Adaptation of Organizational Models for Multi-Agent Systems based on Max Flow Networks

Mark Hoogendoorn
Vrije Universiteit Amsterdam, Department of Artificial Intelligence
De Boelelaan 1081a, 1081 HV Amsterdam, The Netherlands
mhoogen@cs.vu.nl http://www.cs.vu.nl/~mhoogen

Abstract

Organizational models within multi-agent systems literature are of a static nature. Depending upon circumstances adaptation of the organizational model can be essential to ensure a continuous successful function of the system. This paper presents an approach based on max flow networks to dynamically adapt organizational models to environmental fluctuation. First, a formal mapping between a well-known organizational modeling framework and max flow networks is presented. Having such a mapping maintains the insightful structure of an organizational model whereas specifying efficient adaptation algorithms based on max flow networks can be done as well. Thereafter two adaptation mechanisms based on max flow networks are introduced each being appropriate for different environmental characteristics.

1 Introduction

With the need for more complex software, arose the need for a higher abstraction level than the concept agent. As a result, organization modeling is becoming a practiced stage within multi-agent system design (see e.g.[Ciancarini and Wooldridge, 2001] and [Boissier *et al.*, 2005]). The organizational model poses various constraints on agents populating the organization. Frameworks have been introduced for representing such an organizational model e.g. AGR (Agent/Group/Role) [Ferber and Gutknecht, 1998], GAIA [Zambonelli *et al.*, 2001] and MOISE [Hannoun *et al.*, 2000].

A common problem encountered with the current organizational modeling frameworks is their static nature. The frameworks do not support the organizational model itself to be dynamic in that it changes based on e.g. the environment. Especially when a multi-agent system participates in a dynamic and unpredictable environment, an organizational model might become obsolete, making dynamic adaptation of the model essential. Imagine a design for a negotiation system on the Internet, in which buyer and seller agents are present. The organizational model specifies the number of buyers and seller that should be present, based on a certain expected input for the system. Suddenly, an increase in usage requires much more buyer and seller agents. Such an event requires adaptation of the current organizational model to the new usage level, otherwise the system would no longer function correctly due to overload.

The aim of this paper is to introduce a method for capacity management of organizations by dynamically adapting an organizational model based on the environmental fluctuation. For this purpose, the AGR framework is adopted. AGR has been chosen because the framework is closely related to graph representations. Furthermore, extensions of the framework with capacities for each of the elements within the organizational model are introduced. The method itself is based on graph theory, and more specifically, on max flow networks. Specifying methods for adaptation in max flow networks has the advantage of efficient algorithms being available to perform calculations on the network. The method can be incorporated into an agent maintaining such an organizational model, attributing the agent with the capabilities to properly adapt the organizational model. This paper however only deals with evaluating the effectiveness of the method itself.

The paper is organized as follows: Section 2 introduces max flow networks and the terminology associated with it. Thereafter, Section 3 discusses the existing modeling framework for organizations and extends it with capacity elements. Section 4 presents a mapping between the extended organizational modeling framework and max flow networks. Static analysis methods for analyzing the current functioning of the organizational model are presented in Section 5 whereas Section 6 expresses adaptation rules that can be used when the analysis shows an improper functioning. Section 7 presents simulation results of these adaptation mechanisms, and finally, Section 8 is a discussion.

2 Max Flow Networks

This Section provides a brief introduction to max flow networks within graph theory. Max flow theory (see e.g. [Ford and Fulkerson, 1956]) is a very well known part of graph theory, appreciated because of its practical applicability. A max flow network is defined as follows.

Let $G=(V,E)$ be a directed graph with a set of nodes V and a set of edges E. Within V two special nodes are distinguished, namely the source $s \in V$ and the sink $t \in V$. The source has an indegree of 0 and the sink an outdegree of 0. Furthermore, let $c: E \rightarrow \mathbb{R}^+$ be a capacity function for the edges. A network

is then defined as $N = (V,E,s,t,c)$. Now let $f: E \rightarrow \mathbb{R}_0^+$ denote the *flow value* under the following conditions:

$f(x,y) \leq c(x,y)$ for all (x,y) $\in$ E

$f_{in}(v) = f_{out}(v)$ for all v $\in$ V-{s,t}

Where $f_{in}(v)$ and $f_{out}(v)$, respectively the inflow and the outflow of a node *v*, are defined as follows:

$f_{in}(v) = \sum_{x \in V} f(x,v)$ with $(x,v) \in E$

$f_{out}(v) = \sum_{y \in V} f(v,y)$ with $(v,y) \in E$

For the source *s* and the sink *t* the following thus holds:

$f_{in}(s) = f_{out}(t) = 0$

The flow value throughout the network is now defined as follows:

$|f| := f_{out}(s)$ where $f_{out}(s)=f_{in}(t)$

The *max flow* is the maximum among all flows, and the *max flow problem* is to find such a flow. Several algorithms have been published which can find such a flow, in 1956 Ford and Fulkerson [Ford and Fulkerson, 1956] were the first to publish such an algorithm where finding a minimal cut for the graph was proven to be equal to the max flow. Later efficiency improvements have been proposed, see e.g. [Edmunds and Karp, 1972].

To enable a formal mapping between the organizational modeling framework and max flow network, node capacities should be expressible. Specifying capacities for nodes can be incorporated into the classical max flow network as follows: let $c_{node} : V \rightarrow \mathbb{R}^+$ denote the capacity of such a node. Now split up the node *v* with capacity $c_{node}(v)$ into two nodes: v_1 and v_2 where node v_1 inherits all incoming nodes of *v* whereas v_2 inherits all outgoing edges. Finally, draw an edge (v_1,v_2) with the following capacity value:

$c(v_1,v_2) = c_{node}(v)$

3 Multi-Agent Organizational Framework and Extensions

In this Section, the AGR approach is introduced. AGR is used because the representation of the organizational modeling framework is closest to graph theory. As the purpose of this paper is to investigate adaptations in the capacity of an organizational model, AGR is extended with elements specifying such capacity.

3.1 Agent/Group/Role approach

As a basis for representing a multi-agent organization the AGR approach introduced by Ferber and Gutknecht [Ferber and Gutknecht, 1998] is used. In the approach, as the name already suggests, three main elements are used: (1) the **agent** which is only specified as an active communicating entity which plays roles within groups; (2) the **group** defined as atomic sets of agent behavior, and (3) the **role** which is an abstract representation of an agent function, service or identification within a group. More formally on an abstracter level, Ferber and Gutknecht define a *group structure* as a tuple $S=\langle R,G,L\rangle$. In the definition, *R* is a set of role identifiers whereas *G* is an interaction graph specifying the valid interactions between two roles (later referred to as role links): $G:R\times R\rightarrow L$, where *L* is the interaction language. The *organizational structure* is defined as the set of group structures expressing the design of a multi-agent organization scheme. It is expressed as $O=\langle S,Rep\rangle$, where *S* is a set of group structures. *Rep* is a representative graph specifying interactions between role of different groups (later referred to as group links): $Rep:S\times R\times S\times R$, e.g. Rep($S_a$, r_1, S_b, r_2) where $r_1 \in S_a$ and $r_2 \in S_b$, and $S_a, S_b \in S$. A contraint is that a single agent is playing both role r_1 and role r_2

3.2 Agent/Group/Role Extensions

In addition to the AGR approach, it is assumed that in the specification of roles a certain capacity is present. This capacity places a requirement on what an agent to be allocated to the role within the organization should be able to handle computationally per time unit (universal for the whole organization). This capacity is denoted by $RC: R \rightarrow \mathbb{R}^+$. In addition, a capacity can also be set for role links: $CC: R \times R \rightarrow \mathbb{R}^+$ and group links: $SC: R \times R \rightarrow \mathbb{R}^+$.

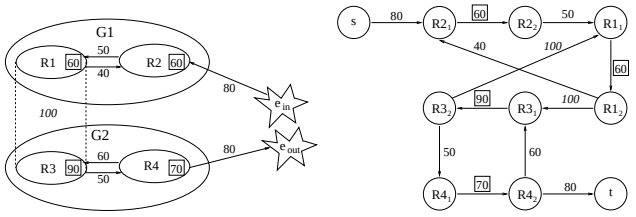
One crucial aspect is however still missing, namely the interaction with the environment. AGR is mainly based on interaction between roles, whereas the emphasis of this paper is to adapt to environmental fluctuations. Therefore, it is assumed that the environment causes a certain pressure upon the organization. Such pressure is expressed as the amount of processing needed by the organization to deal with the pressure, it can be seen as the demand of the environment upon the system. In the organizational model this is represented by adding an entity called the environment (e_{in}) and having links from the environment to the roles receiving that stress directly: (e_{in}, r_i) which has a particular value at a certain time point: $E_{in}: R \rightarrow \mathbb{R}^+$. At different points in time the amount of pressure can differ, requiring different processing capabilities. Furthermore, besides receiving pressure from the environment, most roles are assumed to perform actions in the environment (e_{out}) as well, affecting the environment: (r_i, e_{out}) which again has a value: $E_{out}: R \rightarrow \mathbb{R}^+$. Assumed is that a correctly functioning multi-agent organization affects the environment to the exact same amount as the environment affects the organization.

4 Mapping the Organizational Framework to Max Flow Networks

A mapping between the extended AGR model and the max flow networks as introduced in Section 2 is presented. The translation algorithm of the extended AGR model to a max flow network can be described as follows:

- For each role $r_i \in O$ create a node v_i
- For each role link $G(r_i, r_j)$ create an edge (v_i, v_j)
- For each role link with capacity $CC(r_i, r_j)$ set the capacity of the edge (v_i, v_j) to that value: $c(v_i, v_j)=CC(r_i, r_j)$
- For each group link (r_i, r_j) where $r_i \in S_a$, $r_j \in S_b$ and $S_a \neq S_b$ create an edge (v_i, v_j)
- For each group link with capacity $SC(r_i, r_j)$ set the capacity of the edge (v_i, v_j) to that value: $c(v_i, v_j)=SC(r_i, r_j)$
- For each role with capacity $RC(r_i)$ set the capacity of the node v_i to that value: $c_{node}(v_i)=RC(r_i)$, reduce the graph to a classical max flow graph using the method presented in Section 2

- Add a node *s* to represent the environment e_{in} and add a node *t* to represent the environment e_{out}
- For each (e_{in}, r_i) with capacity $E_{in}(r_i)$ create an edge (s, v_i). Set the capacity c(s, v_i)=$E_{in}(r_i)$
- For each (r_i, e_{in}) with capacity $E_{out}(r_i)$ create an edge (v_i, t). Set the capacity c(v_i, t)=$E_{out}(r_i)$



(a) AGR Representation (b) Max Flow equivalent

Figure 1: Example organizational model

Figure 1(a) shows an example AGR organization. In the figure, the big ovals denote groups, whereas the smaller ovals denote the roles. Capacities of roles are depicted as a box with a number specifying the capacity. Furthermore, interactions between roles within a group or between a role and the environment are depicted by arrows, including a label specifying the capacity. Finally, capacities for interactions between groups are specified by dashed lines, including a capacity number depicted in italics. Figure 1(b) shows the accompanying max flow network using the previously presented translation algorithm (including the translation of node capacity to a classical max flow network) with a *max flow* of 50.

5 Analyzing an Organizational Model using the Max Flow Equivalent

Now that an equivalent max flow network can be derived from the extended form of an AGR organizational model, this Section shows in what way the max flow network can aid in the analysis of the organizational model. Such an analysis can be performed in two ways: (1) checking whether the current organizational model can meet the *expected* environmental conditions (i.e. an analysis beforehand), and (2) checking at runtime whether the organizational model can meet the *actual* environmental conditions.

5.1 Analysis of the Organizational Model based on Expected Values

Creating an organizational model is done having certain requirements in mind. In this paper organizational requirements in the form of organizational capacities are considered. Requirements on capacities are in the form of pressure from the environment (i.e. the organization should be able to handle *x* requests of a certain type). For checking whether such a requirement can theoretically be met by a multi-agent organization, the max flow equivalent can be used, enabling the usage of tools and algorithms from graph theory. To this end an organizational model, including capacities for the various organization elements as introduced in the previous section, is translated into a max flow problem. Remember the notation for the flow: $|f| := f_{out}(s)$ where $f_{out}(s)=f_{in}(t)$. Now let the maximum flow be noted as follows:

$|f|_{max}(N) := f_{out,max}(s)$ where $f_{out,max}(s)=f_{in,max}(t)$

The requirements posed for the organizational model can be translated into requirements on the flow. Requirements regarding the amount of pressure can be translated to a max flow requirement of the form

$f_{out,max}(s) \geq r$

where *r* is the requirement. Using the max flow problem algorithms from e.g. [Ford and Fulkerson, 1956],[Edmunds and Karp, 1972] it can be determined whether the organization can theoretically fulfill the requirements. Note that non-fulfillment of the requirements *guarantees* that the organization will never be able to meet the requirements when complying to the design specification.

5.2 Analysis of the Organizational Model based on Observed Values

Besides performing an analysis at design time, an analysis at runtime can also be performed. It can be the case that the environment in which the multi-agent system is participating is highly dynamic and hard to predict, causing an unknown amount of pressure for the organization. Therefore, the requirement *r* posed for the system is dynamic. The requirement can however be observed: observing the amount of pressure received by the multi-agent system. In case this exceeds the maximum flow in the network equivalent, the organizational model is incorrect. Note that it is still possible that the multi-agent system is functioning correctly, since the agents allocated to the roles within the organizational model might have a higher capacity than required. The model however should always be updated to make sure that the system continues to function correctly. It could for example be that an agent can handle the pressure for a while, but after a certain duration suffers a burn out. Methods for updating such a model are presented in the next section.

6 Adaptation of an Organizational Model using the Max Flow equivalent

As shown in the previous Section, when participating in a highly dynamic environment the organizational model sometimes needs to be changed in order to handle the environmental fluctuations appropriately. This Section proposes two methods to perform such a change, each working under specific circumstances. These methods only concern extending the capacity as the aim of this paper is to adapt the organizational model in such a way that the environment can be handled, which does not include decreasing the capacity.

6.1 Adapting the Bottlenecks

The first method for improving the network equivalent of the organizational model involves finding the path which requires a minimum additional capacity, and adding capacity to the bottleneck within the path. In other words, pinpointing the bottleneck within the organization and improving it. Let $P = s \longrightarrow ... \longrightarrow t$ denote a path from the source *s* to the sink

t in the network. In the explanation of the method, capacities of edges are assumed to be natural numbers, however this can easily be extended to rational numbers. Given that the environment has imposed requirement *r*, and the parameter η which represents the safety margin to be taken:

- calculate the current max flow of the network $|f|_{max}(N)$
- if $|f|_{max}(N) < (\eta \times r)$:
  1. n=1
  2. try finding a path P of the form $P = s \longrightarrow^{+1} ... \; x_i \longrightarrow^{0} y_i \longrightarrow^{+1} t$ where n edges $\{(x_1,y_1),...,(x_n,y_n)\}$ do not have sufficient capacity to add a flow of 1 to the path. In case such a path cannot be found: n=n+1
  3. set the capacity for all the edges (x_i,y_i) in the set $\{(x_1,y_1),...,(x_n,y_n)\}$ to $c(x_i,y_i) = c(x_i,y_i) + 1$
  4. if the new max flow $|f|_{max}(N) < (\eta \times r)$ then continue at point 2, else the algorithm ends

The specification of the algorithm draws inspiration from the algorithms proposed for finding the max flow through a network. These algorithms work with finding paths from source to sink that can be increased with a flow of 1. Finding a path which can almost be increased and extending the capacity therefore results in an immediate increase in the max flow of the network.

6.2 Adding Organizational Elements

The capacity required for allocation of an agent to a certain role is of course limited; agents with very high capacities might be too expensive. Therefore, an algorithm for adding roles and the accompanying role and group links to an organization is presented here. The algorithm as presented above is therefore adapted to cope with addition of organizational elements as well. The extension states the following: in case the max flow of the network can no longer be increased as a further increase of the max flow would necessarily require exceeding of the maximum capacity set (i.e. the current capacity $c(x_i,y_i)$ exceeds the maximum capacity $c_{max}(x_i,y_i)$):

1. For each two nodes v_{i1}, v_{i2} and set of edges of the form (v_x, v_{i1}) where $x \in V$ and $(v_x, v_{i1}) \in E$, (v_{i1}, v_{i2}), and (v_{i2}, v_y) where $y \in V$ and $(v_{i2}, v_y) \in E$ representing a role r_i and its role and group links:

   If at least one of the elements has reached the maximum capacity: calculate the increase of the max flow in case the nodes and edges were to be doubled (i.e. N_{new}=N$\cup$ $\{v_{i1,2}, v_{i2,2}, (v_x, v_{i1,2}), (v_{i1,2}, v_{i2,2}), (v_{i2,2}, v_y)\}$): $|f|_{max}(N_{new}) - |f|_{max}(N)$
   - If the highest increase exceeds 0, in other words the network can be improved by copying a single role, copy the two nodes and edges having the highest value.
   - Otherwise, copy the two nodes v_{i1}, v_{i2} and accompanying edges (v_x, v_{i1}) where $v_x \in V$ and $(v_x, v_{i1}) \in E$, (v_{i1}, v_{i2}), and (v_{i2}, v_y) where $v_y \in V$ and $(v_{i2}, v_y) \in E$ which maximize the following: $min\ (|\{(v_x, v_{i1})|(v_x, v_{i1}) \in E\}|, |\{(v_{i2}, v_y)|v(_{i2}, v_y) \in E\}|)$

   Where $\{(v_x, v_{i1})|(v_x, v_{i1}) \in E\}$ is the set of incoming edges of v_{i1} and $\{(v_{i2}, v_y)|v(_{i2}, v_y) \in E\}$ the set of outgoing edges of v_{i2}. In other words, take the node which has a maximal connection with other nodes. Thereafter, return to 1.

The intuition behind the algorithm is to find the nodes and edges representing the role of which a copy would improve the max flow most. If no increase is possible, take the nodes and edges representing the role which is most connected within the organization to maximize the chances that an addition of a role after the copy will result in an increase of the max flow.

7 Evaluation of the Different Methods for Improving an Organizational Model

In order to characterize the methods presented in the previous Section, this section presents an evaluation method, and compares the results of the different methods using different settings. First, a cost model is presented expressing the cost function used for evaluation. Thereafter, the different methods are evaluated based on the cost model and several environmental settings.

7.1 Cost Model

Each element within the organizational model has a certain cost attached to it. The cost for role and group links is expressed as $cost_{link}$: $R \times R \rightarrow \mathbb{R}^+$ and defined as follows:

$cost_{link}(x,y) = e^{\frac{CC(x,y)}{\alpha}}$ where $(x,y) \in (G \cup Rep)$

In other words, cost for links increase exponentially (a commonly used type of cost function), where the parameter α can be varied. The same holds for the cost of a node $cost_{role}$: $R \rightarrow \mathbb{R}^+$ which is thus defined as follows:

$cost_{role}(r) = e^{\frac{RC(r)}{0.5\alpha}}$ where $r \in R$

The factor 0.5 is arbitrarily set in the cost function because typically interaction capacity costs are lower than agent capacity costs. In order to punish an organizational model not being able to meet the environmental pressure, a penalty is introduced of the form p: $N \times r \rightarrow \mathbb{R}^+$, where r is the environmental requirement. The penalty function is defined as follows:

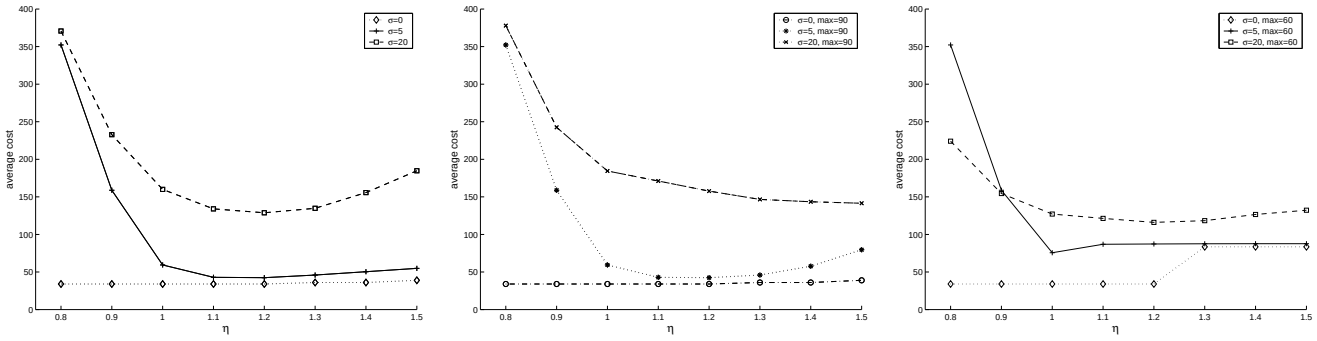
$p(N, r) = \beta \times (r - |f|_{max}(N))$

The parameter β specifies the penalty for each requirement unit not fulfilled. The network N represents the multi-agent organization. Finally, the overall cost for the organizational model is defined as follows:

$cost_{total} = p(N, r) + \sum_{r \in R} cost_{role}(r) + \sum_{(x,y) \in G \cup Rep} cost_{link}(x,y)$

7.2 Evaluation

An implementation in Java has been created of the two algorithms for reorganization, and the translation procedure of an organizational model to the accompanying network. In order to evaluate the performance of the two algorithms the example organizational model shown in Figure 1 is used. As a benchmark, no adaptation of the organizational model is



(a) bottleneck (b) role addition, max capacity=90 (c) role addition, max capacity=60

Figure 2: Bottleneck and role addition algorithm performance

used. Using the implementation, simulation runs are performed with an environmental pressure causing a requirement *r* based on a normal distribution *f(x;μ,σ)*. One step within such a simulation entails: (1) generating the environmental requirement *r* based on the normal distribution; (2) calculating the current max flow of the network: $|f|_{max}(N)$; (3) calculating the cost $cost_{total}$, and (4) updating the network for the next step, using one of the improvement methods. Each step is performed 100 times, and each simulation is performed 10 times, generating from a different seed each time. After the steps have been performed, the average of the $cost_{total}$ per step is calculated. In order to evaluate the different methods, two settings for the cost model have been used: relatively high penalty cost compared to agent/communication cost (e.g. a critical domain such as incident management), and relatively low penalty cost compared to agent/communication cost (a non-critical domain). Furthermore, the environment setting μ is by default set to the initial max flow (in this case 50), whereas the fluctuation σ has been set to different values: σ=$\{$0,5,20$\}$.

Relatively High Penalty Cost

The results of an organization in which penalty costs are relatively high compared to labor cost are presented here. This reflects in the cost model in which α is set to 80, meaning an initial cost of 34 for the whole organization, whereas β is set to 200, a penalty cost significantly higher than the initial cost of the whole organization.

First, the bottleneck algorithm is used. Figure 2(a) shows the results for different setting given environmental fluctuation σ and varying η value (the algorithm parameter specifying how much to update the capacity). As can be seen, with no environmental fluctuation, the costs stay stable for the lower η values whereas they slightly increase for the higher settings, as the capacities are even increased when the current max flow is identical to the requirement r. For small environmental fluctuation (σ=5) an η value of 1.2 gives the best results, which is also the case for large environmental fluctuation. In both cases, having a higher η value results in the capacity being too high (i.e. costs too high) whereas a lower η value increases the amount of penalties.

Figure 2(b) and (c) show the results of the bottleneck algorithm extended with role addition. In Figure 2(b) the max capacity has been set to 90 whereas in Figure 2(c) 60 is used. In the case of no environmental fluctuation, Figure 2(b) shows a similar shape as the bottleneck algorithm, whereas Figure 2(c) shows a large increase above a value of η=1.2. This is the result of roles being copied due to the low setting of the maximum capacity. Each role in the network has already reached the maximum capacity, resulting in a need for a copy, which causes a severe overcapacity. In case of little environmental fluctuation, the setting 90 for the maximum capacity is better since no large increase for capacity is required, whereas with high environmental fluctuation 60 is best since larger capacity increases are required.

β	σ	**No adaptation max flow μ**	**No adaptation max flow 2μ**	**Optimal Bottleneck**	**Role addition max=90**	**Role addition max=60**
200	0	34	199	34	34	34
200	5	370	199	42	42	75.6
200	20	1518	210	129	141	116
5	0	34	199	34($\eta \leq 1.2$)	34($\eta = \leq 1.2$)	34($\eta = \leq 1.2$)
5	5	42	199	35(η=1.0)	35(η=1.0)	37(η=0.9)
5	20	71	210	35(η=0.8)	37(η=0.7)	69(η=0.5)

Table 1: Average costs of the organizational model resulting from the different algorithms, penalty is set to either 5 or 200

Finally, the top part of Table 1 (the rows where β=200) shows the comparison between the different methods, given optimal parameter settings. As can be seen, no adaptation is always worse compared to the other algorithms, even when an overcapacity is initially present. Furthermore, in case of small environmental fluctuation adding capacity to the current roles is best (despite the exponential cost function) whereas for larger fluctuation, adding roles immediately is the best option due to the exponential cost function.

Relatively Low Penalty Cost

Table 1 also shows the result when the penalty is set to a relatively low value (the rows where β=5). Still the algorithms are better than no adaptation, however the value for η needs to be set to a lower value as having too much capacity is relatively expensive compared to the penalty for not meeting the requirements. Therefore, the role addition algorithm with a maximum capacity 60 performs worse than the one with 90.

8 Discussion

This paper has presented an approach to adapt an organizational model to fluctuations within the environment. Such an approach can be incorporated into an agent responsible for maintaining the organizational model. In a highly dynamic and unpredictable environment, such an adaptation mechanism might be a necessity to guarantee successfulness of a multi-agent system. The approach used in this paper is to translate an organizational model to a max flow network and specify two algorithms for adaptation. Specifying analysis constructs and reorganization algorithms for the graph representation has as an advantage that knowledge and algorithms from the well established graph domain can be reused (such as calculation of the max flow [Edmunds and Karp, 1972]). The algorithm used for adaptation of bottlenecks is indeed a known algorithm within graph theory for addition of capacity. The addition of organizational elements however requires knowledge about the meaning of the elements within the max flow network (e.g. what the roles are), algorithms from graph theory can therefore not be re-used. The algorithms as proposed in this paper have been evaluated by simulation runs, and were shown to be more effective compared to no adaptation, especially in critical domains in which the penalty function is relatively high. Limits of the approach for instance include the case where the environmental pressure is at first above the capacity of the organizational model and thereafter steadily decreases. In such cases the organizational model using the adaptation mechanism will adapt to the initial high pressure, and suffer from an overcapacity at the later time points. The total sum of the penalties received in the beginning by the non-adaptive organizational model might be lower than the total cost of the overcapacity in the adaptive case. Very rare outliers with a very high environmental pressure can have the same effect.

In the field of adaptive agents and multi-agent systems (see e.g. [Schaal *et al.*, 2004][Kudenko *et al.*, 2005]) learning from the environment is an important topic. Adapting organizational models based on the environmental conditions is, as argued before in this paper, one of the necessities for the new organizational paradigm. Especially with continuously changing circumstances and agents leaving and arriving a well specified and up-to-date organizational model is required to guarantee proper functioning of the organization.

When comparing the approach with other organizational modeling approaches in multi-agent systems, those approaches often include much more concepts than capacity of a role. An extension of GAIA [Zambonelli *et al.*, 2001] for example adds the notion of organizational rules. Such rules express relationships and constraints between roles, protocols, and roles and protocols. These relationships and constraints can be incorporated in the approach presented in this paper as well. When copying a role one can simply copy those relationships involving the role being copied and adapt them to specify the relationships and constraints of the copy of the role. MOISE [Hannoun *et al.*, 2000] defines missions for roles, which can include concepts such as goals, plans, actions, and resources. Furthermore, authority links between roles can also be specified. Again, as already stated for GAIA, these concepts can be reused when copying a role. Several of these organizational models have been extended with organizational change notions, see for example [Hoogendoorn *et al.*, 2006]. These extensions are however typically very generic models without going into specific details on how to reorganize the organization whereas this paper does.

Finally, for future work an interesting continuation would be to look at the performance in other simulation settings, and possibly compare how well different types of organizational structures perform in changing circumstances.

Acknowledgements

The author would like to thank Catholijn Jonker, Jan Treur, and Evert Wattel for the fruitful discussions, and the anonymous reviewers for their useful comments.

References

[Boissier *et al.*, 2005] O. Boissier, V. Dignum, E. Matson, and J. Sichman, editors. *Proceedings of the First OOOP Workshop*. Utrecht, 2005.

[Ciancarini and Wooldridge, 2001] P. Ciancarini and M. Wooldridge, editors. *Agent-Oriented Software Engineering*. LNCS. Spinger Verlag, 1957 edition, 2001.

[Edmunds and Karp, 1972] J. Edmunds and R.M. Karp. Theoretical improvements in algorithmic efficiency for netowrk flow problems. *Journal of the Association for Computing Machinery*, 19:248–264, 1972.

[Ferber and Gutknecht, 1998] J. Ferber and O. Gutknecht. A meta-model for the analysis and design of organizations in multiagent systems. In *Proc. of ICMAS 1998*, pages 128–135. IEEE CS Press, 1998.

[Ford and Fulkerson, 1956] L.R. Ford and D.R. Fulkerson. Maximum flow though a network. *Canadian Journal of Mathematics*, 8:399–404, 1956.

[Hannoun *et al.*, 2000] M. Hannoun, O. Boissier, J. Sichman, and C. Sayettat. Moise: An organizational model for multi-agent systems. In *IBERAMIA-SBIA '00: Proceedings of the International Joint Conference, 7th Ibero-American Conference on AI*, pages 156–165. Springer-Verlag, 2000.

[Hoogendoorn *et al.*, 2006] M. Hoogendoorn, C.M. Jonker, M.C. Schut, and J. Treur. Modelling centralized organisation of organisational change. *Computational and Mathematical Organization Theory*, In Press, 2006.

[Kudenko *et al.*, 2005] D. Kudenko, D. Kazakov, and E. Alonso, editors. *Adaptive Agents and Multi-Agent Systems II*. Springer Verlag, 2005.

[Schaal *et al.*, 2004] S. Schaal, A.J. Ijspeert, A. Billard, S. Vijayakumar, J. Hallam, and J.A. Meyer, editors. *From animals to animats 8*. MIT Press, 2004.

[Zambonelli *et al.*, 2001] F. Zambonelli, N. Jennings, and M. Wooldridge. Organizational rules as an abstraction for the analysis and design of multi-agent systems. *Journal of Software and Knowledge Engineering*, 11:303–328, 2001.