VNU Journal of Science, Natural Sciences and Technology 24 (2008) 147-161

Learning approaches to support dynamics in communication networks

Abdelhamid Mellouk[1,*], Saïd Hoceïni[1], Saida Ziane[1], Malika Bourennane[2]

[1]*LISSI/SCTIC Laboratory, IUT Creteil/Vitry*
University Paris XII, France. 122, rue Paul Armangot, 94400 Vitry sur Seine, France
[2]*Department of Computer Science, University Es Senia, Algeria*

Received 31 October 2007

Abstract, In the context of modern high-speed communication networks, decision reactivity is often complicated by the notion of guaranteed Quality of Service (QoS), which can either be related to time, packet loss or bandwidth requirements: constraints related to various types of QoS make some algorithms not acceptable. Due to emerging real-time and multimedia applications, efficient routing of information packets in dynamically changing communication network requires that as the load levels, traffic patterns and topology of the network change, the decision policy also adapts. We focused in this paper on QoS based mechanisms by developing a neuro-dynamic programming to construct dynamic state-dependent policies. In this paper, we present an accurate description of the current state- of-the-art and give an overview of our work in the use of reinforcement learning concepts focused on communications networks. We focus our attention by developing a system based on this paradigm and study the use of reinforcement learning approaches in three different communication networking domains: wired networks, mobile ad hoc networks, and packet router's scheduling networks.

Keywords: Self-Depedent Mechanism Decision, Quality of Service based Routing, Multi Path Routing. Dynamic Networks, Reinforcement Learning, Adaptive Scheduling.

1. Introduction

Today, providing a good quality of service (QoS) in irregular traffic networks is an important challenge. Besides, the impressive emergence and the important demand of the rising generation of real-time Multi-service (such as Data, Voice VoD, Video-Conference, etc.) over communication heterogeneous networks, require scalability while considering a continuous QoS. This emergence of rising generation Internet required intensive studies these last years which were based on QoS routing for heterogeneous networks on the one hand and on the backbone architecture level of communication networks characterized by a high and irregular traffic on the other hand [1].

The basic function of QoS routing is to find a network path which satisfies the given constraints and optimize the resource utilization. The integration of QoS parameters increases the complexity of the used routing

* Corresponding author. E-mail: mellouk@univ-paris12.fr

algorithms. Thus, the problem of determining a QoS route that satisfies two or more path constraints (for example, delay and cost) is known to be NP- complete [2]. A difficulty is that the time required to solve the Multi-Constrained Optimal path problem exactly cannot be upper-bounded by a polynomial function. Hence the focus has been on the development of pseudo-polynomial time algorithms, heuristics and approximation algorithms for multi- constrained QoS paths [3].

At present, several studies have been conducted on QoS routing algorithms which integrate the QoS requirements problematic for the routing algorithm. [4] introduce heuristics to find a source-to-destination path that satisfies two or more additive constraints on edge weights. [5] has proposed a polynomial time approximation algorithm for k multi-constrained path which uses a shortest path algorithm such as Dijkstra's [6,7] propose a randomized heuristic that employs two phases. In the first one, a shortest path is computed for each of the k QoS constraints as well as for a linear combination of all k constraints. The second phase performs a randomized breadth-first search for a solution of k multi-constrained problem. In [3], authors suggest that QoS routing in realistic networks could not be NP-complete regarding to a particular class of networks (topology and link weight structure).

Due this complexity, QoS routing problems are divided on several classes according to some aspects. For example, we distinguish the single path routing problem and the multipath routing problem, where routers maintain multiple distinct paths of arbitrary costs between a source and a destination. The Multipath routing offers several advantages like good bandwidth, bounding delay variation, minimizing delay, and improved fault tolerance. So, it makes an effective use of the graph structure on a network, as opposed to single path routing which superimposes a logical routing tree upon the network topology. We find in literature many and various approaches that have been proposed to take into account the QoS requirement. The reader can refer to [8] for an overview.

Constraints imposed by QoS requirements, such as bandwidth, delay, or loss, are referred to as QoS constraints, and the associated routing is referred to as QoS routing which is a part of Constrained-Based Routing (CBR). Interest in constrained-based routing has been steadily growing in the Networks. Based on heuristics used in all of these approaches to reduce their complexity, we can classified it in three main categories:

Label Switching/Reservation Approaches- spurred by approaches like ATM PNNI, MPLS or GMPLS. With MPLS, fixed length labels are attached to packets at an ingress router, and forwarding decisions are based on these labels in the interior routers of the label-switched path. MPLS Traffic Engineering allows overriding the default routing protocol, thus forwarding over paths not normally considered. A resource reservation protocol such as RSVP must be employed to reserve the required resources. Another Architecture proposed for providing Internet QoS is the Differentiated Services architecture. Diffserv scales well by pushing complexity to network domain boundaries.

Multi-Constrained Path Approaches (MCP) - The goal of all of these approaches is to retrieve the shortest path among the set of feasible paths between two nodes. Considerable work in the literature has focused on a special case of the MCP problem known as the Restricted Shortest Path (RSP) problem. The goal is to find the least-cost path among those

that satisfy only one constraint. An overview of these approaches can be found in [9].

Inductive approaches- To be able to make an optimal routing decision, according to relevant performance criteria, a network node requires to have a complete knowledge of the entire network state and an accurate prediction of the evolution of the networks and its dynamics. This, however, is impossible unless the routing algorithm is capable of adapting to the network state changes in almost real time. Thus, it is necessary to design intelligent and adaptive optimizing routing algorithms which take into account the network state and its evolution. We need to talk about QoS based state dependent routing algorithm.

In this contribution, we present an accurate description of the current state-of-the-art and give an overview of our work in the use of reinforcement learning concepts focused on communication networks. We focus our attention by developing a system based on this paradigm called KOCRA for K Optimal Constrained path Routing Algorithm. Basically, these inductive approaches selects routes based on flow QoS requirements and network resource availability. After developing in section 2 the concept of routing in high speed networks, we present in section 3 the family of inductive approaches. After, we present our works based on reinforcement learning approaches in three different communication networking domains: wired networks, mobile ad hoc networks, and packet router's scheduling networks. Last section concludes and gives some perspectives of this work.

2. Routing problem

As Internet is a large collection of more than 25,000 independent domains called autonomous systems (Ases), the cooperation between ASes is not optimized at the network level, but rather it is based on the business relationships between organizations. The fully-independent management actions in each AS are expressed in terms of a policy-based routing strategy which primarily controls the outbound traffic of an AS and can include conflicting policies. A global solution for QoS routing over all the ASes must be able to handle both the differing QoS provisioning mechanisms and service specifications. This latter solution of building models of large ISP's is so complex to obtain [10]. For this, Routing is divided onto two classes: IGP and EGP. IGP, such as OSPF or IS-IS, compute the interior paths in one AS, while EGP, such as BGP, is responsible for the selection of the inter-domain paths. To fulfill application QoS requirements, many ISPs have deployed mechanisms to provide differentiated services in their networks. In fact, in the last decade, the development of none of QoS routing proposals has turned out to be sufficiently appealing to become deployed in practice. This is because ISPs have preferred to overprovision their networks rather than deliver and manage QoS [11].

In the IGP or EGP cases, a routing algorithm is based on the hop-by-hop shortest-path paradigm. The source of a packet specifies the address of the destination, and each router along the route forwards the packet to a neighbor located "closest" to the destination. The best optimal path is chosen according to given criteria. When the network is heavily loaded, some of the routers introduce an excessive delay while others are under-utilized. In some cases, this non-optimized usage of the network resources may introduce not only excessive delays but also high packet loss rate. Among routing algorithms extensively employed in the same AS routers, one can note: distance vector algorithm such as RIP and the link state algorithm such as OSPF or IS-IS [12].

3. Inductive approaches

Modern communication networks is becoming a large complex distributed system composed by higher interoperating complex sub-systems based on several dynamic parameters. The drivers of this growth have included changes in technology and changes in regulation. In this context, the famous methodology approach that allows us to formulate this problem is dynamic programming which, however, is very complex to be solved exactly. The most popular formulation of the optimal distributed routing problem in a data network is based on a multi-commodity flow optimization whereby a separable objective function is minimized with respect to the types of flow subject to multi-commodity flow constraints [13], [14]. In order to design adaptive algorithms for dynamic networks routing problems, many of works are largely oriented and based on the Reinforcement Learning (RL) notion [15]. The salient feature of RL algorithms is the nature of their routing table entries which are probabilistic. In such algorithms, to improve the routing decision quality, a router tries out different links to see if they produce good routes. This mode of operation is called exploration. Information learnt during this exploration phase is used to take future decisions. This mode of operation is called exploitation. Both exploration and exploitation phases are necessary for effective routing and the choice of the outgoing interface is the action taken by the router. In RL algorithms, those learning and evaluation modes are assumed to happen continually. Note that, the RL algorithms assigns credit to actions based on reinforcement from the environment. In the case where such credit assignment is conducted systematically over large number of routing decisions, so that all actions have been sufficiently explored, RL algorithms converge to solve stochastic shortest path routing problems. Finally, algorithms for RL are distributed algorithms that take into account the dynamics of the network where initially no model of the network dynamics is assumed to be given. Then, the RL algorithm has to sample, estimate and build the model of pertinent aspects of the environment.

Many of works has done to investigate the use of inductive approaches based on artificial neuronal intelligence together with biologically inspired techniques such as reinforcement learning and genetic algorithms, to control network behavior in real-time so as to provide users with the QoS that they request, and to improve network provide robustness and resilience [16-18].

4. KOCRA system based reinforcement learning in routing wired networks

Our system, called “K Optimal Constrained path Routing Algorithm (KOCRA)”, contains three stages. The objective of the first stage is to select the K Best candidate paths according to the cost cumulative path from the source and the destination nodes (for simplicity, we consider here all link costs equal to 1). The second stage is used to integrate the dynamics of traffic. For this, a continuous end-to-end delay among the K Best selected Paths is computed using a reinforcement Q- learning function. In order to force the router to take the alternative routes regarding to the second stage, we used a third one which compute automatically a probability affected to each path based on packet delivery time obtained by the second stage and the time latency in queuing file associated for each path.

4.1. First stage: constructing K-best paths

First of all, in spite of exploring the entire network environment which needs large computational time and space memory, our approach reduces this environment to K best no loop paths in terms of cost cumulative links. Thus, each router maintains a link state database as map of the network topology. We used a label setting algorithm based on the optimality principle and being a generalization of Dijkstra's algorithm [6]. In order to find these K best paths, a variant of Dijkstra's algorithm proposed in [19] was used. By using a pertinent data structure, the space complexity is *O(Kmn)*, where K is the number of paths, *m* (resp. *n*) is the number of edges (resp. the number of links). The time complexity can be kept at *O(knlog(kn)+k2mn)* [27]. When a network link changes its state (i.e., goes up or down, or its utilization is increased or decreased), the network is flooded with a link state advertisement (LSA) message. This message can be issued periodically or when the actual link state change exceeds a certain relative or absolute threshold. Obviously, there is tradeoff between the frequency of state updates (the accuracy of the link state database) and the cost of performing those updates. In our approach, the link state information is updated when the actual link state change. Once the link state database at each router is updated, the router computes the K optimal paths.

4.2. Second stage: Q-learning lgorithm for optimizing the end-to-end delay

After finding our K best Optimal Paths based on link costs, the second step is to distribute the traffic on these K candidate paths. For this, we use another criteria based on the end-to-end delay. The reinforcement signal which is chosen corresponds to the estimated time to transfer a packet to its destination. This value is computed by a variant of Q-Routing algorithm which is considered as an asynchronous relaxation of the Bellman-Ford algorithm used in distance vector protocols. Typically, the packet delivery time includes three variables: the packet transmission time, the packet treatment time in the router and the latency in the waiting queue. In our case, the packet transmission time is not taken into account. In fact, this parameter can be neglected in comparison to the other ones and has no effect on the routing process.

In this approach, each router *x* maintains in a Q-table a collection of values of *Q(x, y, d)* for every destination *d* and for every interface *y*. This value reflects a delay of delivering a packet for destination *d* via interface *s*. Then, the router *x* forwards the packet to the best next router *y* determined from the Q-table. Just after receiving this packet, the router *y* provides *x* an estimate of its best Q value to reach the destination. This new information is then added in the Q- values of the router *x*.

The reinforcement signal *T* employed in the Q-learning algorithm can be defined as the minimum of the sum of the estimated *Q (x, y, d)* sent by the router *y* neighbor of router *x* and the latency in waiting queue q_x corresponding to router *x*.

$$T = \min_{y \in \text{neighbor of x}} \{q_x + Q(x,y,d)\} \quad (1)$$

Where *Q(x, y, d)*, denote the estimated time by the router x so that the packet *p* reaches its destination *d* through the router *y*. This parameter does not include the latency in the waiting queue of the router *x*. The packet is sent to the router *y* which determines the optimal path to send this packet.

Once the choice of the next router is made, the router y puts the packet in the waiting queue, and sends back the value *T* as a

reinforcement signal to the router *x*. It can therefore update its reinforcement function as:

$$\Delta Q(x, y, d) = \eta(a + T - Q(x, y, d)) \quad (2)$$

α and η are the packet transmission time between *x* and *y* and the learning rate respectively.

So, the new estimation $Q'(x, y, d)$ can be written as follows:

$$Q'(x, y,d) = Q(x, y,d)\ (1-\eta)+\eta(T + a) \quad (3)$$

4.3. Third stage: adaptive probabilistic path selection

The goal of this stage is to distribute the traffic on K best paths in probabilistic manner. To force the router to take alternative routes find in K best paths and not only the best one, we compute a probability affected to each path automatically. In this manner, the flow packets reach their destination with a time close to optimal, while ensuring a good exploration of the remaining paths. The process is based on the packet delivery time computed by our Q reinforcement learning and the latency in queuing file associated for each path.

Let $D_i\ (t)$ be the packet delivery time for path *i* at time *t*. Let $T_i^{n'}(t)$ be the latency in queuing file associated to closest router *n'* in the direction of path *i* at time *t* (that is, the neighbor of router *n*). The following formula allows us to count the probability $P_i^n(t)$ for the i^{th} path in router *n* at time *t*:

$$P_i^n = \left[\left(\frac{1}{D_i}\right)^{\alpha} * \left(\frac{1}{T_i^{n'}}\right)^{\beta}\right] \Big/ \left[\sum_{i=1}^{K}\left(\frac{1}{D_i}\right)^{\alpha} * \left(\frac{1}{T_i^{n'}}\right)^{\beta}\right] \quad (4)$$

Where α and β are two tuneable parameters that determine respectively the influence of delay time and waited queue time. They have an equivalent influence in the case of $a = \beta$. This formula associates a very small probability for paths with high delay time and/or high queue time. This is due to the fact that when delay time (respectively waited time) increase the value of $\left[1/D_i(t)\right]^{\alpha}$ respectively $\left[1/T_i(t)\right]^{\beta}$ decreases.

4.4. Performance evaluation

To validate our results in the case of irregular traffic in wired networks, we take the results given by a well-known Djikstra's algorithm (which offers to use an existing polynomial-time path computation) used in protocols such OSPF, IS-IS or CISCO EIGRP as a reference for our study. This choice of this classical approach is argued by the fact that the majority of ISP's used actually this kind of protocols to exchange routing information in their networks. In order to do comparison with KOCRA, parameters of standard approach used here are fixed in order to optimize the delay and cost criteria simultaneously (on the rest of paper, we used the notation "Standard Optimal Multi-Path Routing Algorithm (SOMRA)" for this kind of algorithm). All algorithms have been implemented with OPNET and used the same data structure. OPNET software constitutes for telecommunications networks an appropriate modeling, scheduling and simulation tool. It allows the visualization of a physical topology of a local, metropolitan, distant or on board network. The protocol specification language is based on a formal description of a finite state automaton.

The simulations presented in this article consisted of creating a traffic merged in irregular network topology, through which the two families of algorithms (KOCRA and SOMRA) computed the best paths between two nodes. QoS measures of each of tested algorithms concerns two additive constraints: cost and delay criteria. Results given in all the figures are evaluated in terms of average packet

end-to-end delivery time on both topologies. Time simulation is represented on the other axis of the figures.

1) Simulation parameters on the irregular topology

The topology of the network is specified by a collection of routers and a set of links that bind these routers elements. The network traffic is specified in the source router by setting several parameters like: the start time, the stop time, the statistical distribution for packet inter-arrival times, the statistical distribution for packet size and the destination node.

To ensure a meaningful validation of our algorithm performance, we devised a realistic simulation environment in terms of network characteristics, communications protocols and traffic patterns. We focus on IP datagram networks with irregular topology. The topology of the network employed for simulations includes 36 interconnected nodes with essentially two parts of the network, as shown in Fig. 1. This topology is the same used in [17] for their Q learning approach.

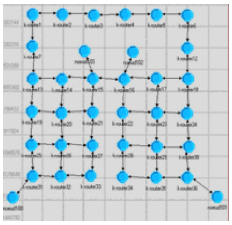

Fig. 1. Network topology.

The traffic is sent/received by four end nodes (marked in the figure noeud100, noeud101, noeud102 and noeud103).

We model traffic in terms of requests characterized by its source and destination. While we concern ourselves with arrival and departure of flows, we do not model the data traffic of the flows. For simplicity, we also chose not to implement a proper management of error, flow and congestion control. In act, each additional control component has a considerable impact on the network performance, making very difficult to evaluate and to study properties of each control algorithm without taking in consideration the complex way it interacts with all the other control components [18]. Therefore, we chose to test the behavior of our algorithm such that the routing component can be evaluated in isolation.

For our simulation results, we studied the performance of the algorithms for increasing traffic load, examining the evolution of the network status toward a saturation condition, and for temporary saturation conditions. For this topology, we study the performance of our routing strategies according a Poisson Law inter-arrival times statistical distribution.

2) Simulation results

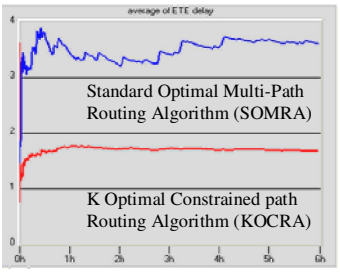

Fig. 2. Poisson law distribution simulations results.

As shown in Fig. 2 which represent time simulation versus the average packet delivery time, our probabilistic K Optimal Constrained path Routing Algorithm (KOCRA) give better results than the well-known N best optimal path routing Algorithm SOMRA. This is due to the fact that in our new approach, routers are able to take into account not only the average of delivery delay but also the waiting queue time. Thus, they are able to adapt their decisions very fast and in close concordance with the network dynamics. In spite of the many packages taking secondary ways, N-optimal routing does not

present better performances because it rests on a probabilistic method to distribute the load of the network over the closest cost paths, and not on the degradation of the times of routing. So, in classical approach, the routers take their decisions only according to the average of delivery delay and the exploration of potentials good paths, none trivially best and that can give us betters results, is not realized. Our approach, with the introduction of a probabilistic module, responds to this inconvenience and shows better results for Poisson law distribution of traffic. Thus, mean of average packet delivery time obtained by KOCRA is reduced by 37% compared to traditional N best optimal routing Algorithm.

5. AMDR based reinforcement learning in mobile ad hoc networks

AMDR (Adaptive Mean Delay Routing) is a new adaptive routing protocol based on probabilities and built around two exploration RL agents. Exploration agents gather mean delay information available at each node in their route and calculate total delay between source and destination. According to the delay value gathered, probabilistic routing tables are updated at each intermediate node. In order to deal with mobile nodes synchronisation we consider, in our protocol, delay estimation model proposed in [20], instead of instantaneous delay considered in the most oriented delay routing protocols.

Unlike data packets, control packets, used in adaptive routing, are sent in broadcast manner and so treated at IEEE 802.11, MAC layer differently than unicast packets. For this, we consider that trip delay of a control packet is not the same of a data packet.

In AMDR, routing function is determined by means of very complex interactions of forward and backward network exploration agents. Forward agents report network delay conditions to the backward ones. So, no node routing updates are performed by the forward agents.

AMDR uses two kinds of agents: Forward Exploration Packets (FEP) and Backward Exploration Packets (BEP). Forward agents explore the paths of the network, for the first time in reactive manner, but it continues the exploration proactively.

FEP packets create a probability distribution at each node for its neighbors. Backward agents are used to propagate the information gathered by forward agents through the network, and to adjust the routing table entries.

5.1. Updating routing tables

Routing tables are updated when a BEP agent is received. The probabilities updating can take many forms, and we have chosen updating rules (5), (6), (7) and (8) described in [21]. As soon as, routing table is calculated, data packets are then routed according to the highest probabilities in the probabilistic routing tables.

Unlike on demand routing protocols, there is no guarantee to route all packets on the same route because of the proactive exploration. The BEP agent make changes to the probability values at the intermediate and final node according to the following update rules:

$$p_{fd} \leftarrow (p_{fd} + r)\,(1+r) \quad (5)$$

$$p_{nd} \leftarrow p_{nd}/(1+r) \quad (6)$$

$$p_{nd} \leftarrow p_{nd} - rp_{nd} \quad (7)$$

$$p_{fd} \leftarrow p_{fd} + r(1-p_{fd}) \quad (8)$$

In both the above cases, the reinforcement parameter r can be defined as a function of

delay. Here, $r=k/f(c)$, where $k > 0$ and $f(c)$ is the cost function used in [21].

5.2. Flooding optimization

In order to improve the performance of our routing protocol, we introduce the MPR [22] concept in the broadcast process. However, the MPR selection according to native OLSR is unable to build path satisfying a given QoS request. To avoid this problem, we propose a new algorithm for MPR selection. We keep at each node a table called MPR table containing a partial view of MPR neighbors. Our algorithm takes into account the mean delay available at each node. The MPR selection algorithm based on mean delay is the same proposed for bandwidth in [22], unlike their approach for bandwidth MPR; we define only one kind of MPR which are delay MPR. Mean delay MPR selection algorithm is composed of the following steps:

1. A node Ni selects, first, all its neighbors that are the only neighbors of a two hop node from Ni.
2. Sort the remaining one-hop delay neighbors in increasing order of mean delay.
3. Consider each one-hop neighbor in that order: this neighbor is selected as MPR if it covers at least one two-hop neighbor that has not yet been covered by the previous MPR.
4. Mark all the selected node neighbors as covered and repeat step 3 until all two-hop neighbors are covered.

With the present MPR selection algorithm, we guarantee that paths having best delays will be discovered but there are any guarantees about the overhead generated [23].

5.3. Performance evaluation in mobility scenario

We use NS-2 simulator to implement and test AMDR protocol. We test the impact of mobility on AMDR and compare its performances with OLSR and AODV. We define a random topology of 50 nodes.

Table 1. Simulation settings scenario 2

Traffic model	Exponential
Surface of simulation	1000m,1000m
Packets size	512 byte
Bandwidth	1Mbs
Rate of mobility	5m /s , 10m/s
Number of connections	5, 10, 15, 20, 25
Rate	5 paquets/s
Simulation duration	500 s

Table 1 summarizes the simulation setting. We injected different loads of traffics. After each simulation we calculate the end to end delay realized by each protocol. Figure 3 summarizes our comparison. We can observe that with low load, there is no difference in end to end delays. However, more the network is loaded more AMDR is better in term of delay. Such performance is justified by the adaptation of AMDR to changes in the network load. In the case of AODV and OLSR an additional delay is impossible to circumvent for adapting to changes.

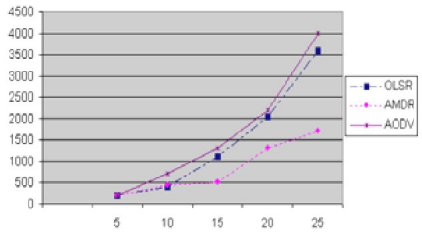

Fig. 3. Packets delay comparison for mobility scenario.

Comparing loss rate performance between AODV, AMDR and OLSR, shows in figure 4 that both AMDR and OLSR have, in a low loaded network, the same performance when AODV realises the best performances. However, in a high loaded network (case of 20 or 25 connexions), AODV becomes less good than AMDR and OLSR. We justify such results by the adaptation of AMDR to load changes when AODV needs more route request function.

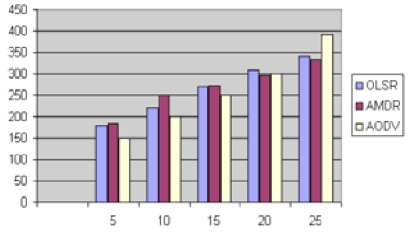

Fig. 4. Loss rate comparison for mobility scenario.

6. A system based reinforcement learning in packet scheduling communications network routing

In the dynamic environment the scheduler take the actual evolution of the process into account. It is allowed to make the decisions as the scheduling process actually evolves and more information becomes available. For that, we consider at each router an agent that can make decision. This decision-maker collects information gathered by mobile agents and then decides which action to perform after learning the current situation. We will focus on dynamic technique and will formulate the packet scheduling problem through several routers as a multi-agent Markov Decision Problem (MDP). As Machine learning techniques, we use reinforcement learning to compute a good policy in a multi-agent system. Simultaneous decision making in a dynamic environment is modelled using multi-agent Markov Decision Processes (MMDPs) [24]. However, learning in multi-agent system suffers from several limitations such the exponential growing of number of states, actions and parameters with the number of agents. In addition, since agents carry out actions simultaneously so they have evolving behaviours, transitions are non-stationary. Since centralized MAS may be considered as a huge MDP, we work with decentralized system where each agent learns individually in environment improved with information gathered by mobile agents.

6.1. The learning algorithm

The model of the environment's dynamics, the transition probabilities and rewards is unknown in learning of a single agent MDP and consequently the subsequent multi-agent MDP. So, the learning of the optimal solution of a problem is done by agents through interaction with the environment.

We describe the global scheduling problem as a multi-agent MDPs in a decentralized approach. We derive a multi-agent learning algorithm from traditional reinforcement learning method based on Markov decision process to construct global solutions from solutions to the individual MDPs. In this case, we assume that the agents work independently by making their trials in the simulated environment. The system state s is described by the space state of all agents; an action a^i

describes which queue is serviced in the time slot. Therefore, the goal of scheduling is to find an optimal policy π^* such that the rewards accumulated are maximized

The proposed algorithm converges to the optimal policy and optimal action value function for the multi-agent MDP since the difference between standard multi-agent and our decentralized multi-agent MDP model is the global states space for each action set A^i of an agent *i*.

The rewards may depend both on the current situation and on the selected action and express the desired optimization goal. In our approach, the global action *a* is a vector of single action made by distributed agents each associated with one of the *n* routers.

Learning here means iteratively improving the selection policy according to the maximization of the global reward. This is done by a Q-learning rule adapted to the local selection process (eq. 19). The learning rule relates the local scheduling process of agent *i* to the global optimization goal by considering the global reward *R*.

If Q^i converges the $Q^{i,*}$ predicts if the action a^i would be selected next. This action will be chosen by a policy greedy.

In a single-agent learning case, Q-learning converges to the optimal action independent of the action selection strategy. However, in a multi-agent situation, the action selection strategy becomes crucial for convergence to any joint action. A major challenge in defining a suitable strategy for the selection of actions is to make a trade-off between exploration of new policies and exploitation of existing policies.

In our research, we use a Boltzmann distribution [25] for the probability of choosing an action by each agent. In this strategy, each agent derive a scheduling policy from the current value of Q^i matrix and then update Q^i using the rewards from actions chosen by the current scheduling policy according to a probability distribution $\pi^i(s, a^i)$:

$$\pi^i\left(s, a^i\right) = \frac{exp\left(Q^i\left(s, a^i\right) / T\right)}{\sum_{a^{i'} \in A^i} exp\left(Q^i\left(s, a^{i'}\right) / T\right)} \quad (9)$$

where *exp* is the exponential function and *T* is a parameter called temperature. The value of the temperature determines the possibility for an agent to balance between exploration and exploitation. For high temperature, even when an expected value of a given action is high, an agent may still choose an action that appears less desirable. In contrast, low temperature values support more exploitation, as the agent is more expected to have discovered the true estimates of different actions. The three important settings for the temperature are the initial value, the rate of decrease and the number of steps until it reaches its lowest limit. This lower limit must be set to a value close enough to 0 to allow the learners to converge by stopping their exploration.

In our work, we start with a very high value for the temperature to force the agents to make random moves until the temperature reaches a low enough value to play a part in the learning. This is done when the agents are gathering information about the environment or the other agents. The temperature defined as a function of iterations is given by:

$$T(x) = (e^{-sx} * T_{\max}) + 1 \quad (10)$$

where *x* is the iteration number, s is the rate of decay and T_{max} is the starting temperature.

In this section we present an algorithm called DEMAL (Decentralized Multi-Agent Learning) that uses Q-learning and decentralization on the level of the action.

Algorithm DEMAL

Repeat

Initialize s = (s^1, ….., s^n)

Repeat

For each agent i
Choose a^i using Boltzman formula
Take action a^i , observe reward r^i and state s'
$Q^i(s, a^i) \leftarrow Q^i(s, a^i) + \alpha\{R + \gamma \max_{a^{i'}} [Q^i(s', a^{i'}) + \xi B(s', a^{i'})] - Q^i(s, a^i)\}$
$s \leftarrow s'$
until s is terminal
until algorithm converges

6.2. Performance evaluation

We carried out our evaluation in two stages. The first stage consists to realizing the scheduling on level of one router. For that, we just consider in this stage a single agent MDP. In the second stage, we solve the whole problem which concerns the optimization of the end to end queuing delay through the global scheduling. Hence, we apply our algorithm based on the multi-agent MDP in its decentralized version. We start to describe the context of the first phase.

In each router, an agent deals with scheduling N classes of traffic, where each traffic class has its own queue q_i, for i = 1…N. Let q_N denote the queue for best-effort traffic, which has no predefined delay requirements and R_1, R_2,.., R_{N-1} denote the delay requirements of the remaining classes. Let M_1, M_2,…, M_{N-1} denote the measured delays of these classes observed over the last P packets. We assume that all packets have a fixed size. We consider also that a fixed length timeslot is required for transmitting a packet and at most one packet can be serviced at each timeslot. The arrival of packets is described by a Bernoulli process, where the mean arrival rate μ_i for q_i is represented by the probability of a packet arriving for q_i in any timeslot. Our goal is to learn a scheduling policy that ensures $M_i \leq R_i$ for i=1,…,N-1. For the simulation, we used a three queue system that is Q_1, Q_2 and the best effort queue and the parameters of this simulation are given in table 2. We have considered two cases according to the availability of resource. For investigating the case where the output link capacity of the router is sufficient we assume that this capacity is 500 Kbps. In this case, a sufficient amount of capacity is provided for each queue so our algorithm satisfied the mean delay requirements for Q_1 and Q_2 (see fig.5). We have also observed that our approach requires 1.5 x 104 timeslots in terms of convergence time. In the second scenario (table 3) we consider the case where the output link capacity of the router is small and equal to 300 Kbps. The result of this case is shown in fig. 6. We observe that an allocation of a share of the available bandwidth is given to the delay-sensitive class Q_1 and then to Q_2 and the best effort queue. This is carried out on the basis of information gathered by a mobile agent. Also, we take $\varepsilon = 0.2$ and $\gamma = 0.5$.

In the second part of our evaluation, we consider a network with several routers connected to each other like in [26]. We introduce also the mobile agents to gather and distribute necessary and complete information in order to help the agents to update their knowledge of the environment. The figures 7 show that in both scenarios, the presence of mobile agents provides a better queuing delay for all routers.

Table 2. Simulation parameters: scenario 1.

Queue	Arrival Rate (packets/ timeslot)	Mean Delay Requirement	eBi Kbps
Q1	0.30	8	64
Q2	0.20	2	128
BE	0.40	Best-effort	Best-effort

Table 3. Simulation parameters: scenario 2

Queue	Arrival Rate packets/ timeslot	Mean Delay Requirement	eBi Kbps
Q1	0.30	4	128
Q2	0.20	6	256
BE	0.40	BE	BE

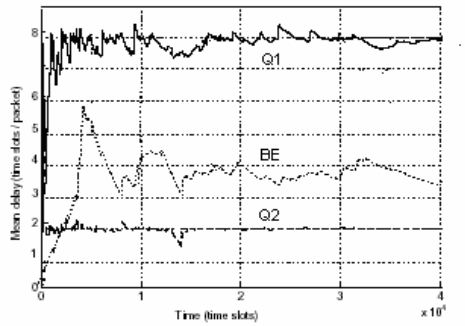

Fig. 5. Mean Delay for three classes.

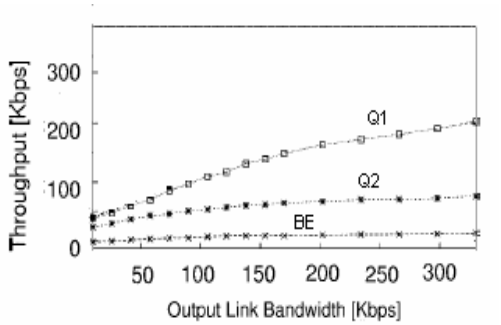

Fig. 6. Average throughput of three queues.

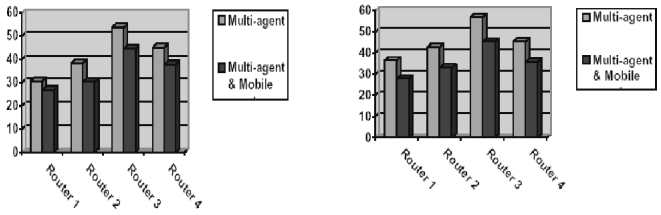

Fig. 7. Average queuing delay (left: scenario 1, right: scenario2).

7. Conclusion

We presented in this paper our system based on reinforcement learning for different network communication domains.

First of all, we have focused our attention in some special kind of Constrained Based Routing in wired networks which we called QoS self-optimization Routing. Our algorithm is based on a multi-path routing technique combined with the Q- Routing algorithm and is tested for improving distribution of traffic on N-Best paths. The learning algorithm is based on founding N-Best paths in term of hops router and the minimization of the average packet delivery time on these paths. The performance of our algorithm is evaluated experimentally with OPNET simulator for different levels of traffic's load and compared to standard optimal path routing algorithms. Our approach proves superior to a classical algorithms and is able to route efficiently in networks even when critical aspects are allowed to vary dynamically. The fact that the reinforcement signal is continuously updated, parameter's adaptation of our system take into account variations of traffic.

Secondary, we study the use of reinforcement leaning in AMDR algorithm in the case of Mobile Ad Hoc Networks. It is shown from simulation results that combining proactive exploration agents with the on-demand route discovery mechanism, the AMDR routing algorithm would give reduced end-to-end delay and route discovery latency with high connectivity. This is ensured because of the availability of alternative routes in our algorithm. The alone case where our approach can provide more important delay is the first connection where any route is yet established. On the other hand, the use of delay-MPR mechanism, guarantees that the overhead generated will be reduced.

In the last part, we address the problem of optimizing the queuing delay in several routers of a network, through a global packet scheduling. We formulated this problem as a multi-agent MDP and used the decentralized version since multi-agent MDPs usually have huge state and action spaces (because they grow exponentially with the number of agents). This decentralized MDP is improved by ant-like mobile agent on the level of each router to guarantee a global view of the system's state. We presented a modified Q- learning algorithm in the decentralized approach. Our simulation shows that the proposed approach leads to better results than when the multi- agent system acts alone.

Finally, extensions of the framework for using these techniques across hybrid networks to achieve end-to-end QoS needs to be investigated, in particular on large scalable networks. Another challenging area concerns the composite metric used in routing packets (especially residual bandwidth) which is so complex and the conditioning of different models in order to take into account other parameters like the information type of each flow packet (real-time, VBR, …).

References

[1] A. Mellouk, P. Lorenz, A. Boukerche, M.H.. Lee, "Impact of Adaptive Quality of Service Based Routing Algorithms in the next generation heterogeneous networks", *IEEE Communication Magazine, IEEE Press* Vol. 45, n°2, (2007) 65.

[2] M.R. Garvey, D.S. Jhonson, "Computers and Intractability: A Guide to the Theory of NP-Completeness". *Freeman*, San Francisco, 1979.

[3] Kuipers, F.A. P. Van Mieghem, "Conditions that impact the Complexity of QoS Routing", *IEEE/ACM Transaction on Networking*, Vol. 13(4) (2005) 717.

[4] M. Song, S. Sahni, "Approximation Algorithms for Multiconstrained Quality-of-Service Routing", *IEEE Transaction on Computers*, Vol. 55, No. 8, (2006) 1048.

[5] J.M. Jaffe, "Algorithms for Finding Paths with Multiple Constraints", *IEEE Networks* Vol. 14 (1984) 95.

[6] S. Sahni, Data Structures, Algorithms, and Applications in C++. second Edition, Silicon Press, 2005.

[7] T. Korkmaz, M. Krunz, "A Randomized Algorithm for Findind a Path Subject to Multiple QoS Requirements", *Computer Networks* Vol. 36 (2001) 251.

[8] X. Masip-Bruin et al., "Research challenges in QoS Routing", *Computer Communications* Vol. 29 (2006) 563.

[9] F.A. Kuipers, T. Korkmaz, M. Krunz, P. Van Mieghem, "Performance Evaluation of Constraint-Based Path Selection Algorithms", *IEEE Network* Vol. 18, No. 5 (2004) 16.

[10] B. Quoitin, S. Uhlig, "Modeling the Routing of an Automous System with C-BGP", *IEEE Network,* Vol. 19, No.6 (2005) 12.

[11] M. Yanuzzi, X. Masip-Bruin, O. Bonaventure, "Open Issues in Interdomain Routing : A Survey", *IEEE Network* Vol. 19, No.6 (2005) 49.

[12] W.D. Grover, "Mesh-based Survivable Transport Networks: Options and Strategies for Optical, MPLS, SONET and ATM Networking". *Ed. Prentice Hall PTR*, 2003.

[13] R.G. Gallager, "A minimum delay routing algorithm using distributed computations". *IEEE Transactions on Communications* 25(1) (1977) 73.

[14] A.E. Ozdaglar, D.P. Bertsekas, "Optimal Solution of Integer Multicommodity Flow Problem with Application in Optical Networks", *Proc. Of Symposium on Global Optimization*, June (2003) 411.

[15] R.S. Sutton, A.G. Barto, "Reinforcement Learning: An Introduction", *MIT Press/Bradford Books*, 1998.
[16] E. Gelenbe, L. Lent, Z. Xu, "Networking with Cognitive Packets", *Proc. ICANN 2002*, Madrid, Spain (2002) 27.
[17] J.A. Boyan, M.L. Littman, "Packet routing in dynamically changing networks: A reinforcement learning approach", *Advances in Neural Information Processing Systems 6,* Morgan Kaufmann, San Francisco, CA, (1994) 671.
[18] M. Dorigo, T. Stüzle, "Ant Colony Optimization". *MIT Press*, Cambridge, MA, 2004.
[19] D. Eppstein, "Finding the K shortest paths", S*IAM J. Computing 28:0* (1998) 652.
[20] A.M. Naimi, P. Jacquet, "One Hop Delay Estimation In 802.11 Ad Hoc Networks Using The OLSR Protocol", *Research Report INRIA, 2004*, N° 5327.
[21] J.S. Baras, H. Mehta, "A Probabilistic Emergent Routing Algorithm (PERA) for Mobile Ad Hoc Networks", *Proceedings of WiOpt '03: Modeling and Optimization in Mobile, AdHoc and Wireless Networks, Sophia-Antipolis*, France. 2003.
[22] D.Q. Nguyen, P. Minet, "Analysis of Multipoint Relays Selection in the OLSR Routing Protocol with and without QoS Support", *Research Report INRIA*, N° 6067, 2006.
[23] S. Ziane, A. Mellouk, "A Swarm Quality of Service Based Multi-Path Routing Algorithm (SAMRA) for Wireless Ad Hoc Networks", *International Review on Computers and Software Journal* Vol.1, N°1 (2006) 11.
[24] Puterman, M. Markov. "Decision Processes: Discrete Stochastic Dynamic Programming, Wiley-Interscience", 2005
[25] S. Kapetanakis, D. Kudenko, "Reinforcement learning of coordination in cooperative multi-agent systems", *Proceedings of AAAI* (2002) 326.
[26] M. Bourenane, A. Mellouk, D. Benhamamouche, "A QoS-based scheduling by Neurodynamic Learning". *System and Information Sciences Journal* Vol. 2, N° 2 (2007) 138.
[27] A. Mellouk, S. Hoceini, M. Cheurfa, "Reinforcing Probabilistic Selective Quality of service Routes in Dynamic Heterogeneous Networks" In *Elsevier Journal of Computer Communication*, Elsevier Ed., ISSN: 0140-3664, on line 2007, to appear.