Reducing Communication Cost via Overhearing

Adrian Perreau de Pinninck[1], Gery Gutnik[2], and Gal Kaminka[2]

[1] IIIA – Artificial Intelligence Research Institute
CSIC – Spanish National Research Council
Bellaterra (Barcelona), Catalonia, Spain
adrianp@iiia.csic.es
[2] The MAVERICK Group
Computer Science Department Bar-Ilan University, Israel
{gutnikg,galk}@cs.biu.ac.il

Abstract. Coordination of task execution, in open distributed multi-agent systems, is often accomplished through communications between agents. Thus, most tasks carry some cost associated with those communications. The work in this article proposes to use overhearing for reducing some of this cost, and therefore reducing the overall cost of task execution. Since in open distributed systems, and in particular in large-scale settings, it is rare for two agents to communicate directly, their communications (as in real-world networks) are routed to their destination through other agents. We allow those intermediate agents to overhear and monitor passing through communications. In doing so, the intermediate agents detect some of the errors generated by communicating agents before they reach their final destination. Filtering those erroneous messages in advance reduces the cost associated with routing them through the system. This work first formalizes this problem, and then proposes algorithms for finding an effective set of filtering agents. Unfortunately, finding an optimal solution for this problem turned out to be intractable. Thus, an efficient heuristic solution is proposed. An empirical simulation of these algorithms shows that the heuristic algorithm achieves similar performance to that of the optimal algorithm, while maintaining efficient run-time complexity.

1 Introduction

Open distributed multi-agent systems (MAS) often rely on inter-agent communications to coordinate the execution of complex tasks by multiple agents. Accordingly, most tasks in such settings, carry some cost associated with communication between agents. Minimizing this cost has the potential to reduce the overall cost of task execution.

This work proposes to use overhearing for tackling this problem. Overhearing is a generic approach that allows monitoring task performance and inference about agents executing them based on the routine communications performed by monitored agents [1, 12, 14]. A detailed description of previous investigations on overhearing can be found in Section 2.

Addressing inter-agent communications in open distributed MAS, it is highly improbable for two agents to have a direct communication channel. In particular, this is true for large-scale MAS. Instead, it is more likely, as in real-world networks, that communication between two agents will be done through a chain of intermediate agents. Allowing intermediate agents to overhear and monitor the communications they route, presents the opportunity to detect in advance some of the erroneous messages generated by communicating agents. Filtering those messages before they reach their final destination, reduces the unnecessary cost of routing them through the network.

There already exist regulated MAS in which exchanged messages are being overheard. These regulated MAS are called Electronic Institutions (EI) which, similar to real-world institutions, define strict protocols agents must follow to achieve their goals [2, 8, 16] (see also Section 2 for more details). In EI systems, it is of utmost importance that only valid messages are delivered. Thus, the main purpose of monitoring in such settings is to ensure that the exchanged messages conform to the set of defined protocols and to filter invalid messages. Our work focuses on how this type of filtering can be done effectively.

We address the problem in two stages. First, we introduce a formal model of the problem (see Section 3). Building on a formal definition of agent communication networks, we define the cost of inter-agent communication with and without filtering. Then, based on the proposed model, we present algorithms for finding an effective set of filtering agents, given a network and a set of communicating agents (see Section 4).

Unfortunately, we find that the algorithm for finding the optimal set of filtering agents is exponential in the number of intermediate agents. Thus, using this algorithm in large-scale MAS, where the number of intermediate agents can be relatively high, is infeasible. Therefore, we present a heuristic algorithm. This algorithm, though not always able to determine an optimal set of filtering agents, has a polynomial run-time complexity.

Finally in Section 5, we evaluate the effectiveness of the proposed heuristic solution, measuring the level of approximation it provides compared to the optimal solution. To achieve this, both algorithms have been evaluated empirically in different communication network settings, changing the numbers of communicating agents and potential error rates. The conducted experiments were performed both on randomly-generated communication networks, as well as on networks with real-world characteristics. The results of those experiments show that no significant difference can be found between the effectiveness of the heuristic and the optimal algorithms.

2 Related Work

Overhearing is fast gaining attention as a general method for monitoring open distributed MAS. Previous works on overhearing proposed a variety of potential applications. Novick and Ward [14] introduced an early use of overhearing where pilots maintained their situational awareness by listening to conversations of other pilots with the controller. Similarly, Legras [13] and Rossi and Busetta [17, 18] applied overhearing for maintaining organizational knowledge of roles performed by other agents.

Cabri et al. [4] extended the latter approach by not only allowing the overhearer to monitor the roles performed by other agents, but also by allowing it to provide suggestions as to other roles the agents might be interested in. Some other works also addressed assistance that might be provided based on overheard communications. Aiello et al. [1] and Busetta et al. [3] discussed providing expert assistance by an overhearer, while Fan and Yen [9] have focused on anticipating and satisfying the information needs of overheard agents.

Other works investigated the use of overhearing for monitoring agent's progress in task execution. Kaminka et al. [12] applied overhearing for plan recognition, while Poutakidis et al. [15] and Rossi and Busetta [17, 18] used overhearing to detect inconsistencies in the performance of monitored tasks. The information gathered during the monitoring process can later be used for issuing alerts and notifications [5, 17, 18].

All other works on overhearing assume that messages are always delivered. Our work takes a different direction in applying overhearing. We propose to use overhearing for reducing the communication cost associated with the execution of tasks. By allowing agents to monitor and overhear those communications they have routed, those agents can detect erroneous messages and filter them before they reach their final destination.

This type of monitoring already exists in regulated multi-agent settings called Electronic Institutions (EI) [2, 8, 16]. EI systems define communication protocols that agents must follow in order to achieve certain goals. For example, EIs have been defined for auction houses, where agents must follow explicit protocols to sell or buy goods. In EI systems, it is of highest importance that invalid messages do not reach their final destination. To ensure this, each agent joining an EI is assigned a governor agent whose task is to verify that the messages the agent sends conform to the communication protocols defined in the EI. Furthermore, there are specific inter-governor protocols in place, routing messages and propagating states of monitored protocols amongst governors, in order to verify that exchanged messages are indeed valid.

This work differs from traditional EIs in two characteristics. Firstly, we do not assume the existence of special governor agents in charge of the filtering. Instead, we allow agents within the MAS to fulfill this role by monitoring their peers. Secondly, we do not allow (for now) communication between filtering agents. Thus, we can not ensure that all invalid messages can be filtered. However, our focus is the reduction of the communication cost associated with task execution, rather than maintaining communication integrity as done in EIs.

3 Problem Formalization

This section presents a formal definition of the filtering problem. Section 3.1 introduces a formalization of communication networks, Section 3.2 presents the concept of filtering, and finally, Section 3.3 defines the cost of inter-agent communications with and without filtering.

3.1 Communication Networks

A communication network can be modeled as an undirected connected graph without self-loops. We denote this graph as $\langle AG, CH \rangle$. The vertices AG in the communication network represent MAS agents, while the edges CH define whether a direct communication channel exists between two agents.

Furthermore, we define additional attributes for both vertices and edges. In the process of inter-agent communications, agents generate errors (i.e., messages that do not follow any of the valid communication protocols). For example, at the end of the querying communication protocol, instead of the expected inform message, another query message is received. We define an error rate for each agent: the ratio of invalid messages to total messages sent by the agent. This rate is defined as $err : AG \rightarrow [0, 1]$.

Each edge has an associated cost; the cost of sending a message from one agent to another through the channel. We refer to it as bandwidth cost and denote it as $bw : AG \times AG \rightarrow \mathbb{R}$. The differences in bandwidth costs represent the network quality of the communication channel. However, most agents do not have a direct communication channel between them. Instead, they communicate through a sequence of intermediate agents which forms a path in the network. Multiple paths may connect two agents. However, it is reasonable to assume that communications will be directed through the path with the lowest cost. The cost of communicating through a path can be defined as the sum of the costs of its edges. Thus, for a given shortest-weighted path $[a_1, a_2, ..., a_n]$, the bandwidth cost can be calculated as $bw(a_1, a_2)+...+bw(a_{n-1}, a_n)$. The shorthand notation $bw(a_1, a_n)$ will be used for the rest of this paper. We define the path with the minimal bandwidth cost between two agents as $path : AG \times AG \rightarrow AG^*$.

Through these lowest-cost paths, agents communicate with each other by sending messages. A message is defined as a $\langle s, r, d, id_c \rangle$ tuple where s and r are the identifiers of sender and receiver agents, respectively, d is the data transmitted, and id_c is the conversation identifier. Each agent, receiving a message, checks whether it is the target receiver. If not, it acts as an intermediate agent and forwards the message further to its final destination. Intermediate agents hold routing tables (according to $path$s) which allow to do this forwarding efficiently. Such routing tables can be calculated using any shortest-weighted path algorithm, such as Dijkstra's.

Any message received by the target agent is acknowledged by sending a feedback message back to the sender through the same path. This feedback informs the sender agent whether the message was considered valid or not by the receiver. The target receiver agent has all the previous messages of a certain conversation, and thus can verify whether the received message is valid or not. It can also handle some amount of losses as done in [9, 10].

3.2 Filtering

The approach proposed in this paper allows intermediate agents to overhear and monitor messages they route. Intermediate agents may verify whether overheard messages are valid or invalid with respect to the communication protocol being used. Thus, we assume that agents share a communication protocol library containing the protocols that can be used in communications between agents. Otherwise, any overheard message could be considered valid with respect to some unknown protocol. New agents joining the system can learn the protocol library through interaction [11].

Intermediate agents may perform a verification on overheard messages. Algorithms for the verification process have previously been addressed in [9, 10, 15]. We assume filtering agents to be truthful, filtering only invalid messages in monitored communications. When an intermediate agent detects an invalid message, there is no need to forward it further through the network. Thus, intermediate agents may filter invalid messages and send back appropriate feedback. On the other hand, if the intermediate agent can not determine a message to be invalid, it is simply forwarded on as described in Section 3.1.

Filtering invalid messages reduces the bandwidth cost of routing them through the network. However, the filtering process comes with a cost of its own. The cost the intermediate agent incurs for verifying the overheard message. We denote this cost as $cpu : AG \rightarrow \mathbb{R}$, an additional attribute to those shown in Section 3.1, associated with every agent. The differences in cpu costs represent the differences in computation power of the computers on which different agents run.

In the path between two communicating agents, any intermediate agent may act as a filtering agent. Choosing the filtering agents for a certain path depends on three parameters: The accumulated bandwidth cost between the sender and the filtering agent, the cpu cost of the filtering agent, and the probability that it can filter the message. We have already discussed the first two. Thus, the remaining part of this section will focus on the latter.

Intermediate agents are only aware of messages they route. In this paper, we do not allow filtering agents to communicate with each other leaving this for future research. Thus, each intermediate agent only has partial knowledge of the monitored conversation. Accordingly, it verifies the overheard messages using techniques that rely on availability of only partial knowledge [9, 10].

The probability of detecting invalid messages depends on the communication protocol being followed, and the network topology. It is out of the scope of this paper to study this in detail. Nonetheless, we do consider that an agent will only be able to filter messages that another agent could not filter, if and only if it has overheard some communication messages that the other agent did not. Furthermore, if an agent has complete knowledge of a conversation, it will always be able to distinguish valid messages from invalid ones.

3.3 Communication Costs

A conversation can be viewed as a sequence of messages sent between agents. The cost of a conversation is the accumulated cost of its messages (both valid and invalid according to the protocol). The formal definition of this cost is presented in Equation 1. For each sender and receiver pair, denoted as s and r, respectively, in the set of communicating agents, denoted by C, we calculate the communication costs between those two agents. The total conversation cost (i.e. $cost_c$) is the sum of the communication cost between each sender-receiver pair. The function num_m defines the number of messages sent from one agent to another in the conversation. Thus, $num_m(s, r)$ represents the number of messages sent through the path from agent s to r. For each of those messages, the $cost_m$ function, described below, represents the average cost of communicating a message between s and r given a set of filtering agents defined by $FM(s, r)$. FM can be considered as a mapping function that provides the set of filtering agents for each pair of sender and receiver agents in the conversation. Thus, the total communication cost between s and r is the number of messages s sends to r times the average cost per message between s and r, assuming that filtering is performed by the set of agents denoted as $FM(s, r)$.

$$cost_c(C, num_m, FM) = \sum_{s \in C} \sum_{r \in C} num_m(s, r) cost_m(s, r, FM(s, r)) \tag{1}$$

Let us now consider the cost of communicating a message between any two agents. Equation 2 provides the formal definition of this cost. The cost of sending a message from s to r, given a set of filtering agents F, is the probability that the message is valid times the cost of sending a valid message (i.e. $cost_{mv}$) plus the probability that the message is invalid times the average cost of sending an invalid message (i.e. $cost_{miv}$).

$$cost_m(s, r, F) = (1 - err(s)) cost_{mv}(s, r, F) + err(s) cost_{miv}(s, r, F) \tag{2}$$

In case the message is valid, it will not be filtered. Thus, the communication cost is equal to the bandwidth cost of sending it all the way from the sender to the receiver and back (due to the feedback), plus the verification cost of each of the filtering agents along the path. The verification in this case will only verify that indeed the message is valid. This cost is formally defined in Equation 3.

$$cost_{mv}(s, r, F) = 2bw(s, r) + \sum_{f \in F} cpu(f) \tag{3}$$

On the other hand, in case the message is invalid, its associated cost is determined based on whether it is filtered or not. Along the path between sender and receiver, there might be a number of filtering agents. Initially, the first filtering agent attempts to detect whether the message is invalid. If it succeeds, it filters the invalid message and sends a feedback to the sender. If not, the message is forwarded on, and it is now the turn of the next filtering agent to do the same. The process continues until the last filtering agent forwards the message to the receiver.

Accordingly, a filtering agent will only attempt to filter the invalid message if the other filtering agents before it were not successful in doing so. Thus, we define the $cost_{miv}(s, r, F)$ function, representing the cost of an invalid message from s to r given a set of filtering agents F, using an overloaded function $cost_{miv}(s, r, F, F')$ where a fourth parameter is added to represent filtering agents that have already attempted (and failed) to filter the invalid message. The first time this function is called, the F' parameter is set to $\emptyset$ representing an empty set of filtering agents.

The $cost_{miv}(s, r, F, F')$ function is defined recursively as shown in Equations 4 and 5. When the set of filtering agents is empty, the communication cost per message from agent s to agent r equals twice the bandwidth cost between those agents (message and feedback). This stop condition is formalized in Equation 4.

$$cost_{miv}(s, r, \emptyset, F') = 2bw(s, r) \tag{4}$$

Finally, Equation 5 formalizes the recursive call. This equation is composed of two parts. The first part entails the cost associated with the current filtering agent, denoted as

f. The current filtering agent f is the first within the list of filtering agents represented by a concatenation of f and the remaining set of filtering agents denoted as F. The communication cost associated with f is equal to twice the bandwidth between the sender and the filtering agent, i.e. $2bw(s, f)$, plus the verification cost, i.e. $cpu(f)$. In case the filtering agent f detects that the message is invalid, this is also the final cost, since the message is filtered and a feedback is sent back to the sender. Otherwise, the message is forwarded through the intended path. This case is handled by the second part of the equation. The probability function P represents the conditional probability of agent f not being able to filter the invalid message given that all previous filtering agents, denoted by F', were not being able to do so either. The $fil_f(m)$ function returns false if filtering agent f cannot detect that message m is invalid. In such case, the invalid message is forwarded on to be handled by the remaining filtering agents. Thus, the communication cost is calculated through a recursive call to $cost_{miv}$ with f as the sender, r as the target receiver, a concatenation of F' and f as the set of previous filtering agents and F as the remaining set of filtering agents.

$$\begin{aligned} cost_{miv}(s, r, f \cdot F, F') &= 2bw(s, f) + cpu(f) \\ &+P[\neg fil_f(m) \mid \bigwedge_{f' \in F'} \neg fil_{f'}(m)] cost_{miv}(f, r, F, F' \cdot f) \end{aligned} \tag{5}$$

Using the equations presented throughout this section, we can calculate the cost of a conversation between agents given a the set of filtering agents in each path. The cost of communication with no filtering can be considered as a special case where the set of filtering agents is empty.

4 Finding Filtering Agents

This section addresses the problem of determining an effective set of filtering agents such that it minimizes the associated communication cost. Section 4.1 presents an an algorithm that finds an optimal solution which has an exponential run-time complexity, while Section 4.2 proposes an alternative heuristic algorithm. Though the latter may provide a non-optimal solution, it has an efficient polynomial run-time complexity.

4.1 An Optimal Solution

This section presents algorithms for finding the optimal set of filtering agents. First, we show the naive algorithm that finds the optimal solution, and then explain how it can be optimized.

The naive version is described in Algorithm 1. Given a set of communicating agents CA, it iterates through each path between sender and receiver agents, denoted as s and r respectively (lines 1–2). For each pair of communicating agents s and r, it determines the optimal set of filtering agents for the path between them (lines 3–9), and accumulates it in the final map of filtering agent sets, denoted as FM (lines 10–11). For each path, it calculates the set of intermediate agents for this path (line 3). This set is denoted as IA. Then, it considers each subset of IA to be the potential set of filtering agents

Algorithm 1 NaiveOptimalAlgorithm(communicating agents CA)

```
1: for all $s \in CA$ do
2:   for all $r \in CA$ do
3:     $IA \leftarrow$ intermediate agents in $path(s, r)$
4:     $set_{best} \leftarrow \emptyset$
5:     $cost_{min} \leftarrow \infty$
6:     for all $set \subseteq IA$ do
7:       if $cost_m(s, r, set) < cost_{min}$ then
8:         $cost_{min} \leftarrow cost_m(s, r, set)$
9:         $set_{best} \leftarrow set$
10:    $FM(s, r) \leftarrow set_{best}$
11: return $FM$
```

(line 6). The optimal filtering set is the subset that minimizes the communication cost (i.e. $cost_m$) between s and r. It is chosen in lines 7–9.

Considering the complexity of this naive algorithm, we will denote $|CA| = n$ and $|IA| = m$. The iteration through all pairs of communicating agents has the complexity of $O(n^2)$. Finding the IA set, in line 3, can be done efficiently using routing tables of each agent in communication network (see Section 3.1). Using those routing tables, each agent can determine in $O(1)$ to whom it should forward the message in case it is not the final target. Thus, determining IA can be done with run-time complexity of $O(m)$. Iterating through all subsets of IA has the complexity of $O(2^m)$. Finally in each iteration the message cost is calculated, which can be done in $O(m)$. Thus, the total complexity of the naive optimal algorithm is $O(n^2(m + m2^m))$, which is the same as $O(n^2 2^m)$.

A small optimization can be done with respect to the proposed algorithm. It can be proved that if the communication cost is lower with no filtering agents than having a certain intermediate agent as a filtering agent, then this intermediate agent will never be part of the optimal filtering set. Due to lack of space, we have omitted this proof from the paper. Based on this insight, we can better define the set of intermediate agents to consider (line 3 of Algorithm 1) removing those *useless* agents from the IA set. This will reduce the number of subsets which need to be considered, and thus improve the run-time of the naive algorithm. However, in the worst-case, the complexity of the algorithm, even with this optimization, remains $O(n^2 2^m)$.

4.2 A Heuristic Solution

The analysis provided in the previous section shows that the optimal algorithm cannot be used in large-scale settings where the number of intermediate agents tends to be relatively high. As an alternative, we propose a heuristic algorithm with polynomial run-time complexity which can be efficiently applied in large-scale MAS. Furthermore in Section 5 it is shown that no significant difference can be appreciated between the effectiveness of the heuristic and the optimal algorithm.

The proposed heuristic is formalized by Algorithm 2. Its basic idea is similar to Algorithm 1 presented in Section 4.1. However, instead of iterating through all possible subsets of IA, as done in Algorithm 1, only allows one filtering agent per path. Thus, in line 6, it considers each intermediate agent ia in IA to form a potential set of filtering agents containing only a single agent—itself, denoted by $\{ia\}$. The best filtering set chosen (lines 7–9) will be the one that contains the single filtering agent that minimizes the most the communication cost in the corresponding path. However, sometimes it is more effective not to filter at all. Thus, the initial values for set_{best} and $cost_{min}$ parameters are set according to an empty set of filtering agents (lines 4–5).

Algorithm 2 HeuristicAlgorithm(communicating agents CA)

```
1:  for all $s \in CA$ do
2:    for all $r \in CA$ do
3:      $IA \leftarrow$ intermediate agents in $path(s, r)$
4:      $set_{best} \leftarrow \emptyset$
5:      $cost_{min} \leftarrow cost_m(s, r, \emptyset)$
6:      for all $ia \in IA$ do
7:        if $cost_m(s, r, \{ia\}) < cost_{min}$ then
8:          $cost_{min} \leftarrow cost_m(s, r, \{ia\})$
9:          $set_{best} \leftarrow \{ia\}$
10:     $F(s, r) \leftarrow set_{best}$
11: return $F$
```

Analyzing the complexity of the heuristic algorithm, it can be seen that the $O(2^m)$ part of the optimal algorithm changes to $O(m)$. Thus, the heuristic solution, for a single path, has linear complexity in the number of intermediate agents, and thus can be applied in large-scale settings.

5 Experiments

We empirically evaluate the effectiveness of the proposed heuristic algorithm, comparing it to the optimal algorithm. In Section 5.1, we describe the experimental settings, while Section 5.2 presents the results of the performed experiments.

5.1 Experimental Settings

The proposed algorithms have been evaluated based on communication networks comprised of 1000 agents. Below, we explain in details the construction of these networks, both in terms of their structure and in terms of chosen parameters (e.g., cpu and bandwidth costs, agent error rates, etc).

Addressing the structure of communication network, we considered both randomly-generated networks and networks with real-world characteristics. All networks have been constructed so that the average node degree is 10. The random communication networks have been generated according to the Erdos-Renyi "binomial model" [7] with a 1% probability of direct connection between any two agents. As for communication networks with real-world characteristics, we have followed the characteristics of Gnutella ultra-peer network. The Gnutella network has been found to have both small-world and scale-free properties [19]. Thus, both types of networks have been simulated. The simulated small-world networks follow the Watts model [20] with a ring lattice of degree 10 and a 10% rewiring probability, while the simulation of scale-free networks is done according to the Eppstein-Wang model [6] with an average node degree of 10. Those values represent the necessary adjustments made on the properties of large-scale Gnutella networks to the simulated communication networks of size of 1000. Throughout our experiments, we have witnessed no differences in results between those structures. Thus, the results presented average the simulations done with each of the structures.

The other network have been chosen to represent a broad spectrum of possibilities. For each parameter we choose a value close to each boundary and one in the middle. The average error rate was set to either 0.1, 0.5 or 0.9 simulating low, medium and high error rates respectively. Same principle has been applied to bandwidth cost to cpu cost ratio. Thus, three possible ratios have been defined: low (1/10), medium (1/1) and high (10/1).

The assignment of values has been done in one of the following manners. In *equal cost networks* all vertexes and edges have the same values. *Two-type networks* consist of two different types of agents: regular agents and server agents. The latter are more powerful agents that have lower cpu and bandwidth costs reflecting faster cpus and better network connections. In our simulation, 10% of the agents in two-type networks are considered to be server agents which have smaller costs compared to regular agents: 1/3 cpu cost, 1/3 bandwidth cost between server and regular agent and 1/9 bandwidth cost between two server agents. These networks describe a more realistic scenario. Due to the differences in costs between regular and server agents, agents tend to communicate through the latter. Thus, low-cost paths will be longer and contain more intermediate agents. We have also considered a third type of assignment where costs are normally distributed. However, since those results were similar to those of equal cost networks, and due to lack of space, we do not present them in this paper.

Finally, given a communication network with its parameters, we randomly choose a set of agents to communicate. The set of communicating agents has been set to sizes of 3, 15, 30 and 100 simulating small, medium, large and very-large sets of communicating agents, respectively. Given the set of chosen communicating agents, we calculate its associated communication cost. In order to calculate this, we assume that messages are distributed uniformly between communicating agents, and contain the same informative value. Based on this assumption, the filtering probability becomes a function of the percentage of paths the filtering agent mediates. We will use the identity function. Thus, the probability of filtering is equal to the ratio of mediated paths out of the total number of pathsThen, using algorithms described in Section 4, we find optimal and heuristic sets of filtering agents, and calculate communication cost and execution-time for both solutions.

5.2 Results

However, prior to addressing the effectiveness of the two algorithms, we considered whether the presented approach of filtering invalid messages is overall cost-effective.

Table 1 summarizes the percentage of improvement achieved by reducing the communication cost using an optimal set of filtering agents. We evaluated the attained improvement under different configurations of error rates (rows) and bandwidth to cpu costs ratios (columns) as explained in Section 5.1. The improvement achieved for a certain simulation can be calculated as $(1 - of/nf) * 100$, where of is communication cost using optimal set of filtering agents and nf is communication cost without any filtering. Each data cell in Table 1 represents the average values of 360 independent simulations.

E \ R	Low	Med	High
Low	0.0	0.0	0.0
Med	0.29	13.54	34.49
High	4.26	28.60	52.78

Table 1. Percentage of achievable cost reduction

Table 1 shows that for low error rates and low bandwidth to cpu ratios no significant improvement can be achieved. However, filtering becomes highly cost-effective for high error rates or in cases where the bandwidth costs are much higher than the cpu costs.

When filtering is not cost effective (low error rates and bandwidth to cpu ratios), the optimal and heuristic solutions will be the same: no filtering. The figures shown below only use data from simulations were filtering is cost effective in order to show the difference in cost reduction between the two. Both algorithms were evaluated for equal cost networks (Figures 1 and 2) and for two type networks (Figures 3 and 4). The algorithms were compared in two dimensions: communication cost (Figures 1 and 3) and run-time (Figures 2 and 4). In the case of optimal solution, the run-time was calculated for both the naive algorithm and its optimization without "useless" agents (see Section 4.1).

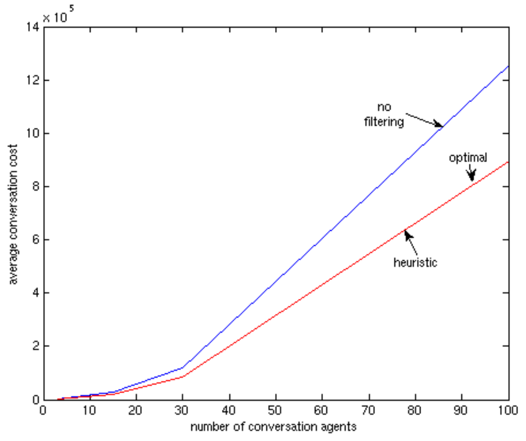

Fig. 1. Communication costs in equal cost networks

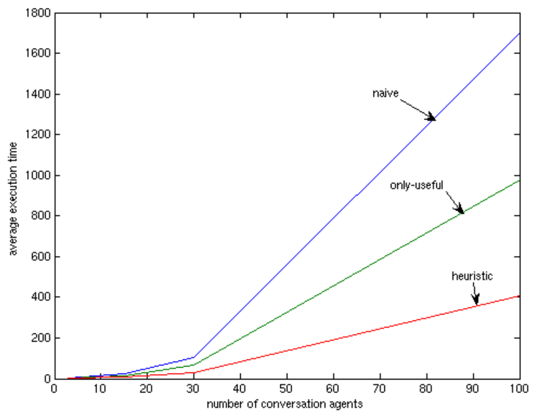

Fig. 2. Execution time in equal cost networks

In all figures, the x-axis always represents the number of communicating agents, while the meaning of the y-axis differs between graphs. In the figures comparing communication costs, the y-axis represents the conversation cost. In contrast, in figures measuring run-time, it represents the time taken to find the solution (*i.e.*, the set of filtering agents) measured in milliseconds. The data points, in the graphs, correspond to the average value of 180 independent experiments, with different parameter values.

Addressing the results in equal cost networks, Figure 1 shows no difference between optimal and heuristic solutions in terms of communication cost. We can see that both solutions entail lower cost than the no filtering solution. However, according to Figure 2, a significant difference exists in the run-time of the heuristic algorithm compared to the naive optimal one. This difference is still significant as we compare the heuristic solution to the optimization of naive optimal solution which does not consider "useless" agents. Thus, in equal cost networks, it is preferable to choose the heuristic algorithm due to its smaller runtime.

Analyzing the results in two type networks, we come to the same conclusion. Figure 3 shows, similar to equal cost networks, that there is no significant difference between communication costs achieved by the optimal and the heuristic solutions. Still both perform better than the no filtering solution. Comparing the run-time of the algorithms, Figure 4 shows that the heuristic algorithm has an execution time close to zero, while the naive optimal algorithm (and even its optimization) has an execution time several orders of magnitude higher. This difference is so acute that simulations with sets of 100 communicating agents would not finish even after one hour, and thus had to be discarded. Thus, Figures 3 and 4 both present the results for sets of up to 30 communicating agents.

The main reason for the difference between the equal cost networks and the two-type networks is the average path length between communicating agents in those networks. In equal cost networks, this average path length is relatively small, thus so is the number of intermediate agents. Accordingly, its effect on the run-time complexity is less significant. In contrast, two-type networks have very long paths, and thus they

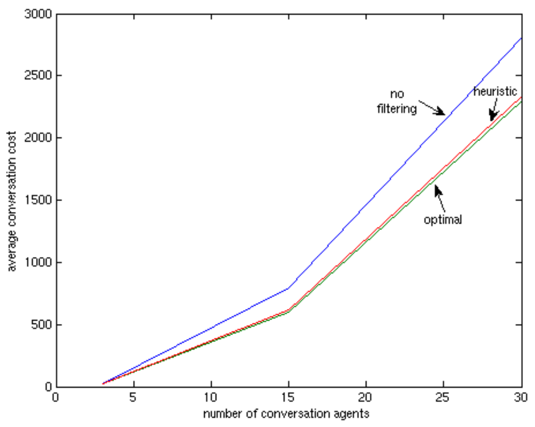

Fig. 3. Communication costs in two type networks

significantly affect the algorithms' run-time due to the higher number of intermediate agents in such long paths.

6 Summary & Future Work

This paper shows the use of overhearing for reducing the inter-agent communication cost associated with task execution in open distributed MAS. In such settings, it is highly improbable for two communicating agents to have a direct communication channel. Instead, their communications will be routed through a chain of intermediate agents. Allowing the intermediate agents to overhear and monitor communications of their peers, some errors in those communications can be detected a priori. Filtering those erroneous messages before they reach their final destination, reduces the unnecessary cost of routing them through the system.

Our work first formalizes this problem. Then, based on the proposed model, we provide algorithms for finding an effective set of filtering agents. Unfortunately, the optimal algorithm has been found to be exponential in the number of intermediate agents. Therefore, such algorithm cannot be applied in real-world settings. As an alternative, an efficient heuristic solution is proposed. This algorithm has a polynomial run-time complexity. To analyze the effectiveness of the heuristic solution, we empirically evaluated both algorithms. The results of the performed experiments show that the proposed heuristic algorithm achieves similar effectiveness to the optimal solution in less time.

Nonetheless, our approach has some limitations. Firstly, paths between conversation agents are to be fixed before the conversation starts. This does not allow the current algorithms to be used in highly dynamic environments. In future work we will study how new paths can be added to the conversation at runtime as the network changes. Secondly, we assume filtering agents will not filter invalid messages. In open multi-agent systems, not all agents may comply. For these cases paths through the network must be searched were mediators are known to be truthful. Finally, we have not taken

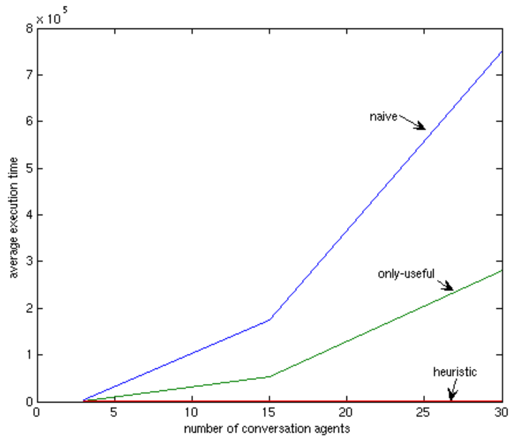

Fig. 4. Execution time in two type networks

into account the cost of receiving an invalid message by a conversing agent. When this cost is high the difference between the optimal and the heuristic solution will tend to be higher.

The algorithms, shown in this paper, assume that each filtering agent acts independently of the other filtering agents. Accordingly, its ability to filter depends on the amount of information it has on the monitored communications. Thus, it might not always be able to filter all erroneous messages. In our future research, we intend to investigate cases of cooperative filtering accomplished through communications between filtering agents as done in EI systems. These communications between filtering agents allow to synchronize information on the monitored communications, and thus to perform filtering more effectively. However, this improved ability to filter carries some cost associated with the communication between the filtering agents. Nevertheless this approach is the only valid one when the cost of receiving an invalid message is not assumable. Our future work will focus on examining this trade-off.

Acknowledgements

This work has been partially supported by the OpenKnowledge STREP[3], funded by the European Commission under contract number FP6-027253, by the Agreement Technologies CONSOLIDER project under contract CSD2007-0022 and INGENIO 2010, by the IEA project under contract TIN2006-15662-C02-01, and by the Generalitat de Catalunya under the grant 2005-SGR-00093

A. Perreau de Pinninck is supported by a CSIC predoctoral fellowship under the I3P program, which is partially funded by the European Social Fund.

[3] `http://www.openk.org`

References

1. M. Aiello, P. Busetta, A. Dona, and L. Serafini. Ontological overhearing. In *Proceedings of ATAL-01*, 2001.
2. Josep Lluis Arcos, Marc Esteva, Pablo Noriega, Juan A. Rodríguez-Aguilar, and Carles Sierra. Engineering open evironments with electronic institutions. *Engineering applications of artificial intelligence.*, 18(2):191 – 204, 2005.
3. P. Busetta, L. Serafini, D. Singh, and F. Zini. Extending multi-agent cooperation by overhearing. In *Proceedings of CoopIS-01*, 2001.
4. G. Cabri, L. Ferrari, L. Leonardi, and R. Quitadamo. Collaboration-driven role suggestion for agents. In *Proc. of Workshop on Distributed Intelligent Systems*, 2006.
5. S. Cranefield. Modelling and monitoring social expectations in multi-agent systems. In *Coordination, Organizations, Institutions, and Norms in Multi-Agent Systems II*. Springer-Verlag, 2007.
6. David Eppstein and Joseph Yannkae Wang. A steady state model for graph power laws. In *2nd Int. Worksh. Web Dynamics*, May 2002.
7. P Erdos and A Renyi. *On the evolution of random graphs*, volume 5, page 17. Publ. Math. Inst. Hung. Acad. Sci., 1960.
8. Marc Esteva. *Electronic Institutions: From specification to development*. PhD thesis, IIIA - CSIC, 2003.
9. X. Fan and J. Yen. Conversation pattern-based anticipation of teammates' information needs via overhearing. In *Proceedings of IAT-05*, 2005.
10. G. Gutnik and G.A. Kaminka. Towards a formal approach to overhearing: Algorithms for conversation identification. In *Proceedings of AAMAS-04*, 2004.
11. Marcel Hiel. *Learning Interaction Protocols by Overhearign*. PhD thesis, Utrech University, 2005.
12. G.A. Kaminka, D.V. Pynadath, and M. Tambe. Monitoring teams by overhearing: A multi-agent plan-recognition approach. *JAIR*, 17, 2002.
13. F. Legras. Using overhearing for local group formation. In *Proceedings of AAMAS-02*, 2002.
14. D.G. Novick and K. Ward. Mutual beliefs of multiple conversants: A computational model of collaboration in air traffic control. In *Proceedings of AAAI-93*, 1993.
15. D. Poutakidis, L. Padgham, and M. Winikoff. Debugging multi-agent systems using design artifacts: The case of interaction protocols. In *Proceedings of AAMAS-02*, 2002.
16. Juan Antonio Rodriguez-Aguilar. *On the Design and Construction of Agent-Mediated Electronic Institutions*. PhD thesis, 2001.
17. S. Rossi and P. Busetta. Towards monitoring of group interactions and social roles via overhearing. In *Proceedings of CIA-04*, 2004.
18. S. Rossi and P. Busetta. With a little help from a friend: Applying overhearing to teamwork. In *Proc. of Workshop on Modelling Others from Observations (MOO)*, 2005.
19. Daniel Stutzbach, Reza Rejaie, and Subhabrata Sen. Characterizing unstructured overlay topologies in modern p2p file-sharing systems. In *Proc. of the Internet Measurement Conf.*, 2005.
20. Duncan J. Watts and Steven H. Strogatz. Collective dynamics of small-world networks. *Nature*, (393):440–442, 1998.