DIFFSERV NETWORK CONTROL USING A BEHAVIORAL MULTI-AGENT SYSTEM

Nada Meskaoui[1,3] nadames@inco.com.lb

Leila Merghem[1] leila.merghem@lip6.fr

Dominique Gaiti[1,2] dominique.gaiti@utt.fr

Karim Y. Kablan[43] kabalan@aub.edu.lb

1. LIP6-Université de Paris 6, 8 rue du Capitaine Scott - 75015 Paris, France,
2. LM2S: Université de Technologie de Troyes, 12 rue Marie Curie - 10010 Troyes Cedex, France.
3. American University of Beirut, Faculty of Engineering and Architecture, Lebanon.

Abstract: For many years, telecommunication networks have steadily grown in size and complexity due to the continually growing of users requirements for dynamicity, adaptability and security. The Internet will so support service convergence for this next network's generation. It also has to provide and guarantee Quality of Services to different types of services, where each service may get its own requirements. Providing and managing such networks is not an easy task. These different features may require more intelligent and dynamic behavior from network management. Existing approaches in network management do not support such a dynamic behavior. This paper presents a behavior-based multi-agent solution to manage dynamic telecommunication networks. The proposed solution is based on a Diffserv network by implementing a multi-agent system within Diffserv networks. The impact of an intelligent network management within Diffserv is also analyzed.

Key words: Agent-Based Systems and Designs, Multi-Agent Algorithms, Information Coordination and Collaboration Algorithms, Quality of Services in Information Systems.

The original version of this chapter was revised: The copyright line was incorrect. This has been corrected. The Erratum to this chapter is available at DOI: 10.1007/978-0-387-35703-4_21

D. Gaïti et al. (eds.), *Network Control and Engineering for QoS, Security and Mobility II*

1. INTRODUCTION

The usage of the Internet has increased enormously in the last few years. At the same time, new real-time as well as non-real-time applications like teleconferencing, remote seminars, and distributed simulation have emerged demanding for much improved service quality than the best effort actually proposed by the Internet. For this reason the complexity and dynamicity of telecommunication networks is continually growing, making their management and control more and more difficult. To adapt the network functioning to the requirements of these different types of applications, some solutions, like integrated services (Intserv) [1] with resources reservation protocols, such as RSVP (Resources reSerVation Protocols) [2] and differentiated services (Diffserv) [3], were proposed.

Multi-agent systems have proved their efficiency in different domains like air-traffic control, road traffic control, information retrieval/management and others. The objective of our work is so to prove that a multi-agent approach in network management is able to improve the quality of services provided to the different types of new applications transiting a Diffserv network.

This paper presents a simulation platform that integrates a multi-agent system within a Diffserv network. This simulator, that supports intelligence over Diffserv, is basically conceived to simulate intelligent traffic control and agents behaviors. So, the multi-agent system controls and adjusts the behavior of the Diffserv network nodes in order to increase the performance of Diffserv.

The remainder of this paper is organized as follows. In section 2, some new types of applications and their different needs of treatment are described, and some existent techniques that propose solutions to satisfy their requirements are outlined. Section 3 presents agents and multi-agent concepts and their applicability to our context. Then, the adopted node structure and the agents' behavior are described respectively in section 4 and 5 and finally the simulation platform and results are detailed in section 6. Conclusion is provided in section 7.

2. NEW APPLICATION TYPES AND TECHNIQUES

New types of applications have emerged in the last years, like real time applications and elastic applications that demand different qualities of service within the network nodes. Real time applications [4] are known by their sensibility to the variation in the delay introduced by the network to the delivered packets. That is why they need some characterization of the

maximum delay their packets will experience. As some real time applications are more sensitive to the delay variation than other real time applications, we can so identify the intolerant and tolerant real time applications that demand respectively guaranteed and predictive services as described in [1]. As real-time applications [4] do not wait for late data arrival, elastic applications will always wait for data to arrive [1]. It is not that these applications are insensitive to delay; to the contrary, significantly increasing the delay of a packet will often harm the application's performance.

The market was pushing in the last years for immediate deployment of QoS solutions that addresses the needs of the Internet as well as enterprise networks. This push has led to the development of a number of architectures such as Intserv/ RSVP, Diffserv, and others.

The integrated Services (Intserv) propose an extension to the original architecture of the Internet [1]. In this architecture, an IS router should also implement a reservation setup protocol known as RSVP (resource reservation setup protocol) [2].

Differentiated services (Diffserv) [3] are a set of technologies that allow network service providers to offer services with different kinds of network quality-of-service (QoS) objectives to different customers. This is achieved by defining three service levels: Expedited Forwarding "EF", Assured Forwarding "AF", and Proportional Forwarding "PF". In this architecture, Diffserv networks classify packets based on the setting of bits in the TOS field of each packet's IP header and eliminate the reliance on per-flow state. Diffserv quality of services can initially be deployed using top-down provisioning, with no requirement for end-to-end signaling [5]. The implementation of Diffserv implies the extension of the network nodes. A differentiated services-compliant network node includes a number of functional elements, namely packet classifiers, traffic conditioners and per-hop forwarding behaviors (PHB).

Intserv/RSVP [6] and Diffserv [3] have both succeeded in providing different levels of treatments for the different types of applications. The basic problem of Intserv/RSVP is the number of the managed states, which increase linearly with the number of flows. These limitations impede the deployment of Intserv/RSVP in the Internet at large. For this reason, the Diffserv technology is chosen to build on our proposed solution.

3. AGENT AND MULTI-AGENT APPROACH

Agents and multi-agents systems (MAS) are two innovative and interesting concepts for a great number of researchers in different domains

like road traffic control, biologic and social phenomena simulations, industrial applications, and others. These agents are characterized by their ability to interact and their knowledge. MAS may be defined as "a loosely-coupled network of problem solvers that work together to solve that problems that are beyond their individual capabilities" [10]. These problem solvers are agents and are defined as autonomous entities able of acting in their environment. They are able to communicate directly with other agents, posses their own resources, perceive their environment (but to a limited extent), have only a partial representation of their environment (and perhaps none at all), and have a behavior [8]. MAS are well used in the discovery of topology in a dynamic network by mobile agents [11,12], the optimization of routing process in a satellite constellation [13], the maximization of channel allocation in a cellular network [14], and others. Existing networking solutions using agents, human assistance are needed to deal with some aspects of network management. Our objectives are to implement intelligent agents that can replace most of the human interventions and so reduce the human's assistance to the minimum and improve the performance of the network by its intelligent behavior.

Different types of agents can be defined like reactive, cognitive, hybrid, mobile, and others. Each of these types is characterized by some properties like autonomy, pro-activeness, collaboration, etc. In our work, we are interested in the reactive and cognitive agents types.

A reactive agent is one whose actions are based entirely on changes in its environment. It continually monitors its situation using sensors and exploits the data collected from the sensors to decide its next action. Cognitive agents or smart agents possess knowledge, have the ability to reason about this knowledge, and can learn from the past. Using all of this information, cognitive agents can calculate a plan of actions that it believes will most efficiently achieve its goals.

We choose to implement in our Diffserv network some hybrid reactive agents and cognitive agents to benefit from the simplicity of reactive agents and intelligence of cognitive agents. To interface these two philosophies, the selection strategy proposed by Chatley [7] is used. In this strategy, the deliberative aspect of the agent collects the required data and uses it to compute plans to prevent congestion by initiating actions and modifying some parameters. The reactive aspect will so get the most appropriate plan to execute until the problem is solved. We also consider that the two concepts should not face conflict situations because the agent is cognitive until congested situation is reached. It will be then reactive and executes the pre-planned actions by the cognitive phase.

4. THE NODE MODEL

We consider that the Diffserv techniques introduce a certain level of intelligence within the network nodes as it adds to the node the ability to differentiate between the different types of traffics and to choose for each type the right treatment. In the proposed node model, the MAS is implemented to manage and treat the incoming traffic in a more intelligent way in order to increase the performance of our Diffserv network and perhaps offer to providers the possibility to accept more qualified traffics without modifying their network configuration.

We propose a Diffserv node structured in two layers: The first layer implements the Diffserv components and the second one implements the agent as seen in Figure 1.

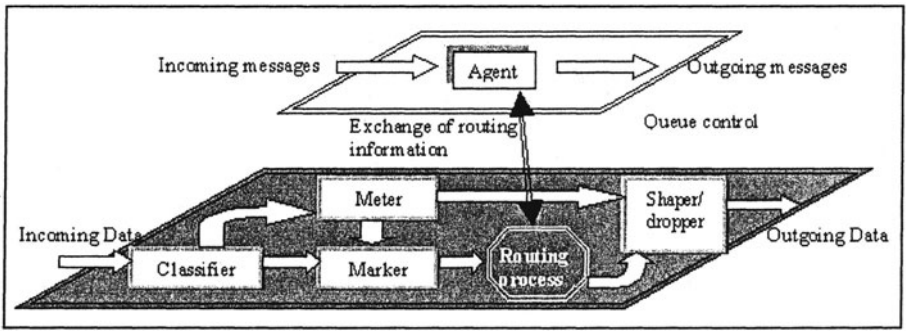

Figure 1. Structure of the ingress/ egress routers

Agents may participate in different functionalities of the network nodes like for example modifying the routing decision of some packets regarding the network load. This means, if the agent detects congestion (by controlling the evolution of the output queues) in one of the node outputs, it takes decision to modify some parameters that helps in resolving the congestion.
In each Diffserv node, an agent is implemented. All our implemented agents are considered to be correct, unfaithful, and inequitable. A correct agent does not send any incorrect information to its neighbors. Unfaithful agents choose between different outputs to route packets and inequitable agents treat traffic in different priority levels. All these agents are compatible and cooperate to achieve their common goal. In our multi-agent system, we can distinguish three levels of cooperation, as shown in Figure 2.

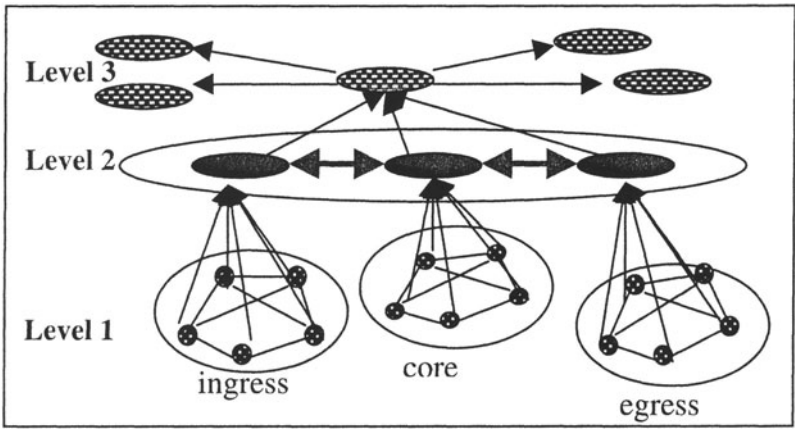

Figure 2. Three level organization of the multi-agent system

In level 1, we can find ingress, egress, and core nodes and cooperation is shown between the nodes related to the same type. Level 2 shows the cooperation between the different types of agents. In level 3, the cooperation between the different Diffserv domains is described.

5. THE AGENT'S BEHAVIOR

We adopt in our proposal a very simple agent's behavior that proposes a modification of the routing table according to the network state. This behavior doesn't react on any Diffserv element of the node; it only modifies the routing process.

The agent reacts if it detects congestion in one of its output links (careful behavior) or if it receives a message from a congested neighbor (unfaithful behavior).

The careful behavior, shown in Figure 3, possesses two rules (CAREFUL_RULE1, and CAREFUL_RULE2) triggered by two different states (S0 and S1). State "S0" means that no congestion is detected yet while the state"S1" means that congestion in one of the output links is detected.

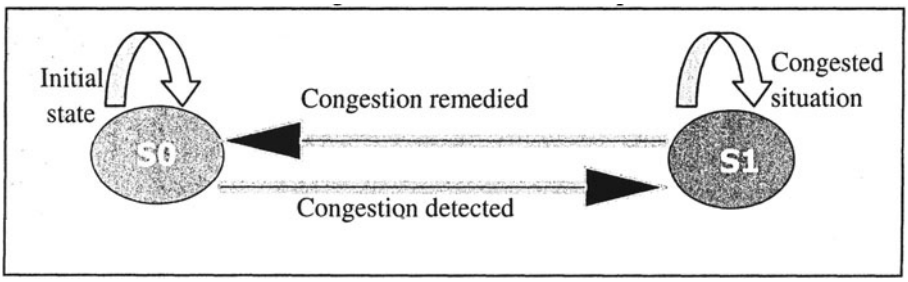

Figure 3. Agent reaction after congestion detection

CAREFUL_RULE1: If "S1" is reached, then

react and send a message M1" to the predecessors to inform them about the current state.

CAREFUL_RULE2: If "S0" is reached again, then

send a message "M2" to predecessors informing them that the congestion is remedied.

The unfaithful behavior has also two rules (UNFAITHFUL_RULE1, and UNFAITHFUL_RULE2) depending on the two states (R0 and R1). It reacts, as described in Figure 4, after the reception of a message informing him that one of the neighbors is congested. When the agent receives the message "M1" from a peer, it modifies its state to R1, and when it receives "M2", the agent state will come back to its initial state R0.

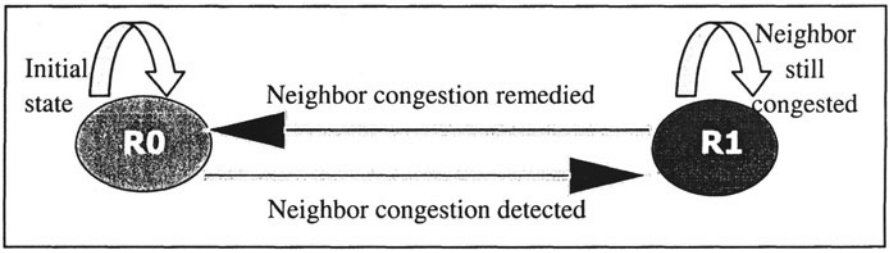

Figure 4. Agent reaction after neighbor congestion detection

FAITHFUL_RULE1: If "R1" is reached, then

react to resolve the congestion of one of the neighbors, by modifying its routing algorithm to re-direct some traffic to a non-congested output.

FAITHFUL_RULE2: If "R0" is reached again, then

re-initialize the parameters. These parameters are the ones modified after the detection of congestion in one of the neighbor's nodes.

Our agents are helpful; they react to the neighbor's congestion by avoiding sending him packets until it comes through this congestion. In other terms they adapt their behavior to the changing situations. They are also cooperative in the sense that they possess rules that take account of the neighbors' state in the process of behaviors' selection.

6. SIMULATION AND RESULTS

The proposed model is next simulated. As our proposed architecture combines networking features and multi-agent system, the simulator should be able to treat these two concepts. For this reason, we choose to realize an extension to the Javasim simulator [9]. This extension implements the agent as a component able to cooperate via the network with other peers and to

exchange information with other components within the node. Javasim [9] is a component-based, compositional simulation environment. It has been built upon the notion of the autonomous component architecture. The basic entity in Javasim is components, but unlike the other component-based software packages/standards, components in Javasim are autonomous. The behavior of Javasim components is defined in terms of contracts and can be individually designed, implemented, tested, and incrementally deployed in a software system. Javasim has been developed entirely in Java. This, coupled with the autonomous component architecture, makes Javasim a truly platform-neutral, extensible, and reusable environment.

This component architecture makes it possible to compose different simulation scenarios for different network architectures from a set of base components. To define/implement such a set, the INET abstract network model that provides a set of standard network components was implemented [9].

A Diffserv package [9] is built on top of Javasim/INET model. It implements the various classes that may serve as the building blocks for Diffserv. Using this Diffserv package, we can construct our Diffserv network and then we still have to define and build our intelligent agent package on top of the INET model. We define an agent package on top of the Javasim/INET node structure. This agent package is defined as a component (see Figure 5) having the capability to interact with other components within the node and with other peer agents implemented within the other nodes of the Diffserv domain. This agent component has also intelligent capabilities. A set of rules is defined and invoked according to the node and network's states.

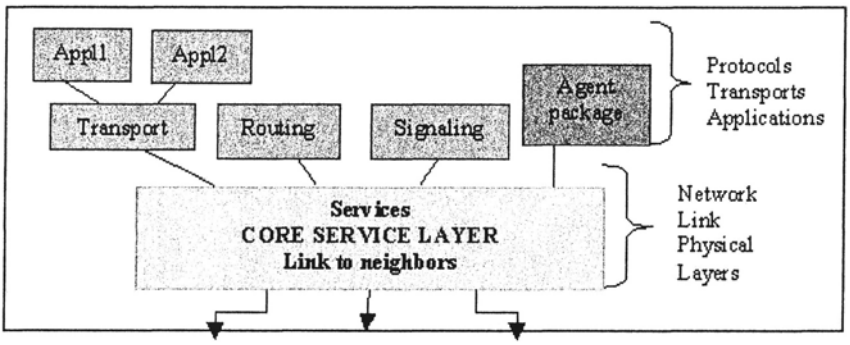

Figure5. Agents and Diffserv integration within Javasim/INET node structure

The agent is able to invoke the appropriate rules according to the node and the environment state. Indeed, the agents optimize the networks parameters (delay, jitter, loss percentage of a class of traffic, etc.), by

adopting the most suitable behaviors following the received traffic' nature and the neighbors nodes' state.

In the following simulations we use a Diffserv network that implements the "DV", distance vector routing algorithm. We adopt the topology described in Figure 6. This topology is composed of three sub-networks N11 (sub-network containing source Hosts H1 and H3), N12 (Diffserv transit network) and N13 (sub-network containing destination Hosts H6 and H7):

(N0, N2) and (N4, N5) are the gateways of respectively the two sub-networks N11 and N13 to access the Diffserv transit network N12. N8 and N9 are the core routers. The cost of each link is marked on the line connecting every two nodes. The agent modifies - according to the state of the successor nodes - the cost of each output link. This cost parameter helps the agent in determining the best less congested route to destination.

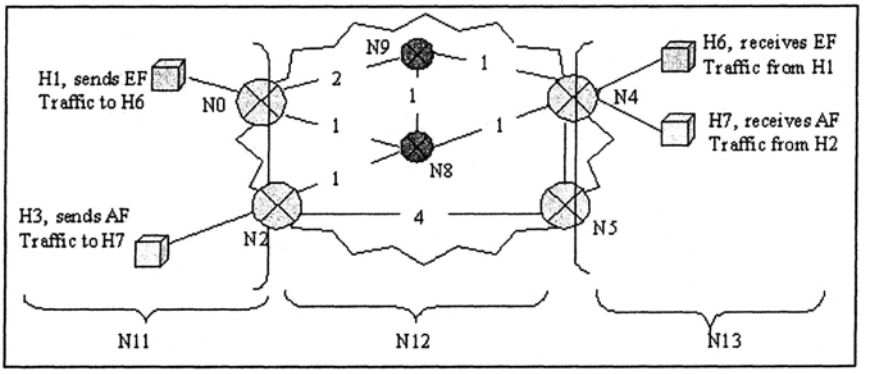

Figure 6. The simulated topology

H1 sends random EF traffic to H6 via the Diffserv network. H3 sends random AF traffic to H7 via the Diffserv network. Routers are interconnected by a 6 Mbps link; only the link between N8 and N4 is about 3 Mbps. This bottleneck on the links N8-N4 was conceived to cause congestion. The propagation delay between these links is set to 0.5s. Packets send by H1 (EF traffic) and H3 (AF traffic) are about 1000 bytes of length. H1 sends EF packets every 0.015s and H3 sends AF packets every 0.01s.

Figures 7 and 8 show the throughput of the EF and AF traffic for duration of 150s. In these figures, simulation shows that before agent's implementation, we have great drops of AF packets – represented by the difference between the blue (— · —) and green (— —) lines (see Figure 7) and after the agents implementation within the Diffserv nodes, results shows a great increase in the AF traffic throughput (see Figure 8).

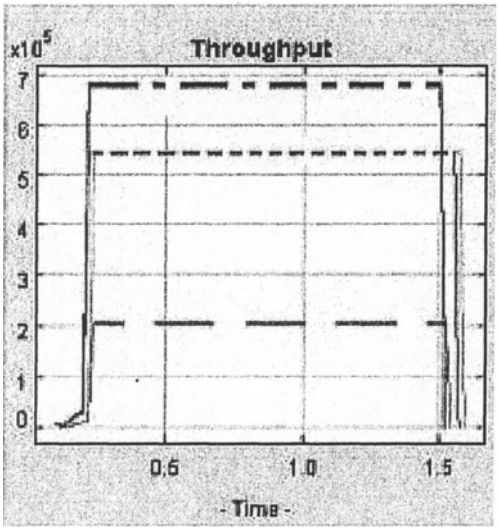

Figure 7. Throughput before agent's integration

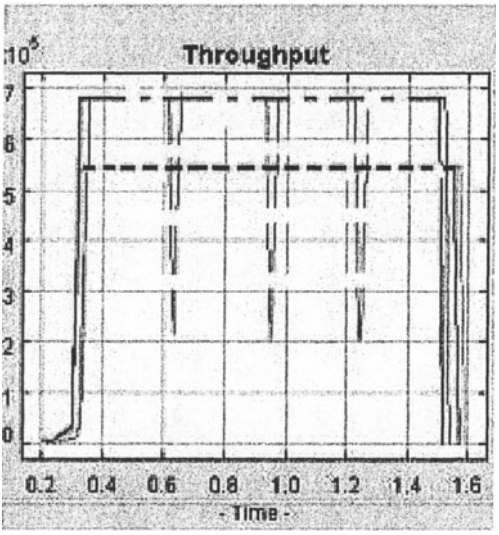

Figure 8. Throughput after agent's integration

The red line (– – –) is the amount of EF traffic sent by the EF source, the blue line (— · —) shows the amount of AF traffic sent by the AF source, the yellow line (– – –) is the amount of EF traffic received by the EF destination and the green line (— —) represents the amount of AF traffic received by the AF destination.

Simulation results of Figures 9 and 10 shows the difference of AF packets drop before and after agents integration. In these two simulations the topology is the same as the one described in figure 6, only the traffic shape was changed: H1 sends random on-off EF traffic to H6 via the Diffserv network. H3 sends random AF traffic to H7 via the Diffserv network. Routers are interconnected by a 6 Mbps link; only the link between N8 and N4 is about 3 Mbps; and the propagation delay between these links is set to 0.5s. Packets send by H3 (AF traffic) are about 1000 bytes of length sent every 0.012s. EF packets are about 1000 bytes of length, having a rate of 200 kbps over the on period. The on and off periods are about 100s.

The red line (– – –) is the amount of EF traffic sent by the EF source, the blue line (— · —) shows the amount of AF traffic sent by the AF source, the yellow (– – –) line is the amount of EF traffic received by the EF destination and the green line (— —) represents the amount of AF traffic received by the AF destination.

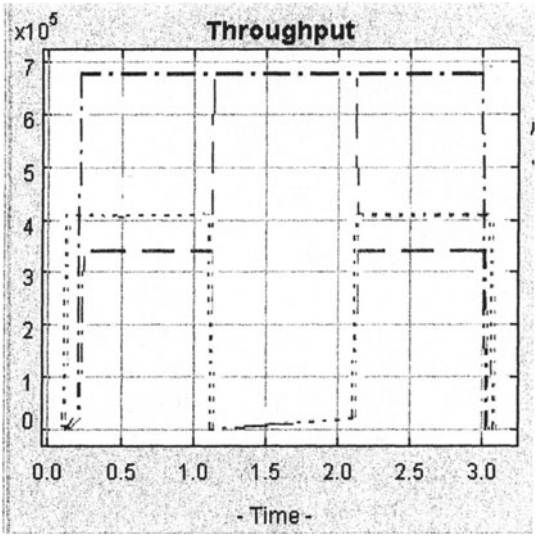

Figure 9. Throughput before agent's integration

Simulation shows that without agents integration some AF packets are dropped during the EF "on" period. During the EF "off" period all the AF traffic is received by the destination. After agent's integration, the amount of AF packets received by the destination is really greater than before the agent's implementation. Only few AF packets are dropped during the agent reaction phase.

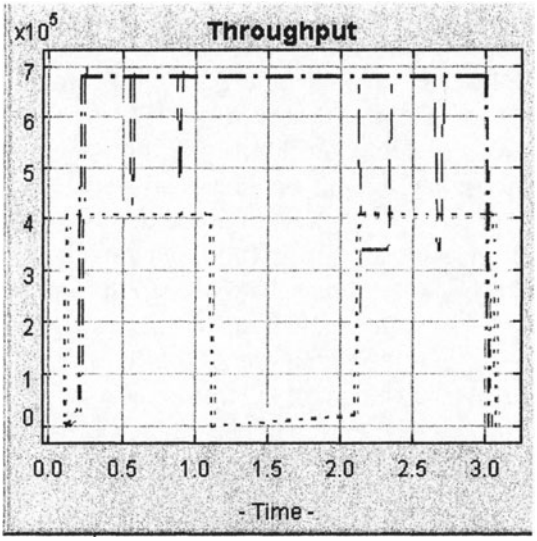

Figure 10. Throughput after agent's integration

7. CONCLUSION

This paper presents a new approach to satisfy the requirements of users and providers in terms of quality of services and dynamicity. This approach is based on the following concept: multi-agent systems for network control. A new simulation platform was proposed to integrate intelligent multi-agent systems within a Diffserv network. The platform and the agent behavior were tested and validated by simulations. It was shown that agents minimize packets loss due to the use of simple behaviors.

In the future, the accuracy and efficiency of the proposed method along with its limitations should be tested within a real network.

REFERENCES

[1] R. Braden – ISI, D. Clark – MIT, S. Shenker - Xerox PARC, "Integrated Services in the Internet Architecture: an Overview", RFC1633, June 1994.

[2] S. Berson- ISI, S. Herzog - IBM Research, S. Jamin - Univ. of Michigan, "Resource ReSerVation Protocol (RSVP) -- Version 1 Functional Specification", RFC2205, September 1997.

[3] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, W. Weiss, "Architecture for Differentiated Services", RFC2475, December 1998.

[4] Schulzrinne, H., Casner, S., Frederick, R., and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", RFC 1889, January 1996.

[5] Y. Bernet, R. Yavatkar, P. Ford, F. Baker,L. Zhang, K. Nichols, M. Speer, "A Framework for Use of RSVP with Diffserv Networks", November, 1998.

[6] J. Wroclawski - MIT LCS, "The Use of RSVP with IETF Integrated Services", RFC2210, September 1997.

[7] *www.iis.ee.ic.ac.uk/~frank/surp99/article2/rbc97*

[8] J. Ferber, "Multi-Agent System: An Introduction to Distributed Artificial Intelligence", Harlow: Addison Wesley Longman, 1999.

[9] *www.javasim.org*

[10] E. H. Durfee, V. Lesser and D. D. Corkill, "Trends in cooperative distributed problem solving" IEEE transaction on knowledge and Data Engineering, vol. KDE-1, pp. 63-83, Mar 1989.

[11] R. Roychoudhuri, S. Bandyopadhyay and K. Paul, "Topology discovery up ad-hoc wireless networks using mobile agents", in Proc. Of 2nd. Intl. Workshop MATA'2000, (Paris, France), pp. 1-15, Sep 2000.

[12] N. Minar, K. H. Kramer, and P. Maes, Software Agents for Future Communication Systems, ch. Cooperating Mobile Agents for Dynamic Network Routing. Springer Verlag, 1999.

[13] E. Sigel, B. Denby and S. L. Hegarat-Mascale, "Application of ant colony optimization to adaptive routing in LEO telecommunication satellite network". Annals of Telecommunications, vol. 57, pp. 520-39, May-Jun 2002.

[14] E. L. Bodanese and L. G. Cuthbert, "A multi-agent channel allocation scheme for cellular mobile networks", in Proc. Of ICMAS'2000, (Boston, USA), pp. 63-70, Jul. 2000.