A Multi-agent Paradigm for the Inter-domain Demand Allocation Process

M. Calisti and Boi Faltings

Laboratoire de Intelligence Artificielle
Swiss Federal Institute of Technology
CH-1015 Lausanne, Switzerland
{calisti, faltings}@lia.di.epfl.ch
http://liawww.epfl.ch/ calisti

Abstract. Market liberalisation and increasing demands for the allocation of services which span several networks are pushing every network operator to evolve the way of interacting with peer operators. The Inter-domain Demand Allocation (IDA) process is a very complex task for several reasons: there are different actors involved, end-to-end routing must take into account QoS requirements, network resources and information are distributed, etc. In this paper we address the problem of the QoS-based multi-domain routing, and more specifically a multi-agent paradigm for supporting the IDA process is defined. We show how Artificial Intelligence methods for distributed problem solving supply a compact way to formalise the multi-domain routing process, and how this formalism enables an agent middle-ware to route demands across distinct domains.

Keywords: Inter-domain routing, service demand allocation, constraint-based routing, agents, network provider.

1 Introduction

Today's networks are controlled at various organisational and functional layers by human managers, in particular the management of end-to-end services across different telecom operators is largely non automated. What seems more suitable for the future scenarios, is a management solution based on static and/or mobile software entities [19], collecting network state information and which have the ability to directly invoke effective changes to switch controllers, without the interaction of a human operator. Such a dynamic and active (reactive and pro-active) paradigm would improve both intra-domain and inter-domain routing capabilities.

The main factors that are pushing Network Operators, NOs, to evolve the way they interact with each other are: the market liberalisation, the technological rate of change, the need for reducing time to market for new services, the aim to enable end-to-end managements of services, the co-ordination of multi-vendors and multi-jurisdictional environments, the need for scalable and flexible systems for future development.

R. Stadler and B. Stiller (Eds.): DSOM'99, LNCS 1700, pp. 76-88, 1999.

There are several approaches to multi-domain connection management[1], e.g., TINA with LNFed reference points [2], the TMN X-interface [12], [6], [7], the mobile agent paradigm using OMG's MAF [8] (Object Management Group, Mobile Agent Facility). Furthermore, extensive work has been devoted for supporting the inter-working between CMIS/GDMO and interfaces specified in the Interface Definition Language (for good references check [9]). Nevertheless, more study is needed for the definition of agent-oriented APIs (Application Programming Interfaces) for the translation of TMN or TINA terms and the associated information models. Additionally, there are still open issues regarding procedures and interactions to achieve connectivity across several domains indipendentely on the under-laying network technology.

Our work aims to address these open issues. This paper examines the correlation between intra- and inter-domain routing, and proposes a mechanism to make local routes consistent with inter-domain constraints. In particular it introduces the use of Artificial Intelligent techniques together with multi-agent system technology, in order to face the *Inter-domain Demand Allocation*, IDA, problem. Agents (static and mobile) can be deployed for dynamically configuring the resources of switches and cross-connects [13]. Agents inside distinct domains can run different routing algorithms, which can be modified by the agents themselves (e.g., changing, adding or modifying metrics). Furthermore, agents can actively modify the strategy they adopt for inter-domain routing.

Section 2 describes the problem we address. Section 3 presents the *Network Provider Interworking*, NPI, paradigm and shows how to apply the Distributed Constraint Satisfaction techniques to formalise the IDA process. Section 4 focuses on the solving algorithm. Some first evaluation results are shown in Section 5. In Section 6 we initiate a discussion on a possible deployment of NPI in a real scenario. Some final remarks and comments on future work are given in Section 7.

2 The problem

The IDA process is a very complex task for NOs for several reasons: there are distinct entities involved (final customers, service providers, service brokers, etc.), the routing must take into account QoS requirements, network resources and information are distributed, a trade-off between the profit optimisation and the end-user satisfaction is required. Figure 1 shows the principal entities involved in this process. An end user is represented by an End User Agent (EUA). This entity can contact one or several Service Provider Agents , SPAs, specifying a service demand and optional negotiation parameters, such as for example a target timeout to receive back an offer. Every SPA can then contact one or several Network Provider Agents, NPAs. Whenever the service demand spans several network domains, several NPAs must interact. A Mask Agent, MA, represents a

[1] We refer the establishment of semi-permanent connections through the management plane.

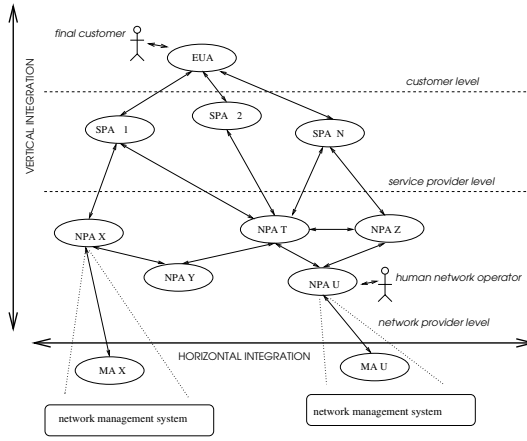

Fig. 1. Dynamic service provisioning scenario.

wrapper for the non-agent-based Network Management System: gateway functionalities must be performed for converting Agent Communication Language messages to, for instance, CMIP (Common Management Information Protocol) primitives.

In the short term such a scenario could be introduced through software tools supporting human operators, for example proposing alternative choices, evaluating prices, QoS and topology constraints, and summarising offers from other operators[2]. This would improve the providers negotiation strategy and would support a more dynamic multi-domain service demand allocation.

Focusing on the interaction among Network Providers every network is represented as a set of nodes and links (Figure 2). Two main sub-problems need to be addressed: (1) Finding the routes which satisfy connectivity and QoS constraints. (2) Selecting a specific route by negotiating with the other providers. This paper focuses on the former sub-task that can be more precisely expressed as it follows. A network operator receiving a demand has:

- To detect the source and the destination network domains. Whenever the destination node resides in a remote network the *inter-domain routing* must be started.
- To compute an abstract path P. An *abstract path* is an ordered list of distinct network provider domains between the source and the destination network domains.
- To contact all the network providers along P.
- To compute the *local routes*, i.e., intra-domain routing inside every network along P.
- To make the set of local routes consistent within inter-domain constraints. All

[2] For an automated version of such a framework more time is required for a larger deployment of agent technology in telecom environments. However, several research activities are focusing on suck kind of scenario [20], [17]

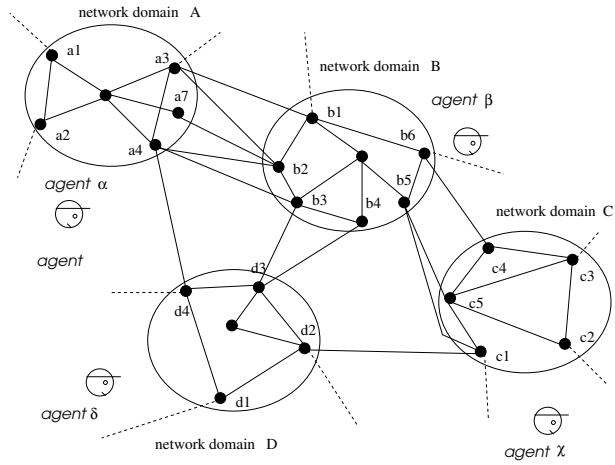

Fig. 2. Network formalisation.

local routes which violates such constraints are discarded.
- To negotiate with the providers along P for allocating a *global route*, i.e., the end-to-end connection consisting of local routes and inter-domain links interconnecting them.
- If an agreement is found the network resources are reserved
- Otherwise the service demand is rejected.
If no answer has been found before a certain timeout, the overall allocation process fails.

3 Network Provider Interworking paradigm

The inter-domain routing requires the coordination of distributed NPAs. All knowledge about the resources needed to allocate a demand cannot be gathered into one single network provider, for various reasons. Fundamentally, for reasons of both scalability and security, network operators do not reveal details of their internal structure. Instead, every network advertises only a summary, or aggregated view, of the costs and availabilities associated with traversing them (this is also proposed by the ATM PNNI standard).

Hence, every NPA assigns a part of the global route, namely the local part inside its network, and then it negotiates with others in order to interconnect its local solution to form a global route.

The interconnection of different providers' network is modelled as a connected abstract network graph G=(*Nodes*, *Links*). A node can correspond to a network node (such as switches and routers), to sub-networks or, at the highest level of abstraction, to networks. The links represent both the connections existing inside every network domain, *intra-domain links*, and the connections to other providers

domains, *inter-domain links*. In our framework a communication demand d_k, or *demand*, is specified by a triple:

$$d_k ::= (x_k, y_k, qos_{req,k})$$

where x_k is the source node, y_k the destination node[3], and $qos_{req,k}$ the required QoS by the demand. We assume that the $qos_{req,k}$ expresses the bandwidth (peak bandwidth) needed by the service. Taking into account other parameters, such as the delay or the number of hops, which are additive metrics, implies to introduce additional constraints in the solving algorithm[4].

3.1 Formalising the route allocation process as a DCSP

Constraint satisfaction is a powerful and extensively used Artificial Intelligence paradigm [21]. *Constraint satisfaction problems* (CSP) involve finding values for problem variables subject to restrictions (constraints) on which combinations of values are acceptable. CSP are solved using search (e.g., backtrack) and inference (e.g., arc consistency) methods.

Finding a route for allocating service demands that cross distinct networks, can be considered as a Distributed Constraint Satisfaction Problem [16], DCSP, since the variables are distributed among agents and since constraints exist among them. We assume that: (1) every agent has exactly one variable, (2) all inter domain constraints are binary, i.e., they involve two variables, (3) there is at least an agent for every domain, (4) all the agents in the scenario know each other, (5) agents communicate using messages.

The *variable* every agent handles is a "local path" (actually it is not a path since it expresses just the two end points of it) specified as a couple:

$$p ::= (inputpoint, outputpoint)$$

The values for each variable are all the possible combinations of boundary points, i.e., input-output points to/from every network, which represent the possible local routes to allocate the demand. Note that only *simple paths*, i.e., loop-free, are considered. The set of all the possible input-output points combinations is the *domain* for each variable. Consider the example depicted in Figure 2. Agent α receives a demand $d_k = (a_1, b_6, qos_{req})$ The variable for agent α is the couple $p_\alpha = (i, o)$ and the domain of p_α is $D_{\alpha,k} = \{(a_1, a_3), (a_1, a_4), (a_1, a_7)\}$. Agent α selectes the abstract path A - B. The domain for agent β is: $D_{\beta,k} = \{(b_1, b_6), (b_2, b_6), (b_3, b_6)\}$. The domains of the variables are dynamically computed for every specific demand. The dynamical re-computation allows

[3] We assumed that a service request corresponds to a single point to point connection request.

[4] There must be a centralised control, for instance performed by the agent in the network provider which first receives the service demand from a service provider, which checks if the sum of all delays along the path does not exceed the delay requirements. These issues are still under investigation.

first to update the variable domains according to the network state, and also to reduce the search space.

There are essentially two categories of constraints: the QoS constraints, that depend on the type of the service requested by the user, and the policies applied by the providers . The first kind of requirements are explicited as qos_{req} in the in the demand (see above). The second class of constraints includes the policies applied by the providers, which correspond to a set of inference rules that can be either static or dynamic. Some of those rules depend on the technology inside every network domain, some others on how the availability of network resources is managed, and finally some others can take into account the past experiences.

Our paradigm does not force the use of any specific intra-domain policy[5]: although distinct network providers can deploy different policies the NPI mechanism is still valid. In our framework the global constraints (QoS and connectivity) are locally translated in constraints between the boundary points, that neighbour network domains use for the inter-domain routing.

4 Solving the IDA process

The *Distributed Arc Consistency* algorithm is based on the use of arc consistency techniques [14], that are used to narrow the space of possible choices before actually performing search. The main principle is that for every existing constraint between two variables i and j, all the values of the variables i and j, that are not consistent with the given constraint, are eliminated. The algorithm consists of the following steps:

1. An agent α receives a demand d.
2. Agent α detects the destination and the source network domains.
3. Based on a global view of the network scenario, which is updated dynamically by the agents, an abstract path P for d is computed. None, one or several paths can exist. If none, the demand is rejected . If more than one path exists the optimal one is selected.
4. Agent α contacts every agent along P. From now on several agents run in parallel similar routines.
5. Each agent defines the variable domain D and the set of constraints C. C is determined considering the available QoS on the inter-domain links interconnecting every provider network with its neighbours.
6. Based on qos_{req} every agent reduces D by performing *node consistency*. Every local path that cannot support the qos_{req} is eliminated from D. The node-consistency process can be supported by the Blocking Island, BI, techniques [15]. The BI formalism allows to quickly detect if there exists a route connecting the input point with an output point guaranteeing the qos_{req}. If the input point i_α and the end point o_α belong to the same β-BI level with

[5] The use of the Blocking Island paradigm, see Section 4, inside every modelled network is not mandatory, although in our opinion it represents a very efficient instrument to check the internal resource availability.

$\beta \geq qos_{req}$, then $(i_\alpha, o_\alpha) \in D$, i.e., there is a local route which guarantees the qos_{req}[6].
If one agent has an empty variable domain D a failure message is sent to all agents along P. A new abstract path can be searched (backtrack to step 3).

7. Establishing arc consistency. Every agent determines all the values of its variable domain which are compatibles with the values contained in the variable domains of the neighbours. If an agent obtains an empty variable domain, a failure message is sent to all agents along P. Then, backtrack to step 3.
8. At least one solution, i.e. one route along P, exists. In order to be sure that a solution will still exist after the negotiation, we assume that there exists a pre-reservation mechanism of the negotiated resources required by the demand *d*. If network failures or physical changes occur the guarantee of having a solution is not valid any more. In this case every agent is notified by the Network Management System and the algorithm backtracks to 3.
9. If the negotiation is successful the resources needed are reserved and the service demand can be allocated, otherwise a failure message is sent to all the agents involved along P and the algorithm goes to 3.

In the worst case scenario, the complexity of the arc consistency algorithm is $O(\mid P \mid N^6)$, with $\mid P \mid$ being the number of networks involved along P and N the maximum number of boundary nodes of a network along P[7].

4.1 Visualizing the DAC mechanism

The DCSP formalism allows to easily visualize which are the links in the scenario that can support the qos_{req}. After the arc consistency propagation, the only links considered are the ones which guarantee the qos_{req} and which satisfy the inter-domain constraints. Consider that NPA A receives the demand d=(a1, b6, 15) (Figure 2). Figure 3 (A) shows the situation before any computation. The routes inside every domain are specified as couples of end-points, e.g., (a1, a3) inside the network domain A. Figure 3 (B) indicates all the links, intra and inter domain, that satisfy the qos_{req}. This is the configuration after every agent has performed steps 5 and 6 of the DAC algorithm. At step 5 the link l3 is excluded since it cannot guarantee the qos_{req}=15. At step 6 every agent checks the node consistency: the local path (b2, b6) is pruned out from the variable domain D_β. Finally, (B) shows D_α={(a1, a3),(a1, a4),(a1, a7)} and D_β={(b1, b6), (b3, b6)}. Figure 3 (C) depicts the situation after the arc consistency propagation (step 7 of the DAC algorithm). The variable domains are : D_α={(a1, a3),(a1, a4)} and D_β={(b1, b6),(b3, b6)}. The value (a1, a7) is pruned out from D_α since (a1, a7) is not consistent with any value in D_β.

[6] **Frei:** *A β-BI for a node x is the set of all nodes of the network reachable from x using links with at least β available resources, including x.*

[7] More detailed considerations about the completeness and the complexity of DAC can be found in [4].

There are two solutions in the end: (a1, a3)-l1-(b1, b6), (a1, a4)-l4-(b3, b6). The agents can now start the negotiation to select which solution to adopt (for brevity's sake the negotiation aspects are not addressed in this paper).

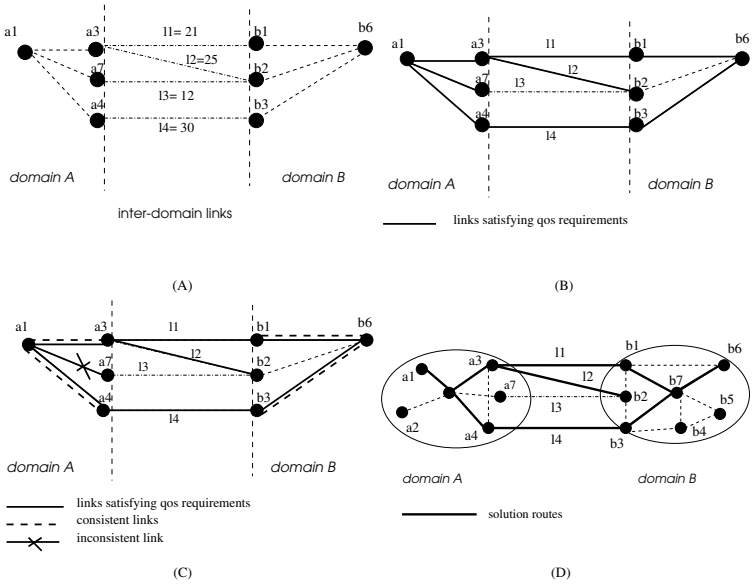

Fig. 3. DAC visualization.

4.2 Global path computation and selection

Every NPA computes the list of abstract paths [8]. The computation is based on Dijkstra's algorithm modified in order to find out the "optimal" paths. An abstract path is *optimal* if there exists no other abstract path that is strictly better in terms of price and that meets bandwidth requirements. The price of an abstract path is the sum of the prices of the abstract links which compose the path itself. The bandwidth of an abstract path is given by the minimum bandwidth value of tits component abstract links. The abstract path computation algorithm takes as input a graph G=(N, L, C, d), where N is a finite non-empty set of nodes, L is the list of links between any two pair of nodes from N and C a function that returns the couple c:=(price, qos) associated with every link l $\in$ L, and finally d is the demand to allocate. The output is the complete set of optimal abstract paths from the demand's source node to all the specified destination node.

The choice of an abstract path P for a service demand d is based on the following heuristics: (1) Eliminate all the paths that do not guarantee enough

[8] On-demand routing: the end-to-end route is computed upon connection request.

bandwidth. (2) Among the paths left select the cheapest (i.e, minimum price). (3) If still more than one path exists, chose the path which has, after having accepted the incoming demand *d*, the largest bandwidth left.

We also tested a different algorithm , the *Fixed Access Points*, FAP, algorithm that mainly differs from DAC for the abstract path specification. With DAC the selected abstract path is a list of network domains: for instance A-B-C. No specific inter-domain links are specified, so that different options are still possible: A-{ l1 | l2 | l3 | l4 | l5}-B-{ l6 | l7}-C. This set is then made consistent and next a specific choice is negotiated. With FAP the abstract path is specified as a list of network domains plus specific inter-domain links, so that the access points for every network are fixed, e.g., A-l2-B-l6-C. In this latter case the NPAs do not need to make arc consistent their local routes. They only have to check that the intra-domain paths terminate at the predefined access points (i.e, the end-points of the pre-selected links). For instance, the local routes inside B are constrained to pass through the the access points b2 and b6.

5 Results

A quantitative evaluation of the performance of DAC and FAP has been obtained by simulating network scenarios within a centralised[9] Java implementation of the two algorithms. A typical scenario is composed of a fixed number of network provider domains (NPDs) and inter-domain links (IDLs), characterised by price and QoS parameters. Inside every domain a fixed number of network nodes are connected by intra-domain links. We estimated the time needed by the two different mechanisms to determine if none, one or many solutions exist (i.e., excluding the negotiation step). The negotiation aspects will play a major role in the distributed multi-agent version of NPI, which is currently under development. Figure 4 compares the time performance of the two mechanisms for the same simulated scenario, that contains 6 providers' networks and 16 inter-domain links randomly distributed between them. Case A and B differ for the set of demands that have to be allocated (see X-axis). For both mechanisms there are common constraints: same global abstract routes, i.e., same network domains to cross, and same QoS requirements. However, within FAP the search space is further reduced because of additional constraints, i.e., the fixed pre-selected access points. This means that if the failure probability, $P_f(P)$, along a global route P is small a conclusion is found faster with FAP than with DAC, (case A). With FAP in fact no arc-consistency takes place, so that there is a time gain. This remains valid as long as no other abstract paths are available. When the $P_f(P)$ increases, DAC performs better (case B). In fact, DAC expresses a set of possible solutions: alternative choices can be available along a pre-selected abstract path, although some access points may have been excluded by the arc consistency. The failure probability along P, $P_f(P)$, is determined by two main factors: the

[9] The resolution of the service demand allocation is performed by a unique agent, that knows about the other domains, so that is the only handling variables and constraints.

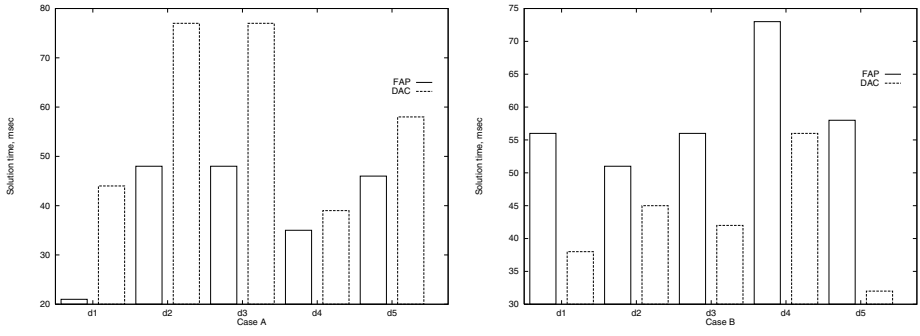

Fig. 4. DAC visualization.

difference d_{av} between the QoS available along P (inter-domain links) and qos_{req}, and the internal configuration of the network domains. The higher the d_{av} is, the more capable the networks along P are (i.e., the intra-domain routing is less likely to fail) and the smaller is P_f. The critical point is that it is not realistic to assume that a network operator is able in advance to exactly compute $P_f(P)$, since a knowledge about the internal topology of competitors domains would be required. However, another major element makes finally DAC more appealing than FAP. In many of the analysed cases, FAP terminates faster because the demand is rejected (no further backtracks are possible so that the algorithm terminates), while DAC is successfully going on for computing a solution through the arc consistency propagation.

In all studied cases DAC terminates in less than 0,1 msec. However, these results are dependent on the simulated scenarios. The crucial area for further work is getting more realistic data input and testing DAC more exhaustively within the distributed NPI architecture.

A distributed version of DAC, which is currently under test, will give a more precise estimation of the time needed for finding a solution to the IDA problem, following the NPI paradigm. The current simulations, with NPAs-Java agents distributed over different local hosts (SUN Sparc stations), terminate in few seconds (from 5 to 10 seconds without considering the negotiation). However, more work is required in order to give more stable and proper results.

6 Discussion

Currently many aspects of the interworking are statically fixed by contracts and many steps of the interaction are regulated by human operators by fax, e-mail etc. The NPI paradigm can be considered as a high level service which could be deployed in two different ways: in a short term period as a smart support for human operators, in a long term perspective as an autonomous system acting on behalf of humans and working at the connection management level. The first version would offer a valid support for human operators: it could compute local routes guaranteeing the qos_{req} and it could automatically verify which ones

are not consistent with the inter-domain constraints (which are for instance the routes fixed in the currently used contracts). In a future scenario NPI could supply an automated mechanism to route and negotiate the allocation of demands across distinct networks without the need for human intervention. This would require the integration of NPI in a real network infrastructure[10]. *Ad hoc* Mask Agents should be defined in order to exchange information with the network management plane. This corresponds to the creation of agents providing gateway functionalities for translating to/from the Agent Communication Language from/to, for example, CMIS/GMDO and IDL at the management system level.

We believe that one of the main strength of the NPI paradigm is the capability of supplying a rapid and efficient answer to the IDA process, without the need for different operators to reveal a substantial amount of internal information. The data to be exchanged concerns the set of possible access points that a network provider can use for a specific demand. We believe that is reasonable to assume that providers which need to interoperate must exchange a minimal amount of data, such as topology aggregated view.

Furthermore, the mechanism for achieving interconnectivity is valid independently on the specific underlying technology, i.e., it does not depend on how the QoS is provided, or on the specific networking mechanisms for reserving resources and for starting the connection. Such kind of paradigm is based on the idea that the complexity of protocols and technical details are hidden by the Mask Agents.

There are several reasons for the choice of agents. They enable abstractions from technical details, they recognise changes in the environment, they react to external events and they can create goals and plans. However, the real strength of agents is based on the community of multi-agent system and the negotiation mechanism and coordination facilities. A dynamic agent-based negotiation phase would allow to flexibly adapt prices and connectivity configuration to changes occurring inside and outside a single provider network. Automated and standardised information exchanges between software agents compared to the exchange of possibly ambiguous faxes, non punctual phone calls and sometimes unpredictable e-mails between humans, seems a valid argument for relying on software mediators.

7 Conclusion and future work

This paper has described a multi-agent paradigm to support the QoS-based inter-domain routing problem in a flexible and dynamic way without the need of human intervention. In particular the paper shows how the DCSP formalism provides a powerful and intuitive way of expressing the QoS-based inter-domain routing problem and a way of solving the problem which have been so expressed.

In our view, the increased flexibility of the NPI approach provides for new opportunities in the area of network inter-operability. In a future scenario NPI

[10] For instance telecommunications architectures such as TINA, IN gaiti, or more generically TMN [1] compliant environments.

agents could program the behaviour of network nodes (switches and/or routers) in order to automate the inter-domain routing process. Furthermore, agents could supply a flexible and dynamic negotiation framework.

We have built prototypes to test many of the concepts outlined in this paper and preliminary considerations enforce the use of DCSP techniques to support the inter-domain routing process. In order to validate the NPI paradigm we are continuing to test it by constructing more realistic scenarios and designing negotiation protocols around them, a problem that is complicated by the fact that the interworking process is itself in flux and no stable data is available.

References

1. I-ETSI 300 653. "Telecommunication Management Network (TMN). Generic managed object class library for the network level view", 1996.
2. "TINA Reference Points", The TINA Consortium, Version 3.1, June 1996.
3. B. Awerbuch, Y. Du, B. Khan and Y. Shavitt, "Routing Through Networks with Hierarchical Topology Aggregation". Technical Report, DIMACS, n. 98-16, 1998.
4. M. Calisti, C. Frei, B. Faltings, "A distributed approach for QoS-based multi-domain routing" AiDIN'99, *AAAI Workshop on Artificial Intelligence for Distributed Information Networking*, 1999.
5. T. Przygienda, O. Crochat, J. Y. Le Boudec, "A Path Selection Method in ATM using Pre-Computation". In: *Proceedings of IZS96*, February 1996.
6. A. Galis and C. Brianza and C. Leone and C. Salvatori "Towards Integrated Network Management for ATM and SDH Networks Supporting a Global Broadband Connectivity Management Service", Mullery A (Eds.)- Springer Verlag, Berlin, 1997, ISBN 3-540-63145-6.
7. D. Griffin, G. Pavlou, T. Tin, "Implementing TMN-like Management Services in a TINA Compliant Architecture: A Case Study on Resource Configuration Management". Lecture Notes in Computer Science, vol. 1238, 1997.
8. "Mobile Agent System Interoperability Facilities Specification", OMG TC Document orbos/97-10-05.
9. A. Galis, D. Griffin, "A comparison of approaches to multi domain connection management", available at the URL http://www.misa.ch/.
10. A. Hopson, R. Janson, "Deployment Scenarios for Interworking". TINA Consortium. TP-AJH.001-0.10-94 TINA-C, 19 January 1995.
11. Gacti D., Pujolle G., "TMN, IN, and DAI". In: *Proceedings TINA'93*, II/111-121, 1993, L'Aquila, Italy.
12. ACTS-MISA project documentation, http://www.misa.ch
13. M. Breugst, T. Magendanz, "Mobile-Agents- Enabling Technology for Active Intelligent Network Implementation". *IEEE Network Magazine, Special Issue on Active and Programmable Networks, May/June 1998, Vol. 12(3).*
14. V. Kumar. "Algorithms for Constraint satisfaction Problems: A Survey", *Appeared in AI Magazine 13(1): 32-44, 1992.*
15. Frei, C. and Faltings, B. "A dynamic hierarchy of intelligent agents for network management". In *Proceedings of Workshop on Artificial Intelligence in Distributed Information Networks IJCAI'97*, 1997.
16. M. Yokoo, E. Durfee, "Distributed Constraint Satisfaction for Formalising Distributed Problem Solving", *12th IEEE International Conference on Distributed Computing System '92, 614-621.*

17. FACTS project. http://www.labs.bt.com/profsoc/facts/
18. FIPA 97 ver. 2, Specifications, http://www.fipa.org/spec/fipa97.html
19. T. Magendanz, K. Rothermel, S. Krause, "Intelligent Age nts: An emerging Technology for the Next Generation Telecommunications?", INFOCOM'96.
20. P712 project: http://www.eurescom.de/Public/Projects/p700-series/P712-/P712.HTM
21. E. Tsang, "Foundations of Constraint Satisfaction", 1993, Academic Press, London, UK.