i) \leftarrow 1$. Finally, if i has no other alternatives for a coalition proposal (beside the trivial coalition $\{i\}$), then $choice\_flag(i) \leftarrow 0$. Hence, an agent whose current value of the choice flag is equal to 0 is quite desperate that its proposal of a coalition be accepted by those neighboring agents to whom the proposal is directed. In contrast, an agent whose current value of the choice flag is equal to 2 has some equally good alternative choices and can therefore change its mind without being penalized in terms of having to settle for a less preferable coalition.

4 Maximal Clique Based Distributed Group Formation

Now that the assumptions have been made explicit and the notation has been introduced, the stage is set for presenting our distributed maximal clique based coalition formation algorithm. The algorithm proceeds in five major stages, as follows:

Stage 1:
Set $counter \leftarrow 1$.
Each node (in parallel) broadcasts a tuple to all its immediate neighbors. The entries in this tuple are (i) the node's UID, (ii) the node's list of (immediate) neighbors, L(i), (iii) the value of the choice flag, and (iv) the value of the decision flag.

WHILE (the agreement has not been reached) DO

Stage 2:
Each node (in parallel) computes the overlaps of its neighborhood list with the neighborhood lists that it has received from its neighbors, $C(i, j) \leftarrow L(i) \cap L(j)$. Repetitions (if any) among this neighborhood list intersections are deleted; the remaining intersections are ordered with respect to the list size (the ties, if any, are broken arbitrarily), and a new (ordered) collection of these intersection lists (heretofore referred to simply as 'lists') is then formed.
If $counter > 1$ then:
Each node looks for information from its neighbors, whether they have joined a group "for good" during the previous round. Those neighbors that have (i.e., whose $decision\text{-}flag = 1$), are deleted from the neighborhood list $L(i)$; the intersection lists $C(i, j)$ are also updated accordingly, and those $C(i, k)$ for which k is deleted from the neighborhood list $L(i)$ are also deleted.

Stage 3:
Each node (in parallel) picks one of the most preferable lists $C(i, j)$; let $C(i) \leftarrow chosen\ [C(i, j)]$. If the list size is the main criterion, then this means, that one of the lists of maximal length is chosen. The value of the choice flag is set, based on whether an agent has other choices of lists as preferable as the chosen clique, and, if not, whether there are any other still available choices at all.

Stage 4:
Each node (in parallel) sends its tuple with its UID, the tentatively chosen list $C(i)$, the value of the choice flag, and the value of the decision flag, to all its neighbors.

Stage 5:
Each node i (in parallel) compares its chosen list $C(i)$ with lists $C(j)$ received from its neighbors. If a (relatively small, of size not exceeding k) clique that includes the node i exists, and all members of this clique have selected it at this stage as their current group or coalition of choice (that is, if $C(i) = C(j)$ for all $j \in C(i)$), this will be efficiently recognized by the nodes forming this clique. The decision flag of each node $j \in C(i)$ is set to 1, a group is formed, and this information is broadcast by each node in the newly formed group to all of its neighbors. Else, if no such agreement is reached, then agent i, based on its UID and priority, and its current value of the *choice-flag*, either does nothing, or else changes its mind about its current group of choice, $C(i)$ (the latter being possible only if $choice\text{-}flag > 0$, meaning that there are other choices of $C(i)$ that have not been tried out yet that are still available to agent i).

$counter \leftarrow counter + 1$;
END DO $[* \ end\ of\ \ WHILE\ \ loop\ *]$

If $round > 1$ then, at Stage 2, each node looks for the information from its neighbors to find out if any of them have joined a group in the previous round. For those nodes that have (i.e., whose decision flag

[5] In the max. clique context, this means, if there are two or more maximal cliques of the same size, one of which is chosen by an appropriate tie-breaker. This idea readily generalizes, as long as each agent has a partial order of preferences over all the possible coalitions that include this agent.

$dec = 1$), each node neighboring any such already committed node deletes this committed node from its neighborhood list $L(i)$, updates all $C(i,j)$ that remain, and selects its choice of $C(i)$ based on the updated collection of group choices $\{C(i,j)\}$. That is, now all those nodes that have already made their commitments and formed groups are not "in the game" any more, and are therefore deleted from all remaining agents' neighborhood lists as well as the tentative choices of coalitions. (Of course, the only coalition a committed agent is *not* deleted from at this stage is *the coalition* that this agent has just joined).

There are some more details in the algorithm that we leave out for the space constraint reasons. One important technicality is that, in order to ensure that the algorithm always avoids to cycle, once an agent changes its mind about the preferred coalition $C(i)$, it is not allowed through the remaining rounds of the algorithm to go back to its old choice(s). Once no other choices are left, this particular agent sticks to its current (and the only remaining) choice, and waits for other agents to converge to their choices. It can be shown that this ensures ultimate convergence to a coalition structure that all agents agree to. That is, under the assumptions stated in the previous section, the agents will reach consensus on the coalition structure after a finite number of rounds inside the WHILE loop (see also *Appendix*). Moreover, if the maximum size of any $L(i)$ is a (small) constant, then the convergence is fast.

5 Discussion and Extensions

We have outlined a fully distributed algorithm for group or coalition formation based on maximal cliques. This algorithm will be efficient when the underlying graph is relatively sparse, and, in particular, when the sizes of all maximal cliques are bounded by a small constant $k = O(1)$. When this is not the case (or when it cannot be guaranteed to always hold), appropriate restrictions can be imposed "from the outside" to ensure that the algorithm converges, and rapidly. For example, for each node i, a range of possible values (UIDs) of those nodes that the node i is allowed to communicate and form coalitions with can be appropriately specified.

Once the groups are formed, these groups will be tight (as everyone in the group can communicate with everyone else), and, in nontrivial cases, therefore as robust as possible for a given number of group elements with respect to either node or link failures. This is a highly desirable property involving coalitions or teams of agents (robots, autonomous unmanned vehicles, etc.) operating in environments where both the agent failures and the agent-to-agent communication link failures can be expected.

The proposed algorithm can be used as a subroutine in many multi-agent system scenarios where, at various points in time, the system needs to reconfigure itself, and the agents need to form new coalitions (or transform the existing ones) in a fully distributed manner, where each agent would join an appropriate (new) coalition because the agent finds this to be in its individual best interest, and where it is important to agents to form and agree on these coalitions efficiently, rather than wasting too much time and other resources on (possibly lengthy) negotiation.

This algorithm as a coordination subroutine in MAS applications can be expected to be useful only when the time scale of significant changes in the inter-agent communication topology is much coarser than the time scale for the coalitions of agents, first, to form according to the algorithm, and, second, once formed, to accomplish something useful in terms of agents' ultimate goals. We are currently exploring using some version of this algorithm as a coordination strategy subroutine in a scalable software simulation of a system of unmanned autonomous aerial vehicles (UAVs) on a multi-task mission.

To be useful in various MAS applications, such as the above-mentioned UAV simulation, the proposed generic algorithm can be appropriately fine-tuned, so that the coalitions are formed that satisfy quality criteria other than robustness with respect to agent or communication link failures.

Let us assume there is an ensemble of self-interested agents moving around and about in an environment, searching for tasks and possible coalition partners and/or other external resources, and trying to service as many tasks as possible. Each task has a certain value to each agent: an agent increases its utility by servicing some task, and consuming this task's value [16]. However, servicing different tasks requires resources, and an agent may lack sufficient resources to be able to service tasks it would like to complete. Hence, such an agent will have an egotistic incentive [2] to cooperatively coordinate with other agents - and, in particular, to try to form a group or a temporary coalition with other agents. The preferred partners in such a coalition would be those agents that would provide sufficient resources for the desired tasks to be completed. Hence, in the algorithm above, a refinement of the criterion of "goodness" (quality) of different groups that can be formed is needed. So, for example, at Stage 3, among the available lists of intersections $C(i,j)$, the agent i may want to choose one where the sum of resources of

all the agents in this list is equal to, or exceeds (but preferably by not too much) the resource requirements of the particular task the agent i desires to service[6].

6 Concluding Remarks

We have proposed herewith a generic algorithm for distributed group formation based on maximal cliques of modest sizes. Among the existing distributed group formation algorithms, we argue that our algorithm is particularly suitable for dynamic coalition formation and transformation in multi-agent systems whose underlying graph structures ("topologies") change frequently, yet not too rapidly. In particular, we find this algorithm, or its appropriately fine-tuned variants, to be a potentially very useful subroutine in many multi-agent system applications, where the interconnection topology of the agents often changes so that the system needs to dynamically reconfigure itself, yet where these topology changes are at a time scale that allows agents to (i) form coalitions, and (ii) do something useful while participating in such coalitions, before the underlying communication topology of the system changes so much as to render the formed coalitions either obsolete or ineffective. We intend to explore and test the applicability and practical usefulness of the proposed algorithm as a coordination strategy for various MAS applications in general, and in the context of our scalable simulation of autonomous unmanned vehicles on a complex, multi-task mission [5, 16, 17], in particular.

Acknowledgment: Many thanks to Nirman Kumar and Reza Ziaei (Open Systems Laboratory, UIUC) for many useful discussions. This work was supported by the *DARPA IPTO TASK Program* under the contract *F30602-00-2-0586*.

Bibliography

[1] N. M. Avouris, L. Gasser (eds.), "Distributed Artificial Intelligence: Theory and Praxis", Euro Courses Comp. & Info. Sci. vol. 5, Kluwer Academic Publ., 1992

[2] D. H. Cansever, "Incentive Control Strategies For Decision Problems With Parametric Uncertainties", Ph.D. thesis, Univ. of Illinois Urbana-Champaign, 1985

[3] T. H. Cormen, C. E. Leiserson, R. L. Rivest, "Introduction to Algorithms", MIT Press, 1990

[4] M. R. Garey, D. S. Johnson, "Computers and Intractability: a Guide to the Theory of NP-completeness", W. H. Freedman & Co., New York, 1979

[5] M. Jang, S. Reddy, P. Tosic, L. Chen, G. Agha, "An Actor-based Simulation for Studying UAV Coordination", Proc. 15th Euro. Symp. Simul. (ESS 2003), Delft, The Netherlands, October 2003

[6] N. Lynch, "Distributed Algorithms", Morgan Kaufmann Publ., Wonderland, 1996

[7] P. J. Modi, H. Jung, W. Shen, M. Tambe, S. Kulkarni, "A dynamic distributed constraint satisfaction approach to resource allocation", in "Principles and Practice of Constraint Programming", 2001

[8] P. J. Modi, W. Shen, M. Tambe, M. Yokoo, "An asynchronous complete method for distributed constraint optimization", Proc. AAMAS 2003

[9] J. von Neumann, O. Morgenstern, "Theory of Games and Economic Behavior", Princeton Univ. Press, 1944

[10] J. Rosenschein, G. Zlotkin, "Rules of Encounter: Designing Conventions for Automated Negotiations among Computers", The MIT Press, Cambridge, Massachusetts, 1994

[11] S. Russell, P. Norvig, "Artificial Intelligence: A Modern Approach", 2nd ed., Prentice Hall Series in AI, 2003

[12] O. Shehory, S. Kraus, "Coalition formation among autonomous agents: Strategies and complexity", Proc. MAAMAW'93, Neuchatel, 1993

[13] O. Shehory, S. Kraus, "Task allocation via coalition formation among autonomous agents", Proc. 14th IJCAI-95, Montreal, August 1995

[14] R. G. Smith, "The contract net protocol: high-level communication and control in a distributed problem solver", IEEE Trans. on Computers, 29 (12), 1980

[15] G. Tel, "Introduction To Distributed Algorithms", 2nd ed., Cambridge Univ. Press, 2000

[16] P. Tosic, M. Jang, S. Reddy, J. Chia, L. Chen, G. Agha, "Modeling a System of UAVs on a Mission", Proc. SCI 2003 (invited session), Orlando, Florida, July 2003

[17] P. Tosic, G. Agha, "Modeling Agents' Autonomous Decision Making in Multiagent, Multitask Environments", Proc. 1st Euro. Workshop MAS (EUMAS 2003), Oxford, England, 2003

[18] G. Weiss (ed.), "Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence", The MIT Press, Cambridge, Massachusetts, 1999

[19] M. Wooldridge, N. Jennings, "Intelligent Agents: Theory and Practice", Knowledge Engin. Rev., 1995

[20] M. Yokoo, K. Hirayama, "Algorithms for Distributed Constraint Satisfaction: A review", AAMAS, Vol. 3, No. 2, 2000

[21] M. Yokoo, "Distributed Constraint Satisfaction: Foundation of Cooperation in Multi-agent Systems", Springer, 2001

[22] G. Zlotkin, J.S. Rosenschein, "Coalition, cryptography and stability: Mechanisms for coalition formation in task oriented domains", Proc. AAAI'94, Seattle, Washington, 1994

[6] At this stage, however, we need to do more work on formalizing some of these variants of the algorithm, and, in particular, determine the criteria that would still ensure, under general assumptions similar as before, that the agents would efficiently converge to an agreement on the coalition structure.

Appendix: Pseudo-code for Max-Clique-Based Distributed Group Formation

Notation:

$i :=$ the i-th agent (node) in the system (say, $i = 1, ..., n$)
$V(i) :=$ the i-th node's UID
$N(i) :=$ the list of neighbors of node i
$L(i) :=$ the extended neighborhood list (i.e., $L(i) = N(i) \bigcup \{i\}$)
$C(i,j) = L(i) \bigcap L(j)$
$C(i) :=$ the group of choice of node i at the current stage (i.e., one of the $C(i,j)$'s)
$choice(i) :=$ the choice flag of node i
$dec(i) :=$ the decision flag of node i

Maximal Clique Based Distributed Group Formation Algorithm:

Step 1:
DOALL $i = 1..n$ (in parallel, i.e., each node i carries the steps below locally)
send $[V(i), L(i), choice(i) = 3, dec(i) = 0]$ to each of your neighbors
END DOALL

WHILE (not all agents have joined a group) DO

Step 2:
DOALL $i = 1..n$
FOR all $j \in N(i)$ DO [$\star$ check if $dec(j) = 1$ $\star$]
if $dec(j) == 1$ then delete j from $N(i), L(i)$, and $C(i,j'), \forall j' \in N(i) - \{j\}$
END DO [$\star$ end of FOR loop $\star$]
FOR all $j \in N(i)$ DO [$\star$ FOR all remaining (undeleted) indices j $\star$]
compute $C(i,j) \leftarrow L(i) \bigcap L(j)$
END DO [$\star$ end of FOR loop $\star$]
END DOALL

Step 3:
DOALL $i = 1..n$
pick $C(i,j)$ s.t. $|C(i,j)|$ is of max. size (not exceeding the pre-specified threshold, k):
$C(i) \leftarrow C(i,j)$;
if more than one such choice, set $choice(i) \leftarrow 2$;
else (if there are other choices $C(i,j')$ but only of smaller sizes)
set $choice(i) \leftarrow 1$;
else (if node i has no alternatives left for a non-trivial coalition that would include i)
set $choice(i) \leftarrow 0$;
END DOALL

Step 4:
DOALL $i = 1..n$
send $[V(i), C(i), choice(i), dec(i) = 0]$
END DOALL

Step 5:
DOALL $i = 1..n$
compare $C(i)$ with $C(j)$ received from one's neighbors;
if there exists a clique $\{i, j_1, j_2, ..., j_l\}$ such that $C(i) = C(j_1) = C(j_2) = ... = C(j_l)$
then set $dec(i) \leftarrow 1$ (an agreement has been reached);
broadcast group $G = (i, j_1, j_2, ...j_l)$ and decision flag value $dec(i) = 1$ to all neighbors
else based on i and the priority (as defined by the relation $\prec$ on the set of nodes $1, 2, ..., n$)
either DO NOTHING
or change your mind: $C(i) \leftarrow newchoice$ (from the list of candidate groups)
END DOALL
END DO [$*$ *end of* $WHILE$ *loop* $*$]