Multiagent and Grid Systems – An International Journal 1 (2005) 73–87
IOS Press

Solving inaccessibility in multi-agent systems by mobile middle-agents

David Šišlák*, Michal Pěchouček, Martin Rehák, Jan Tožička and Petr Benda
Department of Cybernetics, Czech Technical University in Prague, Technická 2, Prague 6, 166 27 Czech Republic

Received 3 May 2005
Revised 6 June 2005
Accepted 8 June 2005

Abstract. This paper analyzes the problem of communication inaccessibility in the multi-agent systems. Apart from listing the main reasons for existence of communication inaccessibility, it suggest a complete inaccessibility model and metrics. Various inaccessibility solutions are presented, together with their applicability in the environments with different degrees of inaccessibility, as verified by experiments. Major contribution of this paper is in suggesting a novel solution to solving inaccessibility based on community of autonomous, migrating middle-agents. A distributed algorithm for dynamic allocation of the middle-agents in a network of partially inaccessible agents is proposed. The suggested solution is supported by a set of experiments comparing the distributed and centralized algorithm for middle-agents allocation.

Keywords: Multi-agent system, ad-hoc network, inaccessibility, middle-agents

1. Introduction

The main idea of multi-agent system is to compose large system of small autonomous agents that cooperate and communicate with each other. It is expected that each agent can interact with any other agent using a reliable communication infrastructure. However, in many situations an agent may become inaccessible from the other members of the multi-agent community due to a failure of the communication links, network traffic overload, dynamic changes of the network topology in a mobile ad-hoc networks, agent leaving the communication infrastructure for accomplishing a specific mission, agent failing to operate, etc. Therefore, agent's inaccessibility in a multi-agent system is problem with high practical importance.

The paper introduces the problem of inaccessibility in multi-agent systems with its formal definition (Section 1) and analysis of possible reasons for inaccessibility (Section 1.2). The Section 2 presents quantities for measuring inaccessibility based on random-graph theory [2]. In the following section the solutions for inaccessibility are discussed and an extended concept of the middle agent is presented. The proposed algorithm for optimal allocation of the middle agents is based on virtual payments combined with social dominance [15, 22] model, as detailed in Section 3.2. Experiments addressing the states of inaccessibility in the real multi-agent system, comparison of possible solutions and the optimization by extended concept of the middle agent architecture in mobile ad-hoc network are presented in Section 4.

1.1. Inaccessibility definition

Communication between agents (according to [24]) is viewed as a complex action with no specific result that can be proved to happen. This uncertainty is what distinguish the multi-agent system from other software systems and what brings about the emergent and unforseen behavior of the agents' communities.

Inaccessibility in multi-agent system is a situation when two agents (who are aware about each other)

*Corresponding author. E-mail: sislakd@feld.cvut.cz.

ISSN 1574-1702/05/$17.00

intend to a exchange a message while their intention fails to be implemented. Using BDI dynamic logic the agent's inaccessibility is defined as follows

$$\begin{aligned}&\mathsf{inaccessible}(A,B) \Leftrightarrow (\mathsf{Bel}A\ \mathsf{agent}(B)) \wedge \\ &\quad (\mathsf{Int}\ A\ \{\mathsf{send}\ A\ B\ \alpha\ \texttt{c-act}\}) \Rightarrow \\ &\quad \neg(\mathsf{Bel}B\ (\mathsf{Int}A\ (\mathsf{Int}B\ \texttt{c-act})),\end{aligned} \tag{1}$$

where `c-act` is an abstract communicative act and the predicate $\mathsf{agent}(B)$ denotes existence and knowledge of communication availability (e.g. IP address, ACL language) of the agent B.

In Definition 1 there are three conditions, which must be satisfied at the same time for inaccessibility to happen in the system:

- **nonexisting agent** – it may happen that one agent believes in existence of another agent who has deregistered from the community. For example the other agent has terminated or its host machine was shut down. This is why the agents need have the most up-to-date white-page knowledge (agent's name, its location, ACL language it uses) about the recipients,
- **intention to communicate** – agent's inaccessibility can be hardly identified if there is no intention to interact. It does not mean that an agent cannot be inaccessible if there is no attempt to send it a message. This sort of "hidden" inaccessibility does not influence the system performance,
- **failure to communicate** – a result of sender's failure to communicate a speech is receiver unawareness about the sender's intention to communicate. This occurrence is very often hard to identify. Most interaction protocols include a "confirmation" speech acts that acknowledge successful message delivery.

1.2. Reasons for inaccessibility

There is no guarantee of complete accessibility of the communication links between agents in highly distributed systems. It may happen that the communication links are operating when the request is sent and they become inaccessible just in the time moment of sending the confirmation. The sender assumes that the message did not reach the recipient, while the receiver relies in its further operation on the senders knowledge of the receiver acknowledgement. One may suggest sending another acknowledgement (acknowledgement of the previous acknowledgment). It is easy to be seen that for absolute certainty that the interaction happened correctly it is needed an infinite number of acknowledgements, which is not feasible.

There are several different reasons why an agent can become inaccessible by the others:

- **agent physical mobility and ad-hoc networks** – agents may physically leave the community. This can happen in the situations where intelligent agents reside on mobile platforms such as cell phones, PDAs or any other IP-addressable devices carried e.g. by a person or mobile vehicle. The inaccessibility can arise when the mobile platform uses for communication short range radio links (e.g. IEEE802.11 [1]). Therefore an agent could communicate only with other agent running on the other platforms located nearby, in the radio range. The movement of agent carriers may be dependent or completely independent from the agents communication requirements,
- **unreliable communication links** – often communication networks are unreliable. This case includes communication link failure and traffic overload. Inaccessibility can be characterized by periodical, singular or permanent communication lags and drop-outs,
- **agent code migration** – when an agent is in transit, it cannot handle incoming messages and therefore it is inaccessible. Software agents may need to migrate to another physical location, for example to cooperate reliably with agents located there. Such migration may lengthen existing communication links with other agents, increasing their potential to fail in the future.
- **adversarial environment** – In a hostile environment the situations may occur when the agents refuse to communicate in order to keep their existence undisclosed. Such an agent is then regarded as inaccessible. There is a variation of this scenario, when agent can receive the information while cannot send any information, e.g. when broadcasting a signal may disclose agents physical location, while receiving is a passive activity that does not reveal agent's position.
- **information inaccessibility** – A specific situation is when agents agree to communicate only specific type of information (semi-private see [11]) and keep the private information confidential. Private information is regarded as inaccessibility. This scenario is typical for the information fusion and intention modelling problems.

While one can speculate and predict, there is no guarantee on the future agents' behavior (which is given by their autonomy). There are also agents that works "off-line" and still wish to be acknowledged as full scope members.

2. Measuring inaccessibility

Let $\overline{\vartheta} \in [0;1]$ denote a **measure of inaccessibility**. This measure is dual to the **measure of accessibility** – $\vartheta \in [0;1]$, where $\vartheta + \overline{\vartheta} = 1$. Complete accessibility is denoted as $\vartheta = 1$ and complete inaccessibility is denoted as $\vartheta = 0$. This measure is consistent with the circuit reliability defined in [8]. In the following text the agents' accessibility is mostly described while the inaccessibility is its complement.

2.1. Agent-to-Agent accessibility

In the following the random graph theory [4] is used in order to describe some general properties of inaccessibility in multi-agent systems. Random graph theory has been recently successfully used for theoretical studies of complex networks [2].

In this context, the multi-agent community is represented as a graph. The agents are represented by nodes and available communication links – connections where the information exchange is possible – by edges. Unlike in the general case of agents inaccessibility, the random graph theory works with an assumption that all edges are present with the same probability p. In agent domain, this probability is represented by **link accessibility**: $p = \vartheta$.

The ϑ link accessibility can be determined in two ways. Firstly as ϑ_t – **direct accessibility** defined using the uptime of the link connecting two agents:

$$\vartheta_t = \frac{t_{\mathrm{acc}}}{t_{\mathrm{inacc}} + t_{\mathrm{acc}}}, \tag{2}$$

where t_{acc} denotes the amount of time when communication is possible while t_{inacc} denotes time when agents are disconnected.

Similarly, accessibility can be measured as a function of communication requests (ϑ_m):

$$\vartheta_m = \frac{|m| - |m_{\mathrm{fail}}|}{|m|}, \tag{3}$$

where $|m|$ denotes the total number of messages sent and $|m_{\mathrm{fail}}|$ the number of messages that failed to be delivered. In the following ϑ_t is discussed while most conclusions apply equally to ϑ_m. The accessibility measure ϑ_t is symmetrical between entities A and B

$$\vartheta_t(A,B) = \vartheta_t(B,A), \tag{4}$$

while the accessibility measure ϑ_m is not necessarily symmetrical.

The **path accessibility** can be defined exactly in the same manner as direct, but the agent is considered to be accessible even if there is no direct link between the agents and messages are forwarded by other agents along the path from the source to the destination agent. Therefore, the path accessibility is always at least equal to link accessibility. In the next paragraph, the relationship between the link and path accessibility is examined.

2.2. Agent-to-Agent accessibility characteristics

Classical result of the random graph theory is that there exists a critical probability at which large cluster appears. In agent domain, there is a **critical accessibility** – ϑ^{C} such that below ϑ^{C} the agent community is composed of several isolated groups but above ϑ^{C} most of agents become mutually path-accessible. The ϑ^{C} value is represented by the quick growth in the Fig. 1. This observation is similar to a percolation transition known in the field of mathematics and statistical mechanics [18].

Second relevant result of random graph theory is the average length l^* of a path between any two vertices and the diameter l^d of a graph (i.e. maximal distance between any two nodes). It holds [2]:

$$l^* \sim l^d = \frac{\ln(n)}{\ln(\vartheta_t n)}, \tag{5}$$

where n is number of agents. In practice, the length of the path defines how many relays must be used in order to convey a message between the agents A and B.

Table 1 summarizes several results of random graph theory important for study of inaccessibility. There are identified three states of the agents' community. All states defined in the table are can be distinguished in the simulated distributed multi-agent system presented in Section 4.1.

3. Solving inaccessibility

Inaccessibility solutions can be divided into mechanisms providing:

– **relaying** – classical mechanisms, where the message is forwarded across the network,

Table 1
Different cases of accessibility as described by random graph theory

$\vartheta_t n < 1$	**low accessibility**	The network is typically composed of isolated trees. The diameter is equal to the diameter of trees.
$\vartheta_t n > 1$	**transition phase**	A large cluster is formed. The diameter is equal to the largest cluster diameter and if $\vartheta_t n > 3.5$ it is proportional to $\frac{\ln(n)}{\ln(\vartheta_t n)}$.
$\vartheta_t n > \ln(n)$	**complete accessibility**	The graph is probably totally connected and the diameter is very close to $\frac{\ln(n)}{\ln(\vartheta_t n)}$.

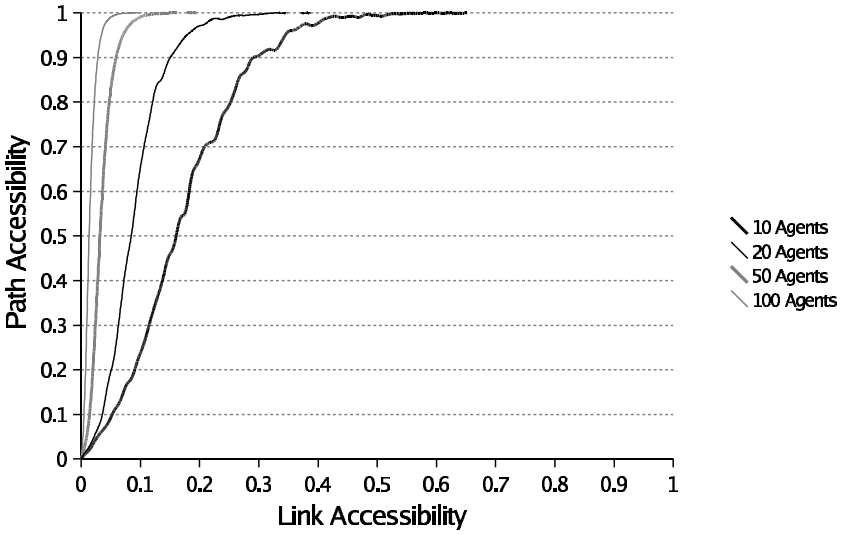

Fig. 1. The dependency of path accessibility (probability of existence path between two agents) on link accessibility. This graph is the same for the link accessibility with or without the symmetry.

– **remote presence** – an ability of inaccessible agent to act remotely, and
– **remote awareness** – providing the inaccessible agent with the knowledge about the inaccessible part of the community and vice-versa.

The most classical solution to the inaccessibility problem is based on relay agents or low-level entities, responsible for setting up a transmission path through other elements when the direct contact between parties is impossible. Such protocols are currently widely implemented for routing in various types of networks, like TCP/IP [17] or on lower levels [23]. There are several factors limiting the use of relayed connection: (i) reduced battery life due to the fact that all the messages must be transmitted several times, (ii) network maintenance overhead, especially in case of mobile networks and (iii) dynamic nature of network topology if the agent platforms are based on moving entities – relaying cost increases as the link maintenance and path-finding in dynamic environment is a non-trivial process. Leontiadis, Dimakopoulos and Pitoura [6] propose solution for synchronization and search in peer-to-peer networks.

Remote presence is usually implemented by various kinds of *middle agent*. In a overview article [21], authors list different types of middle agents – *Matchmakers* and *Brokers* (Facilitators). Matchmakers may provide remote awareness by notifying interested agents about the presence of service providers, while the brokers can act as intermediaries and pass actual service requests between two mutually inaccessible parties. Even if this solution may perform very well in many situations, it may be unusable if middle agents are difficult to find, unreliable, or can not be trusted with private preferences of different parties. Kumar, Cohen and Levesque [9] solve inaccessibility in brokered systems by replacing a single broker by a team of brokers. A multi-agent system remains fully functional as long as there is at least one functional broker in the system. The brokers create new brokers to maintain their number in the system on a specified level despite their failures, thus effectively making the system self-healing. The synchronous operation of multiple agents is requires relatively complex synchronization, without providing any particular advantage. A generic model of the middle agent is suggested and presented

in Section 3.1. *Stand-in agents* are a special type of middle agents, with each stand-in representing a single owner agent. Stand-ins are created by the owner and use cloning and migration to distribute themselves throughout the system. By their nature, they encapsulate partial or complete self and social knowledge of their owner, together with the algorithms working with the knowledge. The fact that stand-ins have a single owner makes resolves potential conflicts of interests and makes the system more transparent – owners can be sure that their agents follow their own goals. However, as in any middle-agent community acting in an inaccessible environment, a synchronization problem occurs. If the stand-ins take commitments on behalf of their owners, appropriate policies must be enforced to avoid overbooking and other conflicts. Adjustable autonomy, similar to the one used in human-robot teamwork is one of the solutions to this problem [14]. On the lower levels, stand-in operations also conform to the middle agent model described in Section 3.1.

Remote awareness is usually assured by agents' *acquaintance models* that collects social knowledge, necessary and optional information which an agent needs for its efficient operation in the multi-agent community. An acquaintance model is a computational model of agent's mutual awareness. There have been various acquaintance models studied and developed in the multi-agent community, eg. *tri-base acquaintance model* [10] and *twin-base acquaintance model* [5]. Social knowledge can be used for making operation of the multi-agent system more efficient. The acquaintance model is an important source of information that would have to be repeatedly communicated otherwise. Besides saving communication traffic the social knowledge aggregated in the acquaintance model can be used in the situations of agents' short term inaccessibility. However, the acquaintance models provides rather '*shallow*' knowledge, that does not represent a complicated dynamics of agent's decision making, future course of intentions, resource allocation or negotiation preferences. This type of information is needed for inter-agent coordination in situation with longer-term inaccessibility.

In Section 4, the boundaries of applicability of the solutions above are presented. As stated in [13], remote-awareness and remote presence results are equivalent under some assumptions, and therefore the performance of stand-in based solution is compared, representing middle agents and social knowledge, with relaying. However, besides simple applicability, we must be also interested by the communication efficiency of

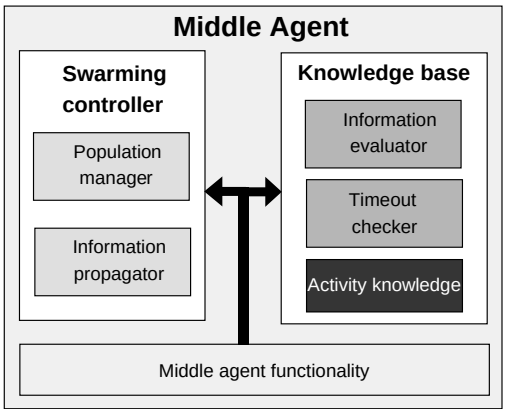

Fig. 2. Middle agent architecture.

the adopted solution. The optimization is crucial in the situations where the path accessibility is almost complete, while the link accessibility remains limited. Many real-world systems operate in this mode and most of them are resource constrained [7].

Therefore, the problem of optimal distribution of middle agents in the distributed dynamic multi-agent system is addressed in this section. Also message flow optimization between these agents is studied to ensure optimum balance between efficiency and redundancy. Presented algorithm is used with particular type of middle agent (Section 3.1), but it can be integrated with all the technologies previously discussed.

It shall be noted that in the real dynamic systems, the accessibility values are rarely homogenous. As analyzed above (see Section 2.2), clusters of agents often tend to move together and encounter other agents directly or via relays. Therefore, the optimization algorithm must (i) ensure the existence of middle agents and message transmissions to *optimally connect* the multi-agent system, while (ii) being *local* in the sense that middle agents shall be able to operate with the local environment information only, without the need for the central coordination. *System adaptivity* (iii) is another crucial factor, as it shall autonomously adapt to current local environment both in time and in place, but the whole algorithm shall be also (iv) *efficient* in the number of messages consumed and computational load. *Community stability* (v) is closely related to efficiency as the overly rapid adaptation can significantly increase computational load of the system.

3.1. Middle agent architecture

Unlike classical middle-agent architectures [20] where the prime functionality is devoted towards

matchmaking and negotiation, the concept of middle agent is extended by the integration of the optimization algorithm described in Section 3.2 to autonomously migrate in the network and clone or destruct individual middle agents.

In Fig. 2 new algorithm component is presented – *swarming controller* in the context of middle agent architecture.

- **Swarming controller** – consists of two modules: **population manager** ensures cloning, migration and destruction of middle agents in the system while the **information propagator** manages information flows through the agent, more specifically the messages or knowledge to transfer or actions to take. The module must balance between two extreme cases of knowledge handling: propagation to all visible targets or no propagation at all. Even if both modules are domain independent, they depend on the domain specific functions included in the knowledge base algorithms. Details of the algorithm used are described in the dedicated Section 3.2.
- **Knowledge base**, a domain specific knowledge structure of the middle agent, consists of three parts: *activity knowledge*, *information evaluator* and *timeout checker*. While the **activity knowledge** contains the domain specific knowledge and the meta-data provided by the propagator, the information evaluator and timeout checker are the algorithms working on this knowledge. The **information evaluator** classifies and indexes the knowledge, so that the index values can be used by information propagator to manage its activity and further propagation. It also evaluates the knowledge usefulness. The **timeout checker** module implements *forgetting* of the activity knowledge.
- **Middle agent functionality** – universal interface between modules and agent platform. It provides fundamental agent functions (clone, migrate and die), message interface and monitoring listeners, as well as original middle agent code. This code depends on the actual type of the middle agent. Via monitoring listeners it notifies modules about visibility of the other nodes, information about accessible other middle agents and also about presence of potential message receiver. Only this part of relay agent needs to be changed to work properly with specific agent platform.

Besides middle agents, the later presented algorithm can be also integrated with simple relaying agents whose functionality is based on sole, uninformed message forwarding.

In the Fig. 3 there is a sample middle agent network: nodes represent fixed locations, lines between nodes means that there exists a communication link between them and residing middle agent is represented by point near the node.

3.2. Swarming controller

As mentioned above one of the key issues in middle agent operation is their proper location in the network. The distributed middle agent allocation mechanism uses only locally accessible information. It does so not only minimize the network maintenance communication, but also allows operation in the disruptive or partially inaccessible environment. Locally accessible information is obtained by monitoring middle agents neighborhood – identifying currently visible targets and other middle agents. The algorithm needs to be lightweight and computationally simple, as the middle agent instances can be constrained by the devices they run on. Scalability in space and density shall also be an important property of the targeted solution.

In principle there are two key approaches to controlling the efficiency of the middle agents allocation:

1. *forward swarming control* – where the middle agent migrates its clone only to the locations with higher possibility of future inaccessibility and higher interaction expectancy and
2. *backward swarming control* – where the middle agents dispatch their clones to every reachable destination and the useless ones are eliminated in the future following the inaccessibility real situation.

Each of the approaches has its pros and cons. The forward swarming control is computationally efficient, as it tries to minimize the number of middle agents in the system and prevent the possible swarming explosion. This is why that approach seems to be particularly suitable for domains with high scalability and operational efficiency requirements. On the other hand, the backward swarming control has got an important advantage. This approach is substantially more domain independent, demands less knowledge about the environment nature and is more robust, as it doesn't *explicitly* use any prediction about the future of the community.

The *backward swarming* is here selected, because this approach is more robust and domain independent.

Abstract criteria of the system quality defined in the introduction were also formulated in a precise manner, with descending priority:

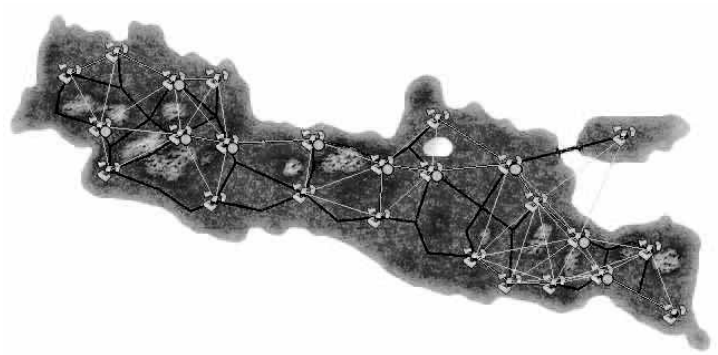

Fig. 3. Middle agent network in a system with inaccessibility.

– Provide connection between any two system elements through the minimum number of middle agents.
– Minimize the number of middle agents in the system.
– Minimize the number of messages for system operation and/or knowledge maintenance.

Population manager is driven by a biology inspired algorithm. Social dominance and altruism models [15, 22] were successfully used to partition the group of agents into those who work for the good of the community and the others, who profit from the altruism of the first group. During the experiments with rats, it was determined that a exactly the sufficient number of individuals behaves in an altruistic manner to optimize the *whole group* fitness. They bring the food and share it with the others, who only consume. This behavior is formalized by a simple mathematical model formulated in [22].

To ensure the target coverage, middle agents can be reproduced in the system using two main propagation strategies:

– *full flood fill* – any middle agent initiates full flood filling reproduction strategy when it identifies a new unserved knowledge target in its reach. To decide whether the target is really new, all agents keep the set of served targets, that includes both the other middle agents and knowledge final users. Target is removed from the set when it is not used for specified duration – *forget time*. In practice, the middle agent is cloned to every visible node where it is not running yet if the new knowledge target is not reachable from current position of the agent. Created clones further clone themselves to new locations without existing middle agents using the same cloning termination condition: target reachability. Only this simple strategy can ensure that the middle agent network will reach the target. Using random walk instead of flood fill is possible, but not advisable, because the random walk does not guarantee finding the target, as known from Pólya's random walk theorem [3] – Pólya considered a d-dimensional array of lattice points where a point moves to any of its neighbors with equal probability. He asked whether given an arbitrary point A in the lattice, a point executing a random walk starting from the origin would reach A with Probability 1. Pólya's surprising answer was that it would for $d = 1$ and for $d = 2$, but it would not for $d \geqslant 3$. In later work he also analyzed two points executing independent random walks and also at random walks satisfying the condition that the moving point never passed through the same lattice point twice,
– *bounded flood fill* – this is depth-limited version of the previous reproduction strategy. After the initiation, the middle agents are successively cloned only to depth specified by `FloodFillDepth` constant. This reproduction strategy is triggered by a local accessibility change when the source agent holds relevant, non-expired knowledge. Application of this mechanism can identify shorter paths enabled by the accessibility change or can deliver the knowledge to the isolated cluster by the middle agent on the mobile node.

Both flooding strategies are time limited. There is specified constant *flood duration* during which the middle agent retains reproduction intention. When this period expires, the agent no longer reproduces until the new reproduction is started by the agent itself or the others.

To keep the number of middle agents close to optimum, the population manager contains also the meth-

ods that decrease the number of middle agents in the system:

- In *random duels* the attacking middle agent randomly selects an adversary between accessible agents and launches an attack with force proportional to its profit during specific period, as determined by *information propagator* (see bellow). Besides the attacks force, the attack also includes the information about its *active target* set at the time of the attack. The attacked agent evaluates the attack and decides whether it won or lost. If the attacked agent loses, it removes itself from the system; losing attacker is not penalized for the attack. Attack evaluation compares the *active targets* first and when one is a subset of the other, its owner loses the fight. When the sets are identical, force of the attack decides the fight – stronger agent wins. Active target set size is evaluated differently for new and old agents. For the young agents that are not yet completely adapted to the environment, the set contains all directly accessible targets, while it contains only the really used ones. Besides this advantage, the youngest agents benefit from the immunity period, during which they can not lose a fight while attacked.
- In contrast to the previous case, the *uselessness detection* is an individual process. The middle agent can remove itself while it is isolated from the rest of the community and has no access to relevant knowledge anymore.

Information propagator manages knowledge propagation and use in the system. This component uses virtual payments to reward the other agents for the knowledge, receives payments from the others for the information provided and generates the profit also from acting on behalf of the represented agent. Each agent optimizes its profit, ensuring the overall information flow efficiency. More specifically, middle agents reward the information received from the others in function of information usefulness and redundancy – the first agent from which they receive the information receives significant payment, while the subsequent information is rewarded less. On the other hand, the transmitting the information to other agents is not free of charge for the agent – it must carefully decide to which agents it propagates which information. The decision is taken in function of the previous experience (and current network status) with similar knowledge, and the knowledge source and potential target are important, domain independent similarity criteria. Besides these criteria, the knowledge may be enhanced with metadata specifying to which agent it has been already provided. To make the system more robust, the decision to which nodes it sends the information is probabilistic – agents may therefore send the knowledge to the less rated directions to find better paths in the system. Payment for the information is transmitted as a reply to the knowledge update message.

Historical data (represented as probabilities assigned to knowledge characteristics, origins and targets) that are used to identify the targets to which it sends the information are periodically updated and the old data is discarded. When a new target appears, the initial message transmission probability is set to the level derived automatically from the current network state of the evaluating agent.

The domain specific functions that evaluate the usefulness of the knowledge, indexable knowledge characteristics and rewards for actions are provided by the domain-specific knowledge base.

It shall be kept in mind that the knowledge is not only propagated by messaging in the network of middle agents, but also carried by the agents created during the reproduction process. This is especially important for the communities where the accessibility is relatively low, or where the agents are clustered.

4. Experiments

All experiments with middle agents and relays in mobile ad-hoc network described below were executed using ***A*-globe** platform [16] with inaccessibility support. Agents are running on simulated nodes and can migrate between accessible nodes. In the system, both fixed and mobile nodes are used.

4.1. Measuring accessibility

In the first experiment, several series of measurements with variable communication range were conducted on the system, resulting in a series of data with increasing link and path accessibility. The relation between the two and its conformance to the theoretical prediction from Section 2.2 are presented in this paragraph.

In Fig. 4, three major states of the community from the accessibility point of view can be identified, as defined in Section 2.1. At first, before the communication radius reaches 60, static community members are isolated and information is not transmitted (see first row

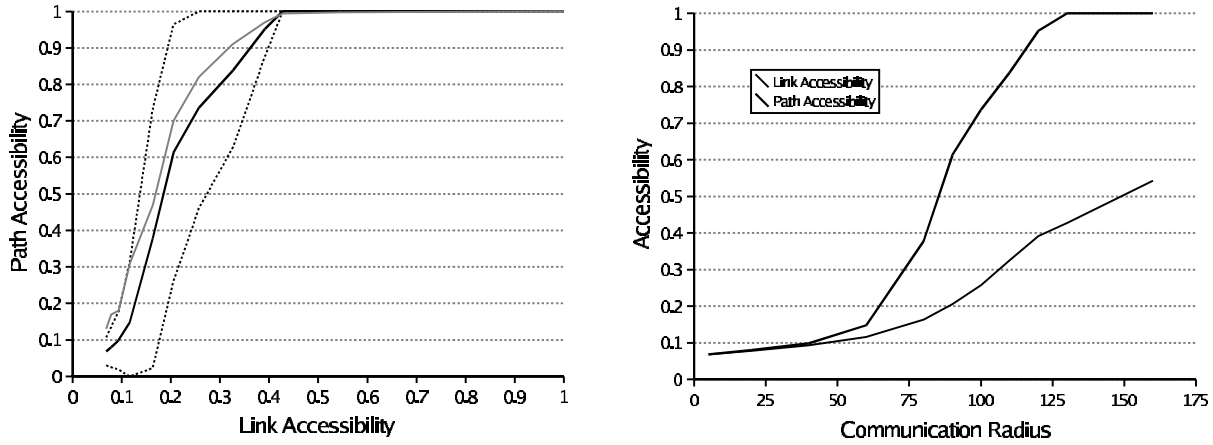

Fig. 4. *Left:* The dependency of probability of existence path between two agents and link accessibility in the test scenario. The dot lines show average deviation of values. The gray thin line shows theoretical value for random graph with 10 nodes (see Fig. 1). *Right:* The dependency of link and path accessibility on communication radius.

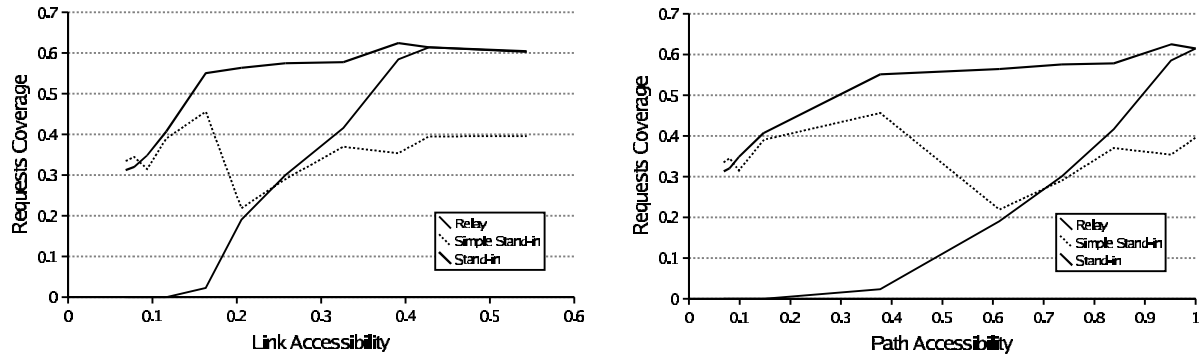

Fig. 5. The average requests coverage of three presented inaccessibility solutions and different accessibility settings.

of Table 1), but only carried by moving entities. In this state, path accessibility is not significantly different from link accessibility. Therefore, probability of relaying is almost negligible.

Then, with increasing communication radius, larger connected components do start to appear, covering several static and mobile entities and allowing the use of relaying over these portions of the graph. This phase appears around the percolation threshold, that can be observed above radius of 80, corresponding with link accessibility of 0.2. This state is characterized by important variability of connected components. Due to the dynamic nature of the system, these components are relatively short-lived, resulting in a high variability of the system. Path accessibility in the community may be described by relation (see second row of Table 1).

In the last state, when communication radius is above 120 and link accessibility reaches 0.4, the entities become almost completely connected. This state of the community is described by relation (see third row of Table 1). System properties does not change when the radius and link accessibility are further increased.

4.2. Comparing the solutions

After having determined the extent of inaccessibility in the system, the inaccessibility effects on the system performance will be studied. The system performance is measured as a ratio of requests that were completed successfully. Zero value means complete failure, when no goods were transported, while 0.6 implies that the maximum system capacity was reached.

The experiments compare the performance of relaying with the use of stand-in agents, selected as representatives of all middle agent and knowledge-based solutions (see Section 3). In the stand-in case, two limit implementations are compared – in the first implementation, stand-ins don't share the knowledge with the

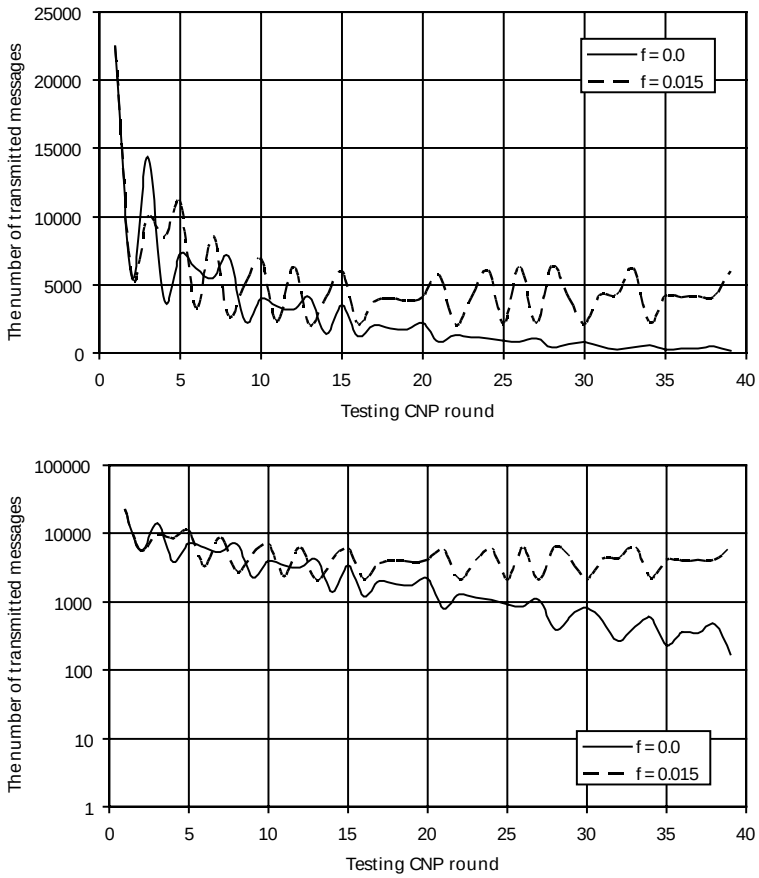

Fig. 6. Message reduction using different forget parameter (f) value in fully connected environment with 24 nodes. Bottom chart has logarithmic scale on Y-axis.

others in their visibility range and swap the information only when two agents encounter each other. In the case with communication, stand-ins share any new information with everyone in the communication range.

In the following graph (see Fig. 5), the relationship between path accessibility and overall system performance can be observed for each of three solutions. Here the average requests coverage for different solution of inaccessibility is presented.

Results do follow the accessibility state partitioning from the previous experiment. It can be seen that relay agents start to be reasonably useful when the link accessibility reaches 0.2, in the middle of the transition phase, well corresponding to the percolation threshold. Performance of isolated, non communicating stand-in remains constant. This is easy to understand, as these agents communicate only locally. They present an optimal solution for disconnected networks, as they require only a small number of messages to function.

On the other hand, performance of interacting community of stand-ins is more than a mere supremum of both previous methods. This is allowed by the dynamic nature of the system, where the stand-ins on mobile entities carry the up-to-date information through the system and spread it in small local communities, but relatively often. Thanks to this approach, the efficiency of system with these stand-ins approaches the optimum level with path accessibility of 0.4, instead of 0.9 for relay agents.

Both relaying and actively communicating stand-ins suffer from the common pain – the need to communicate efficiently and to save the number of messages that conveys the information in the system. Experiment described in the next paragraph addresses this specific issue.

4.3. Efficiency of middle agent network

The second series of experiments address the frequent case when the link accessibility is low, but path accessibility approaches one. In this situation, the ef-

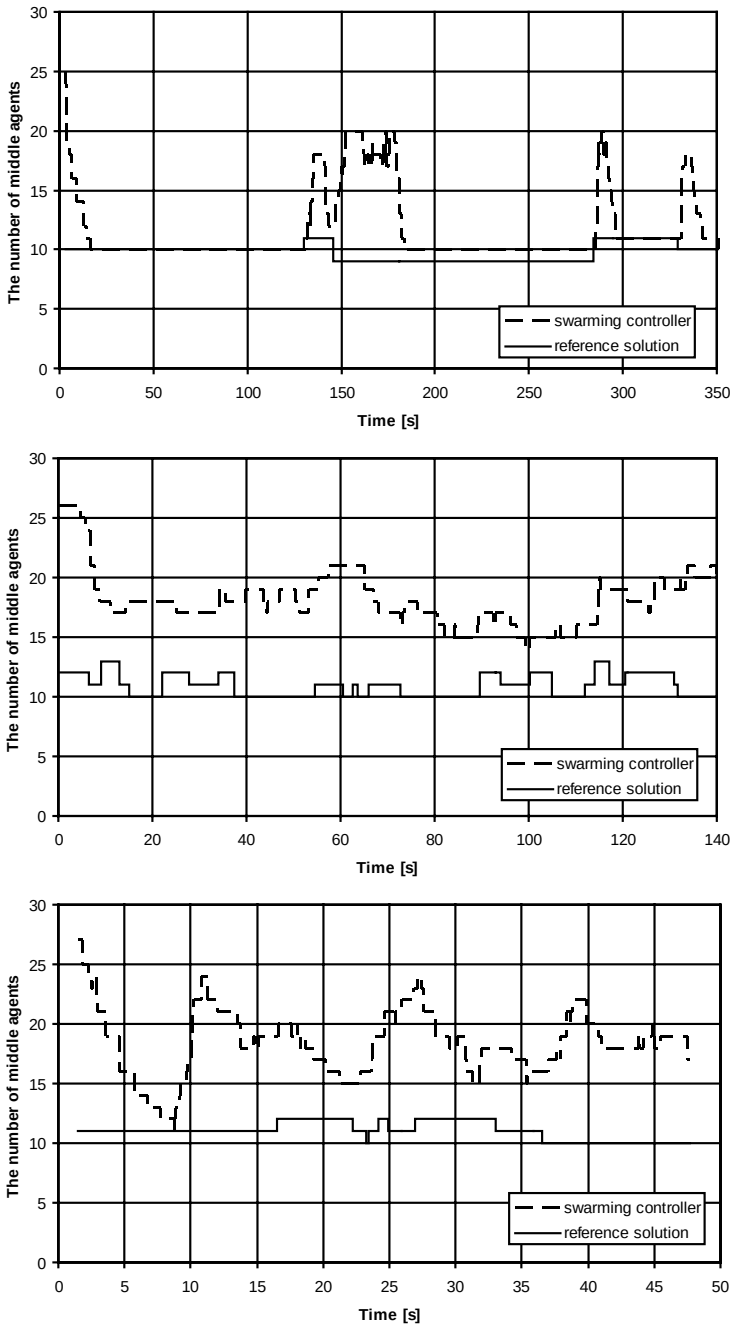

Fig. 7. The figure presents adaptation of middle agent network to the changing environment with different speed of changes: from infrequent changes (top chart) to the fast, frequent changes (bottom chart).

ficiency of the middle agent mechanism is crucial – it needs match the performance of relaying (with supplementary robustness), but to keep the amount of communication between agents low in the same time.

In the experiments, testing agents that implement a FIPA CNP are used. Each contract net request must involve at least one middle agent, even if a direct link between the sender and the receiver exists, and any advanced middle agent capabilities were not used to keep the results middle agent type-independent. Therefore, middle agents worked in a simple relay mode, not using any social knowledge. However, the use of social

knowledge typically further decreases the number of messages necessary [12].

When system is started, no middle agents exist. The first middle agent is created by the sender who wants middle agents to deliver its request to one or more targets. This first middle agent then propagates using the full flood fill as described in Section 3.2.

4.4. Information propagator adaptation

In the experiment, the *population manager* has been deactivated to keep one middle agent and one test agent on each of 24 nodes in the testing network. As a "worst" case scenario, full link accessibility case was set to obtain the slowest convergence of the number of messages towards the optimum, as many loops and alternative paths exist in the network. In the scenario, one of the testing agents periodically starts a testing CNP to all test agents. The results (Fig. 6) show the system wide number of transmitted messages per each CNP round. The results for two different values of the forgetting parameter are provided. This parameter should be zero in cases of non-changing topologies because all knowledge previously stored in information propagator can be reused and the system rapidly converges to the optimal number of 119 messages per CNP round. When the value of the forgetting parameter is increased, the number of messages per round can't converge to the optimum because a certain amount of the knowledge is being lost. The experiment shows that the number of messages is decreasing exponentially until a certain threshold is reached. After reaching the threshold, the number of messages per round remains more or less constant. The experiment also verified that in case of the full accessibility, any routing path contains exactly one middle agent.

4.5. Adaptation to the changing environment

In this experiment, the visibility range was set to minimize the link accessibility, but to maintain complete path accessibility. Testing agents and relay agents were set up as in the previous experiment, but the population manager was enabled this time, causing the number of relay agents to vary over time. The evolution of the number of middle agents were measured at three levels of inaccessibility dynamics. To determine the optimal number of stand-in agents in each moment, an efficient centralized algorithm has been implemented briefly presented in the next paragraph. In the testing domain, where the accessibility is distance-based, this algorithm behaves optimally.

4.5.1. Reference solution

This algorithms finds the shortest routing paths between all pairs of the agents and selects a subset of these paths that is supported by the minimal number of middle agents.

The communication environment is described as a non-oriented and non-valued graph $G = (V, E)$ where V is a set of nodes and E is a set of edges between link-accessible nodes.

Let set `solved` contain the nodes where the middle agents should be placed. $\texttt{distance}(c_i)$ is a number of edges in a shortest path p_i between the couple of nodes c_i. The algorithm processes all couples c_i with $\texttt{distance}(c_i) = 2$. Let $\texttt{fp}_{c_i} = \{p_i\}$ is a set of the shortest paths between a couple of nodes c_i. $\texttt{mid}(p_i)$ is the node in the middle of a path p_i, where $\texttt{length}(p_i) = 2$.

```
solved = ∅
nearCouples = ∅
C = set of all couples of nodes
for (∀c_i ∈ C) {
   if (distance(c_i) == 2)
     nearCouples = nearCouples ∪ c_i
}
while (nearCouples != ∅) {
   allConflicts = ∅
   for (∀ nn_i ∈ nearCouples) {
     if (∀ p_i ∈ fp_nn_i :  mid(p_i) ∉ solved) {
       conflicts = ∅
       for (∀ p_i ∈ fp_nn_i)
         conflicts = conflicts ∪ mid(p_i)
     }
     allConflicts = allConflicts ∪
      {conflicts}
   }
   addThisRelay = most frequented node in
     allConflicts
   for (∀ nn_i ∈ nearCouples) {
     if (mid(nn_i) == addThisRelay)
       remove nn_i from nearCouples
   }
   solved = solved ∪ addThisRelay
}
```

The algorithm returns `solved`, the minimum set of nodes where the middle agents must be placed to connect all couples c_i with $\texttt{distance}(c_i) = 2$. Using induction, it can be proved that the `solved` set supports also the shortest paths between any couple c_i in the system, regardless of their distance.

The algorithm takes a couple of nodes c_i whose $\texttt{distance}(c_i) = 2$ and determines all nodes that lies on the fastest paths between the couple. If there is no middle agent on either of these nodes they are placed into `conficts`. In `allConficts` are separately stored

`conflicts` generated by all the couples in the graph. The most frequently added node in `allConflicts` is added to `solved` set. After that the `allConflict` is set clear and the algorithm repeats until all two-edge paths are interconnected by a middle agent.

4.5.2. Results

At first (Fig. 7, top) the visibility ranges were slowly changed on the network with fixed nodes. In the second setup (Fig. 7, middle), one mobile node has been added into the network. The movement of the mobile node through the whole network introduces local accessibility changes. In the third setup (Fig. 7, bottom), two mobile nodes were moving faster through the community, causing more important disturbances in the network topology.

In the experiments, it is clearly demonstrated that the agents are able to organize themselves efficiently and to approach the optimal number as determined by the reference algorithm. However, after the steep initial decrease, the peaks that correspond to agent propagation in response to the topology changes can be observed. In the mobile scenarios, presented middle agent architecture (Section 3.1) has failed to match the reference solution perfectly, as the adaptation time is somewhat higher than the average change period, but the results remain comparable and the robustness and distribution of the algorithm provides enough of the incentive for its application.

Acknowledgement

The research described in this paper has been supported in parts by the EOARD/AFRL projects number FA8655-04-1-3044, FA-8655-02-M-4057.

5. Conclusions

This contribution defines the concept of inaccessibility in the multi-agent systems, addresses existing or novel solutions to the problem and presents their comparison in various inaccessibility situations.

The theoretical analysis from Section 2.2, as well as experiments in Sections 4.1 and 4.2 prove that the random graph theory can be successfully used to describe the inaccessibility in a real-world multi-agent system. Inaccessibility solutions presented in Section 3 are evaluated in three inaccessibility states defined in Table 1. Therefore, the results allow to select the optimal inaccessibility solution for a specific domain, once this domain is described using the metrics presented.

To address the optimality and scalability of inaccessibility solutions in connected systems, domain-independent distributed algorithm for the optimization of the middle agent community in an inaccessible environment is described and tested. In the experiments, it is shown that the presented solution is efficient in communication, robust with respect to important environment changes and ensures complete system connectivity in the environments with high path accessibility and low link accessibility. In contrast to generic routing algorithm, the system can be integrated with any type of middle agent, making it useful in all types of inaccessible environments – from the completely disconnected ones to the environments with occasional dropouts only.

Future research work will be focused on the algorithm tuning and automatic environmental adaptation; many internal parameters are currently set in an arbitrary manner and their learning from the environment can further increase the usability and efficiency of the solution. To validate the algorithm in the real environment, it is planned to integrate it with the distributed multi-agent systems in the information-intensive mobile robotics applications [19].

References

[1] IEEE802.11 Specification. http://grouper.ieee.org/groups/802/11/index.html.

[2] R. Albert and A. L. Barabási, Statistical mechanics of complex networks, *Rev. Mod. Phys.* **74** (2002), 47–97.

[3] G.L. Alexanderson, *The random walks of George Pólya*, The Mathematical Association of America, 2000.

[4] B. Bollobas, *Random Graphs*, 2nd edition, Cambridge University Press, 2001.

[5] W. Cao, C.-G. Bian and G. Hartvigsen, Achieving efficient cooperation in a multi-agent system: The twin-base modeling, in: *Cooperative Information Agents*, P. Kandzia and M. Klusch, eds, number 1202 in LNAI, Springer-Verlag, Heidelberg, 1997, pp. 210–221.

[6] E.P.E. Leontiadis and V.V. Dimakopoulos, Cache updates in a peer-to-peer network of mobile agents, In *Proceedings of the Fourth IEEE International Conference on Peer-to-Peer Computing*, IEEE Computer Society, 2004, 10–17.

[7] D. Estrin, D. Culler, K. Pister and G. Sukhatme, Connecting the physical world with pervasive networks, *IEEE Pervasive Computing* **1**(1) (2002), 59–69.

[8] FS1037C. Federal standard 1037c: Telecommunications: Glossary of telecommunication terms. http://www.its.bldrdoc.gov/fs-1037/, 1996.

[9] S. Kumar, P.R. Cohen and H.J. Levesque, The adaptive agent architecture: Achieving fault-tolerance usingpersistent broker teams Technical Report CSE-99-016-CHCC, Oregon Graduate Institute, 1999.

[10] M. Pěchouček, V. Mařik and O. Štěpánková, Role of acquaintance models in agent-based production planning systems, in: *Cooperative Infromation Agents IV – LNAI No. 1860*, M. Klusch and L. Kerschberg, eds, Heidelberg, Springer Verlag, July 2000, pp. 179–190.

[11] M. Pěchouček, V. Mařík and J. Bárta, A knowledge-based approach to coalition formation, *IEEE Intelligent Systems* **17**(3) (2002), 17–25.

[12] M. Pěchouček, V. Mařík and O. Štěpánková, Towards reducing communication traffic in multi-agent systems, *Journal of Applied Systems* **2**(1) (2001), 152–174, ISSN 1466–7738.

[13] M. Rehák, M. Pěchouček, J. Tožička and D. Šišlák, Using standin agents in partially accessible multi-agent environment, in: *Proceedings of Engineering Societies in the Agents World V, Toulouse*, October 2004, number 3451 in LNAI, pp. 277–291. Springer-Verlag, Heidelberg, 2005.

[14] M. Sierhuis, J. Bradshaw, A. Acquisiti, R. van Hoof, R. Jeffers and A. Uszok, Human-agent teamworks and adjustable autonomy in practice, In *Proceedings of the 7th International Symposium on Artificial Intelligence, Robotics and Automation in Space: i-SAIRAS – NARA*, Japan, May 2003.

[15] O. Simonin and J. Ferber, Modeling self satisfaction and altruism to handle action selection and reactive cooperation, In *proceedings Supplement SAB 2000, The Sixth International Conference on the Simulation of Adaptative Behavior, FROM ANIMALS TO ANIMATS 6* (*Paris, France*), 2000, 314–323.

[16] D. Šišlák, M. Rollo and M. Pěchouček, A-globe: Agent platform with inaccessibility and mobility support, in: *Cooperative Information Agents VIII*, M. Klusch, S. Ossowski, V. Kashyap and R. Unland, eds, number 3191 in LNAI. Springer-Verlag, Heidelberg, September 2004.

[17] W. Stallings, *Data and computer communications*, (5th ed.), Prentice-Hall, Inc., 1997.

[18] D. Stauffer, *Introduction to percolation theory*, Taylor and Francis, London and Philadelphia, 1985.

[19] N. Suri, M.M. Carvalho, J.M. Bradshaw, M.R. Breedy, T.B. Cowin, P.T. Groth, R. Saavedra and A. Uszok, *Enforcement of communications policies in software agent systems through mobile code*, In POLICY, 2003, 247–250.

[20] K. Sycara, J. Lu, M. Klusch and S. Widoff, Dynamic service matchmaking among agents in open information environments, *ACM SIGMOID Record* **28**(1) (1999), 211–246.

[21] K. Sycara, Multi-agent infrastructure, agent discovery, middle agents for web services and interoperation. In *Mutli-agents systems and applications*, Springer-Verlag New York, Inc., 2001, 17–49.

[22] V. Thomas, C. Bourjot, V. Chevrier and D. Desor, Hamelin: A model for collective adaptation based on internal stimuli, in: *From animal to animats 8 – Eighth International Conference on the Simulation of Adaptive Behaviour 2004 – SAB'04*, Stefan Schaal, Auke Ijspeert, Aude Billard, Sethu Vijayakumar, John Hallam and Jean-Arcady Meyer, eds, Los Angeles, USA, Jul 2004, pp. 425–434.

[23] A. Woo, T. Tong and D. Culler, Taming the underlying challenges of reliable multihop routing in sensor networks. In *Proceedings of the first international conference on Embedded networked sensor systems*, ACM Press, 2003, 14–27.

[24] M. Wooldridge, *Reasoning about Rational Agents*, Intelligent robotics and autonomous agents. The MIT Press, 2000.

Authors' Bios

Michal Pěchouček works as Assistant Professor in Artificial Intelligence at the Department of Cybernetics, CTU FEE. He graduated in Technical Cybernetics from FEE-CTU, got his M.Sc. degree in IT: Knowledge Based Systems from University of Edinburgh and completed his Ph.D. in Artificial intelligence and biocybernetics at CTU in Prague. He is the Head of the Agent Technology Group at the Gerstner Laboratory for Intelligent Decision Making and Control. His research focuses on problems related to multi-agent systems, production planning, knowledge analysis, machine learning. Michal Pěchouček participated in and led several international projects (e.g. ExPlanTech: Exploitation of Agent-based Production Planning using the ProPlanT Technology, US Air Force lab. contract focusing on the Coalition Formation in Operations Other than War. etc.) He is an author or co-author of about 40 publications in proceedings of international conferences and journal papers. In addition, he was the co-chair of the International Workshop on Industrial Applications of Holonic and Multi-Agent Systems (HOLOMAS 2000–2002) organized under the DEXA conferences and organising committee member of KSCO 2002 2nd International Conference on Knowledge Systems for Coalition Operations.

David Šišlák graduated from Faculty of Electrical Engineering Czech Technical University in Technical Cybernetics in 2003. Currently he is a PhD student in the Department of Cybernetics Agent Technology Group, and a researcher in the Gerstner Laboratory. His research interests are in multi-agent systems focusing on efficient communication and knowledge maintenance in inaccessible multi-agent environment. He obtained the international Cooperative Information Agents workshop (CIA) system innovation award in the year 2004 to honor an excellent research and development of an innovative and demonstrated system of intelligent information agents for agent platform ***A*-globe** with inaccessibility and mobility support.

Martin Rehák holds engineering degree from Ecole Centrale Paris. He is currently researcher at Agent Technology Group of the Gerstner Laboratory and in the same time pursues his PhD studies at the Department of Cybernetics of the Czech Technical University. Prior to his current position, he was member of the Mobile Communication Operations team of Schlumberger Smartcards (now Axalto), where he was working on definition, design and integration of novel location-based and other value added services for major Euro-

pean and African operators. Before obtaining his degree, he was a freelance contractor to diverse Czech and French companies, including General Electric Medical Systems. His current research interests are ubiquitous multi-agent systems, security, inaccessibility and distributed planning.

Jan Tožička holds Doctorate in Natural Sciences degree in Theoretical Computer Science from Faculty of Mathematics and Physics at Charles University in Prague where he also received his Master of Science degree in the same branch. He is currently researcher at Agent Technology Group of the Gerstner Laboratory and in the same time finishes his PhD studies at the Department of Cybernetics of the Czech Technical University. His research focuses on knowledge analysis, meta-reasoning, machine learning and formal methods in multi-agent systems.

Petr Benda is a Phd student of Artificial intelligence and biocybernetics at Faculty of Electrical Engineering Czech Technical University. He graduated in Biomedical engineering in 2004. He has been a member of Agent technology group since 2002, where he started the RoboCup Rescue project. His research interests are biological systems, synergic effects in multi-agent systems and general artificial intelligence.