

A self-organization process for communication management in embedded multiagent systems

Jean-Paul Jamont and Michel Occello
Université Pierre Mendès-France , Laboratoire LCIS
51 Rue Barthélémy de Laffemas, 26000 Valence, France
{jean-paul.jamont,michel.occello}@iut-valence.fr

Abstract

This paper deals with a multiagent self-organization process aiming to give adaptive features to distributed embedded systems involving intelligent agents in open real world. We propose a formal description of this model of self-organization.

1 Introduction

Multiagent systems are especially adapted for designing real world artificial complex systems. Embedded multiagent systems are composed by autonomous hardware/software agents that have to achieve perception tasks and control tasks in respect of energy policies. In this context, at a global level, the system must provide an energy efficient communication management. Wireless agents have to adapt their behaviour according to their independent energy resources. In such multiagent systems, agents cannot have all the other society members perception areas. However each agent must help the others to communicate together : this constitutes a multihopping communication. The routing process is thus distributed among all the agents.

Through the MWAC (Multi-Wireless-Agent-Communication) model, we propose an innovative approach for communication management open embedded multiagent systems in the context of wireless communication. This communication layer is based on a multiagent collective intelligence analysis. The cooperation, collaboration and negotiation capabilities of multiagent systems allow the agents which evolve in an open system to increase the overall efficiency of the whole system.

In a first part, we introduce the main difficulties of communication management in wireless multiagent systems. We present in a second part our model exploiting emergence features. We explain how we take benefit from a totally decentralized approach and how the inherent multiagent emer-

gence features are exploited. To conclude, we give an insight of the real world application using the MWAC model.

2 Communication management in wireless multiagent systems

In wireless multiagent systems, agent interacts together and with their environment, for example, to carry out a measure, to realize complex cooperative tasks... and to help other agents to communicate together. Generally these agents have autonomous energy resources.

These constraints must be taken into account in order to optimize communication management (the energy devoted to communication constitutes an important part of the agent energy cost). The energy parameter is important in the sense that it can create internal faults or that it can influence other parameters like the emission range. Furthermore the environment can be hostile and can cause agent internal faults.

The open nature of these networks can be another source of errors. In fact, insertion and departure of the agent occur randomly and often unpredictably. Furthermore, in the case of mobile devices the infrastructure of systems are not persistent and the global data monitoring must be organized from local observation.

Networks of wireless autonomous entities have a distributed routing process. The associated routing protocols are centred on the flooding techniques which consist in sending messages to all the members of the network to be sure the receiver gets the message: the associated power cost is very high. Generally the routing protocols are a compromise between the control traffic reduction and the latency in finding the route to a destination.

Message routing through a multiagent approach aims to decentralize the decision and the knowledge without trying to build a global network model nor a global route model. A few works reaching the same objectives show that the approach is interesting. We can quote the ActComm [1] military project for which the routing of information is es-

sential: it aims at studying the communication management between a soldier team and a military camp via a satellite. Work described in [5] presents a mobile agent approach to optimize the time of message transmission. This work uses agents' affinity which are similar to link state in some ad-hoc routing protocols. The problem described in [6] is very similar to our problem but the approach is very different since the used technique is based on distributed stochastic algorithms. In [4] some algorithms are described for routing message with a swarm cooperation model.

3 Our solution based on the emergence

A multiagent system is a set of agents situated in a common environment, which interact and attempt to reach a set of goals. Through these interactions a global behavior, more intelligent than the sum of the local intelligence of multiagent system components, can emerge. The emergence process is a way to obtain, through cooperation, dynamic results that cannot be calculated in a deterministic way.

3.1 What should emerge?

Our objective is to decrease the energy expense induced by the inherent floodings techniques. For that we will use a group structure inspired by the clusters of the CGSR protocol. Our organizational basic structures are constituted by (see fig 1): one and only one group representative agent (r) managing the communication in its group, some connection agents (c) which know the different representative agents and can belong to several groups, some ordinary members (o) which are active in the communication process only for their own tasks (They don't ensure information relay). With this type of organizational structure, the message path between the source (a) and the receiver (b) is $((a, r), * [(r, c), (c, r)], (r, b))$.

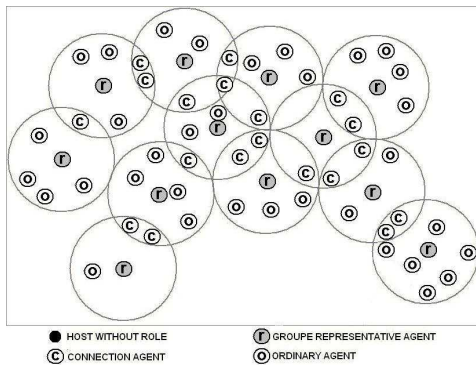


Figure 1. Our organizational structure

The energy saving is obtained owing to the fact that the

flooding is only directed to the representative agent of the groups and to some connection agent. To give an order of idea, a receiver path research with flooding techniques will cost, in the case of a traditional wireless network, a number of emissions equal to the number of stations. In the case of a clustered wireless network, the number of transmitted messages are about twice the numbers of representative agent (all the representative agents are contacted via one connection agent). In our example (fig 1) the cost would be in the first case of 74 messages and in the second of 26 messages.

However, the networks supporting an organizational structure must take care of the maintenance of their routing table. Generally, the adaptive features of these tables come from periodical exchanges between the different nodes. In our approach we do not wish to use this technique to ensure the maintenance of coherence. Indeed, our principle will be "if we do not need to communicate, it is useless to spend energy to ensure the coherence maintenance". However, we will thus use eavesdropping of surrounding agent communications. We extract from these messages exchange knowledge to update our beliefs about our neighbours. Moreover, our self-organization mechanism will integrate a energy management policy. These structures will thus emerge.

3.2 How to make the solution emerge?

We want to obtain an adaptation of our whole multiagent system through the emergence of organizational structures by self-organization based on role allocation modifications. The organization is built according to an exchange of messages between agents. Relations between agents are going to emerge from the evolution of the agents' states and from their interactions. We are only going to fix the organization parameters, i.e. agents' tasks, agents' roles.

The ideal representative agent is the one having the most important number of neighbours and the most important level of energy. The level of energy is an important parameter in the sense that the representative agent is the most solicited agent in the group from a communication point of view. We use role allocation based self-organization mechanisms involving the representative agent election. Our election function integrates some data on neighbours and energy levels. This function estimates the correlation between its desire to be the boss and its capacity to access to this position. The organization is modified only when a problem occurs. We do not try to maintain it if we have no communication. In addition to the configuration messages, all agents use eavesdropping. In fact, when some communicating entities (humans, robots etc.) share a common environment they might intercept some messages (broadcasted or not). From this eavesdropping message they can extract some authorised information like the receiver, the sender, the type of message and the packet's path.

We propose here a formalized description of our model. The notation find their sources in the work described in [3].

Identifier. Hosts of the network are modelled by agent. Each agent have an identifier i . We note A_i the agent identified by i .

The multiagent system. The multiagent system Γ is the set of agents $\Gamma = \{A_1, A_2, \dots, A_i, \dots, A_n\}$ with $card(\Gamma) = n$. Our multiagent system is open : hosts can enter or leave the system.

Time. We note \mathbb{T} the ordered set with the operator $<$ and an element $-\infty$ with $\forall t \in \mathbb{T}, t < -\infty$. So $\mathbb{T} = \mathbb{N} \cup \{-\infty\}$.

Groups. 1) An agent group is noted \mathcal{G} . In our organization, a group is identified by its representant Identifier. The group where the representant is A_R is noted \mathcal{G}_R . All groups are part of the system : $\mathcal{G}_R \in \mathcal{P}(\Gamma)$

(2: *intention*) A group has a finite time to live (with a lower and a higher limit). The lower limit is the most interesting (the group birth) : we note $[A_R, t_0]$ the group created by A_R at t_0 with $(A_R, t_0) \in \Gamma \times \mathbb{T}$

(3: *belief in extension*) We note $[A_R, t_0]^{A_j, t_1}$ the set of agents that A_j think members of the group $[A_R, t_0]$ at t_1 .

(4: *extension*) We note $[A_R, t_0]^t$ the set of agents really in $[A_R, t_0]$ at t .

We note $[A_R, t_0]^t$ the group composition \mathcal{G}_R created at t_0 at the given date t . This knowledge can be defined from the belief of the agents: $[A_R, t_0]^t = \{A_j \in \Gamma \mid A_j \in [A_R, t_0]^{A_j, t} \wedge A_j \in [A_R, t_0]^{A_R, t}\}$

Belief. \mathcal{B}_{A_i} φ minds that the agent A_i thinks φ , in other words it thinks that φ is true. To highlight the recursive feature of the group definition given previously, we can note that $(A_j \in [A_R, t_0]^{A_j, t}) \equiv (\mathcal{B}_{A_i} (A_j \in [A_R, t_0]^t))$.

Desir. \mathcal{D}_{A_i} φ minds that the agent A_i desires φ , in other words it wants to verify φ .

Knowledge. \mathcal{K}_{A_i} φ minds that the agent A_i knows φ .

Roles. (1) We note $role(A_i, t)$ the function that returns the role of the agent A_i at the date t with $(A_i, t) \in \Gamma \times \mathbb{T}$. A role can be R_R for a representant, R_C for a connection agent and R_S for an ordinary member. When an agent is initialized, he has no role. The function $role$ can then return \emptyset to signify that the agent has no role.

(2: *simplification of writing*) We note $role_t(A_i)$ the last role taken by A_i .

(3: *choice of a role*) Each agent chooses a role depending on its neighbourhood. So, choosing a role leads to notify the new role to neighbours and modify its knowledge about its own role. So $\mathcal{K}_{A_i}(role(A_i, t_v) = R_R)$ can be understood following different way. Firstly, we learn simply that the agent A_i is representant, but if $\mathcal{K}_{A_i}(role(A_i, t_{v-1}) \neq role(A_i, t_v))$ then the agent A_i has modified his role to be representant.

Power supply. (1) We note $power(A_i, t)$ the function which returns the energy level (a percentage) of the agent A_i at the date t with $(A_i, t) \in \Gamma \times \mathbb{T}$.

(2: *simplification of writing*) We note $power(A_i)$ the current energy level of the agent A_i .

Neighbourhood. We note N_{A_i} the neighbourhood that A_i knows. It is a set of agents in the emission range of the agent A_i not including itself. So, $N_{A_i} \in \mathcal{P}(\Gamma)$.

An agent knows a neighbour by its unique identifier but can access to its role and its group ($\forall A_j \in N_{A_i}, \mathcal{K}_{A_i} role(A_j) \wedge \mathcal{K}_{A_i} group(A_j)$) with $group$ the function defined similar to $role$ but $group(A_j)$ returns the group identifier of the agent A_j . We can notice that if $\mathcal{K}_{A_i} [A_R, t_0]^{A_j, t_1}$ then $\mathcal{K}_{A_i} group(A_j) = R$. The reciprocal is not true because there is an uncertainty about the time.

Formalized description of the role attribution Choosing a role depends firstly on its neighbourhood (*basic algorithm*). However, because our power level is low, an agent can not desire to be representant (*energetic constraint*). The decision processes of agents are not synchronized. Two neighbours can take the same decision at the same time. It is possible that two close agents choose a representative role: there is a *representant conflict* which must be detected and corrected. It is possible to have two closer groups which don't include a connection agent between them: there is an *inconsistency* which must be detected and corrected.

We begin by focusing on our algorithm which allows to the agent A_i to choose a role in function of its neighbourhood N_{A_i} .

Basic algorithm 1) There is no neighbor : the concept of role doesn't make sense. ($N_{A_i} = \emptyset \Rightarrow (\mathcal{K}_{A_i}(role(A_i) = \emptyset))$)

2) Neighbors exist ($N_{A_i} \neq \emptyset$).

$(\mathcal{K}_{A_i}(card(\{A_j \in N_{A_i} \mid role(A_j) = R_R\}) = 0) \Rightarrow (\mathcal{K}_{A_i}(role(A_i) = R_R))$

$\mathcal{K}_{A_i}((card(\{A_j \in N_{A_i} \mid role(A_j) = R_R\}) = 1) \Rightarrow (\mathcal{K}_{A_i}(role(A_i) = R_S))$

$\mathcal{K}_{A_i}((card(\{A_j \in N_{A_i} \mid role(A_j) = R_R\}) > 1) \Rightarrow (\mathcal{K}_{A_i}(role(A_i) = R_C))$

Energy constraint Generally, the role of representative or connection make that the agents take an active part in the management of communications. From this fact, consumption of energy is higher. So, $(power(A_i) < trigValue) \Rightarrow (\mathcal{K}_{A_i}(role(A_i) = R_S))$.

Detecting and correcting a representant conflict (1: Conflict detection) An agent A_i detects a conflict with other agents if $\mathcal{K}_{A_i}(N_{A_i} \neq \emptyset) \wedge \mathcal{K}_{A_i}(role(A_i) = R_R) \wedge \mathcal{K}_{A_i}((card(\{A_j \in N_{A_i} \mid role(A_j) = R_R\}) > 1)$.

(2: *Conflict correction*) A_i has detected a conflict with other agents. he sends a *ConflictRepresentativeResolution* message (cf. the interaction aspect) to its representative neighbours. This message contains the score of the agent A_i . The agents, which receive this message, calculate their own score. Agents with an inferior score leave their role

and choose another. An agent with a better score sends its score to its neighbours.

An example of *score* function can be simply expressed. The following function supports an agent with a high energy level and a significant neighbour (the interest is to have dense groups in order to limit the flooding volume). $score(A_i) = power(A_i).card(N_{A_i})$

Detecting and correcting an inconsistency (1: Inconsistency detection) An inconsistency can be detected only by one representative starting from beliefs of one of its members. This detection needs an interaction between an agent A_i and its representative A_R (message *VerifyNeighborGroupConsistency*).

The agent A_i will send the list of the groups of its neighbourhood of which it does not know if its representative knows the proximity.

We define $N_{A_i,L} = \{A_k \in N_{A_i} \mid role(A_k) = R_C\}$.

A connection agent is member of many groups, so, if $A_L \in N_{A_i} \wedge role(A_L) = R_C \wedge [A_\alpha, t_\alpha]^{A_L, t_\alpha} \wedge [A_\beta, t_\beta]^{A_L, t_\beta}$, then $\mathcal{K}_{A_i}(group(A_L) = \alpha)$ et $\mathcal{K}_{A_i}(group(A_L) = \beta)$.

We define $\zeta_{A_i} = \{A_j \in N_{A_i} \mid group(A_j) \neq group(A_i) \wedge (\exists A_k \in N_{A_i,L} / (group(A_k) = group(A_j) \wedge group(A_k) = group(A_i)))\}$.

The inconsistency is found by A_i if $card(\zeta_{A_i}) = 0$. The representative agent A_R of A_i receives a message with ζ_{A_i} . For $\forall A_n \in \zeta_{A_i}$, if $card(\{A_y \in N_{A_R,L} \mid group(A_y) = n\}) = 0$ then there is a real inconsistency.

(2: Inconsistency correction) In this case several strategies can be used. We judge that if a path with a low energy cost is available, one will support a stability of the organization to a reorganization. A search for path towards one of the groups soft will thus be sent with a TTL (Time to Live) relatively low.

If a path exists, the organization does not change. If not, the representative proposes to A_i , if $role(A_i) = R_C$, to be a representant (*ISuggestYouToBeRepresentative*). The agent A_i can refuse to become representative (if its energy level is too low) but in all the case, the representant A_R leaves its role.

4 Conclusion

The real implementation : the ENVSYS project The ENVSYS project aims to instrumentate an underground river system [2]. We have chosen for sensors a classical three-layers embedded architecture (physical/link/applicative).

We use the physical layer which is employed by NICOLA system (a voice transmission system used by the French speleological rescue teams). This layer is implemented in a digital signal processor rather than a full analogic system. Thereby we can keep a good flexibility and

we will be able to apply a signal processing algorithm to improve the data transmission. The link layer used is a CAN (Controller Area Network) protocol stemming from the motorcar industry and chosen for its good reliability. The applicative layer is constituted by the agents' system.

These agents are embedded on autonomous processor cards. These cards are equipped with communication modules and with measuring modules to carry out agent tasks relative to the instrumentation. These cards supply a real time kernel which allows multi-task software engineering for C515C microcontroller. We can then quite easily implement the parallelism inherent to agents and satisfy the real-time constraints.

Synthesis We presented in this paper a hybrid software/hardware application for the management of an environmental agents network. We proposed a multiagent analysis of this system and detailed how we use collective features to make the system adaptive. The innovative aspect of this work stands in the use of multiagent self-organization techniques based on the emergence of structures. The concept of emergence is usually quite difficult to defend in an applicative real world context. The MWAC model is implemented as a generic middleware. We work currently on another project using this middleware to coordinate a robot society working in a real world (a manufacturing firm).

All these works aim at contributing to show that artificial intelligence mechanisms (as self-organization) can lead to interesting results and can improve classical techniques.

References

- [1] R. Gray. Soldiers, agents and wireless networks: A report on a military application. In Proceedings of the 5th International Conference and Exhibition on the Practical Application of Intelligent Agents and Multi-Agents, 2000.
- [2] J.-P. Jamont, M. Ocelllo, and A. Lagrèze. A multiagent system for the instrumentation of an underground hydrographic system. In Proceedings of IEEE International Symposium on Virtual and Intelligent Measurement Systems, May 2002.
- [3] F. Legras and C. Tessier. Lotto: group formation by overhearing in large teams. In Proceedings of the 2nd International Joint Conference on Autonomous Agents and Multiagent Systems, pages 425–432, Australia, July 2003. ACM.
- [4] P. Arabshahi et al. Adaptive routing in wireless communication networks using swarm intelligence. In Proceedings of the 9th AIAA Int. Communications Satellite Systems Conference, pages 1–9, April 2001.
- [5] R. Choudhury et al. Topology discovery in ad-hoc wireless networks using mobile agents. In Proceedings of the 2nd ACM Workshop on Mobile Agents for Telecommunication Applications, pages 1–16. Springer, Sep 2000.
- [6] Zhang W. et al. Distributed problem solving in sensor networks. In AAMAS'02, pages 15–19, 2002.