Journal of Engineering Sciences, Assiut University, Vol. 34, No. 3, pp. 823-841, May 2006

PERFORMANCE ANALYSIS FOR QOS PROVISIONING IN MPLS/DIFFSERV- BASED IP NETWORKS WITH NON-PREEMPTIVE PRIORITY QUEUING SYSTEM

***Ibrahim I. Ibrahim**, Professor, Ibsmail@softhome.net*
***Abd ELrahman H. Elsawy**, Professor, Aelsawy@seegypt.com*
***Mahmoud M. Elmesalawy**, Demonstrator, Mlf_egt@yahoo.com*

Faculty of Engineering, Helwan University, Cairo, Egypt

(Received July 17, 2005 Accepted April 20, 2006)

ABSTRACT– *A typical operating environment of a packet switching (store-and-forward) computer communication network is that it is shared by many users with different classes of packets. Packets may be classified in a very general fashion by types of users, messages, applications, transactions, response time requirements, packet parameters such as packet rate and length, and by network parameters such as source-destination and path length. A well-designed network must provide access and performance assurance to all packet classes. An architecture for designing such a communication network is presented in [1]. This architecture is based on MPLS as an underlying technology used for IP transport and DiffServ-like mechanisms for QoS provision which use simple priority queuing to differentiate several traffic classes by assigning them different delay targets.*

In this paper, we introduce a mathematical model for performance evaluation for QoS Provisioning in MPLS/DiffServ-Based IP Networks. This mathematical model is formulated to evaluate the performance of the proposed architecture described in [1]. Two parts are considered in the evaluation. One, is the analysis of the proposed QoS routing algorithms that are used in the proposed architecture, the other is the analysis of the average packet delay over the network. An M/G/1 Non-preemptive priority queuing system is used as the evaluation model to analyze the average packet delay over the network. Relative MPLS/DiffServ with Non-Preemptive Priority Queuing system performance compared to MPLS/DiffServ without priority (FIFO)[2] is evaluated. The results indicate that for well-designed MPLS/DiffServ equipments, original IP packet delay can be reduced by 39% or more with FIFO queuing system [2] and 42% or more with Non-preemptive priority queuing system.

KEYWORDS: *QoS, CoS, MPLS, DiffServ, DSCP, IP.*

1. INTRODUCTION

In providing multiple services in a single network there is two great directions which could be labeled by Quality of Service (QoS) and Class of Service (CoS). With QoS usually some sort of per-flow queueing is associated which allows to give exact performance guarantees and allows to emulate flow isolation. The latter fact allows to some degree to apply basic queueing theory results for average case behavior of a flow. The drawback of per-flow traffic control mainly is that it runs into scalability problems in high multiplexing environments and is usually considered infeasible for the backbone of large networks as, e.g., the Internet [9].

By CoS, a class-based differentiation between traffic is meant. The associated per-class queueing achieves high scalability by supporting only a limited number of traffic classes in the network. Within one class flows compete for the available resources in that class. Note that sometimes CoS is considered as subset of QoS, but for the sake of contrasting the per-flow and per-class traffic control approach we use these two terms (as it is often done in the literature). There are several alternatives for scheduling the different classes in a class-based packet network, the simplest variant is to give non-preemptive priority to packets belonging to more important classes (numerically lower). This is, in the realm of queueing systems, usually called *non-preemptive priority queueing*. Due to its simplicity non-preemptive priority scheduling is usually implemented on today’s routers.

The rest of the paper is structured as follows: In Section 2, we introduce the analytical evaluation model. In section 3, we determine parameters in the evaluation model and then draw a set of conclusions on the results. Finally, section 4, is a short conclusion for this paper.

2. ANALYSIS OF THE PROPOSED QOS ROUTING ALGORITHMS

This section presents a quantitative analysis of the proposed QoS routing algorithms described in [1]. This analysis is based on simulation experiments and mathematical model. To evaluate these algorithms we must explore four different components which are:

1. Weight vector in LCF
2. Processing time for QoS path computation.
3. Complexity of QoS routing table Computation.
4. Blocking probability (Call rejection rate) for the proposed QoS algorithms.

The first three components are described in [2], so we focus in this paper on the last component which aims to measure the blocking probability (Call rejection rate) for the proposed two QoS algorithms.

2.1 Blocking Probability For The Proposed Two Qos Routing Algorithms

Finding a path in the network for each traffic flow able to guarantee some quality parameters such as bandwidth and delay is the task of QoS routing algorithms proposed in [1]. The proposed algorithms are designed so that when more than one path satisfying the bandwidth demand exists, the selection of the path aims to achieving the second constraint and minimizing the blocking probability of future requests.

In this section a mathematical model that allows to determine a theoretical lower bound to the connection rejection rate based on the extended erlang formula is presented. We compare to such bound the performance of the proposed algorithms and show that the blocking probability of this algorithms, is quite close to the bound.

2.2 Mathematical Model Based On The Extended Erlang Formula To Measure The Connection Rejection Rate

Let us consider a directed graph $G=(N,A)$ defined by a set of nodes, N, and a set of arcs, A. Every arc $(i,j)\in A$ is characterized by a capacity C_{ij} that represents the maximum flow that can be shipped over that arc.

Let us consider a network system with C servers, to which M different traffic classes are offered. The connections belonging to the class i request b_i bandwidth units. The connections arrival process is a Poisson process with average equal to λ_i, while the connections duration is distributed according to a generic distribution with mean service time $E(T_{s_i})$. Let $\lambda_t=\sum_{i=1}^{M}\lambda_i$ be the total load offered to the network and $\rho_i=\lambda_i E(T_{s_i})$ the traffic offered to the network by each class.

The state of this system is defined by the number of connections that occupy the servers. If n_i is the number of such connections belonging to the class i, the set of all the possible states is expressed as $S=\{n:(n_1,....,n_M)\,|\,X\le C\}$, with X indicating the total occupation of all the servers, calculated as:

$$X=\sum_{i=1}^{M}n_i b_i = n.b \tag{1}$$

The state space of the system Ω consists of the allowed states as shown in **Figure 1**.

$$\Omega=\{n:n.b\le C\} \tag{2}$$

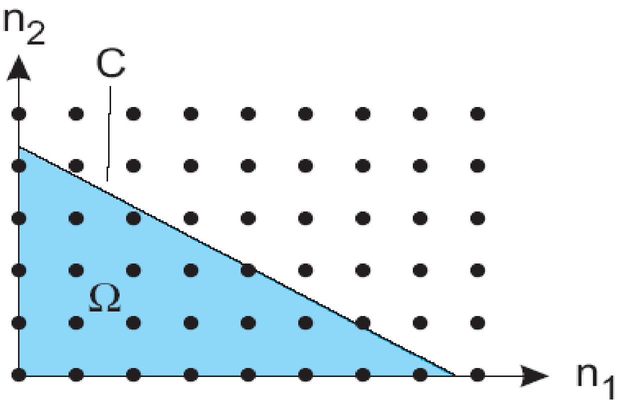


Figure 1: The state space of the link Ω with capacity C.

The set of blocking states β_i of traffic class i consists of those allowed states where no more class-i calls can be admitted without violating the capacity constraint.

$$\beta_i = \{n : C - b_i < n.b \leq C\} \tag{3}$$

As shown from equation (3), The greater b_i, the larger is the set of blocking states.

Denote by e_i the vector with all components equal to zero, except for component i, *which* is one. By means of this, the set of blocking states can be written as:

$$\beta_i = \{n : n \in \Omega, n + e_i \notin \Omega\} \tag{4}$$

A new class-i call would take the system outside the allowed region.

Figures 2 and **3** show the blocking states of classes 1 and 2, respectively. The set of blocking states β_i can be characterized as:

- "Last states" at the end of columns is the i-direction.
- States between the hyperplanes corresponding to the capacity occupancies C and $C - b_i$. This can be obtained by translating the $C -$ hyperplane by one step in the i-direction towards the origin.

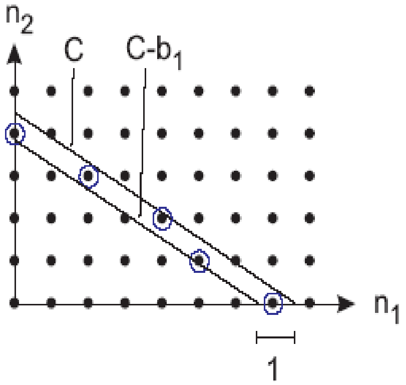


Figure 2: Blocking states of class 1.

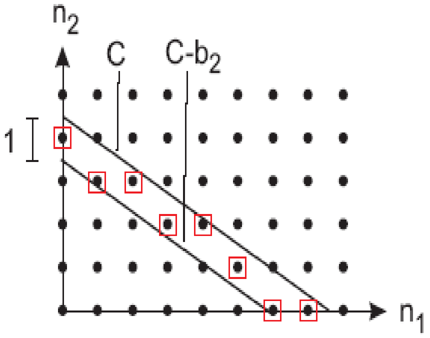


Figure 3: Blocking states of class 2.

Under the assumptions made, the system constitutes a Markov process which is a multi-dimensional birth-death process and the state transition from any state to a typical state within the allowed region are shown in **Figure 4**.

The balance equations have a product form solution as follows:

$$\pi = \frac{1}{G}.\frac{\rho_1^{\;n_1}}{n_1!}\cdots\frac{\rho_M^{\;n_M}}{n_M!} = \frac{1}{G}.\prod_{i=1}^{M}\frac{\rho_i^{\;n_i}}{n_i!} \tag{5}$$

Where G is the normalization constant that ensures that the π sum to one.

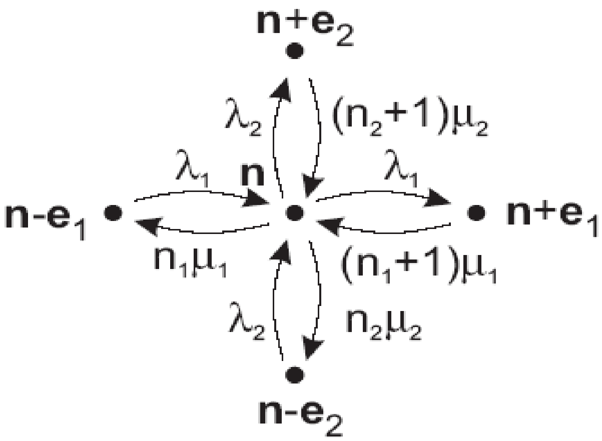


Figure 4: Multidimensional birth-death process.

$$G = \sum_{n \in \Omega} \frac{\rho_1^{\;n_1}}{n_1!} \cdots \frac{\rho_M^{\;n_M}}{n_M!} = \sum_{n \in \Omega} \prod_{i=1}^{M} \frac{\rho_i^{\;n_i}}{n_i!} \tag{6}$$

Using the steady state probability calculated with equation (5) we can calculate the loss probability of generic class i, π_i, as follows:

(1) For FIFO queuing system

$$\pi_i = \sum_{n \in \beta_i} \pi(n_1, \ldots, n_M) \tag{7}$$

(2) For priority queuing system

For M priority classes $1,2,\ldots,M$. The highest–priority class is taken to be 1 and the lowest is M, in descending order as labelled so that a customer with the priority P has higher priority than customers with priority $P+1$. The loss probability of generic class i with priority P, π_P, is given as follows:

$$\pi_P = \sum_{n \in \beta_P} \pi(n_1, \ldots, n_P) \tag{8}$$

In which β_P is the set of the blocking states for the class i with priority P, defined as:

$$\beta_P = \{ n : C - b_P < \sum_{i=1}^{P} n_i b_i \le C \}.$$

The overall connection rejection probability P_{rej} is then given by:

$$P_{rej} = \sum_{i=1}^{M} \frac{\lambda_i \pi_i}{\lambda_t} \tag{9}$$

Example Of Calculating The Blocking Probabilities

Suppose a link with capacity C is used by connections belonging to two different classes. The parameters are:

$$C = 6Mbps, \mu = 10$$
$$\rho_1 = 1, b_1 = 1Mbps$$
$$\rho_2 = 0.5, b_2 = 2Mbps$$

The state space, the allowed states and the blocking states of the traffic classes are shown in **Figure 5**.

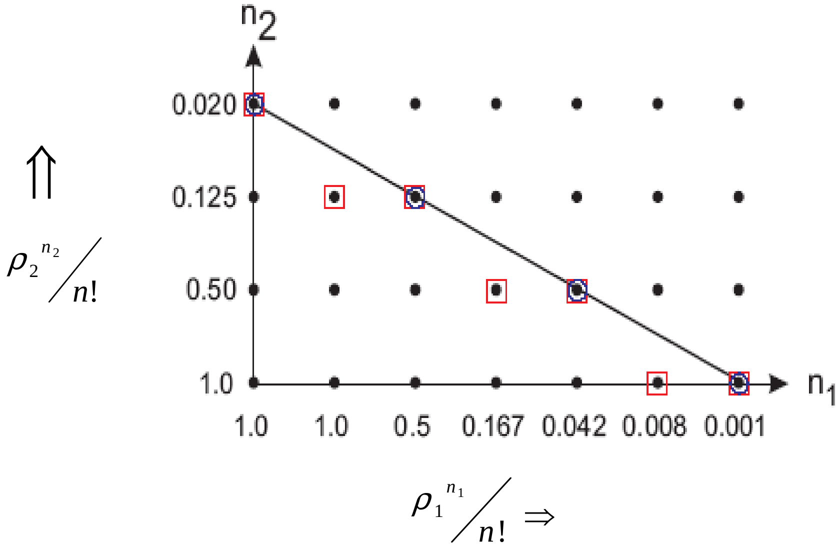


Figure 5: The state space, the allowed states and the blocking states of the traffic classes.

The normalization factor G is the sum of the unnormalized probabilities,

$$G = \sum_{n\in\Omega} \frac{\rho_1^{n_1}}{n_1!} \cdots \frac{\rho_M^{n_M}}{n_M!} = \sum_{n\in\Omega} \prod_{i=1}^{2} \frac{\rho_i^{n_i}}{n_i!} = 4.41$$

Assuming that FIFO queuing system, the sums of the normalized probabilities over the sets of blocking states (loss probability) are:

$$\pi_1 = \sum_{n\in\beta_1} \pi(n_1, n_2) = \frac{1}{G} . \sum_{n\in\beta_1} \prod_{i=1}^{2} \frac{\rho_i^{n_i}}{n_i!} = \frac{1}{4.41} * 0.11 = 2.4\%$$

$$\pi_2 = \sum_{n\in\beta_2} \pi(n_1, n_2) = \frac{1}{G} . \sum_{n\in\beta_2} \prod_{i=1}^{2} \frac{\rho_i^{n_i}}{n_i!} = \frac{1}{4.41} * 0.11 = 7.3\%$$

As shown from the results of this example, the traffic of class 2 has higher loss probability than traffic of class 1, which has high capacity requirement.

The overall connection rejection probability P_{rej} is then given by:

$$P_{rej} = \sum_{i=1}^{2} \frac{\lambda_i \pi_i}{\lambda_t} = \frac{10}{15} * 2.4 + \frac{5}{15} * 7.3 = 4\%$$

2.3 Simulation And Numerical Results

In this section we compare the performance of the proposed algorithms in the two cases of queuing systems (FIFO and priority queuing system). The performance function we consider is the percentage of rejected calls versus the average total load offered to the network. We also compare the performance achieved by these routing algorithms with the theoretical bound provided by the mathematical model that have been presented in the previous section.

To perform this evaluation we have considered the NSFNET network topology shown in **Figure 6**, with 14 nodes and 21 links, all having a capacity equal to 30 bandwidth units. The traffic offered to the network is divided into M = 5 traffic classes with the following parameters:

Service class-1(SC-1) $\Rightarrow$ $b_1 = 5$ units.

Service class-2(SC-2) $\Rightarrow$ $b_1 = 10$ units.

Service class-3(SC-3) $\Rightarrow$ $b_1 = 12$ units.

Service class-4(SC-4) $\Rightarrow$ $b_1 = 15$ units.

Service class-5(SC-5) $\Rightarrow$ $b_1 = 20$ units.

Figures 7-10 show the results of our performance evaluation. **Figure 7** shows the overall call rejection probability for the two proposed algorithms which compared to the lower bound calculated by the mathematical model in equation (9). Figure 8 shows the call rejection probability for different service class with different bandwidth request. **Figures 9** and **10** show how the call rejection probability is affected by the priority level of the traffic class.

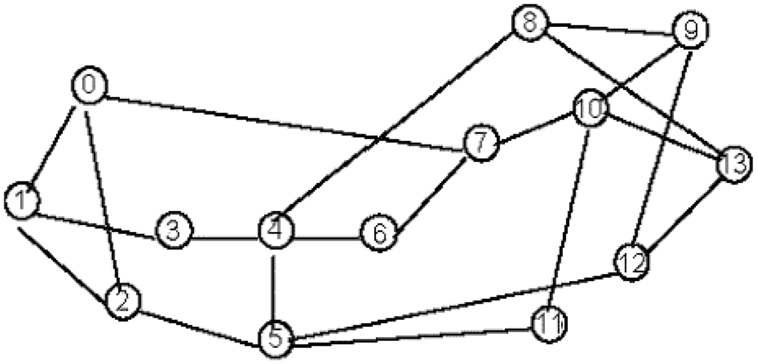


Figure 6: (14 Nodes, 21 Links) NSFNET topology.

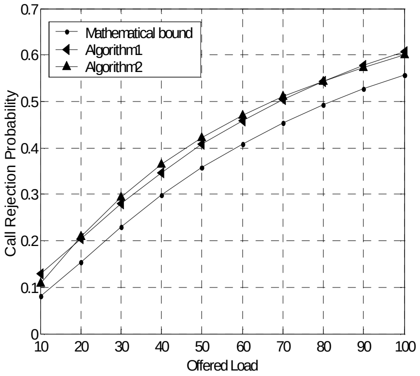


Figure 7: Connection rejection probability versus average offered load to the network.

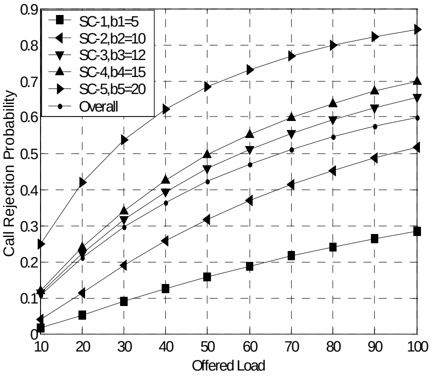


Figure 8: Connection rejection probability for different traffic classes.

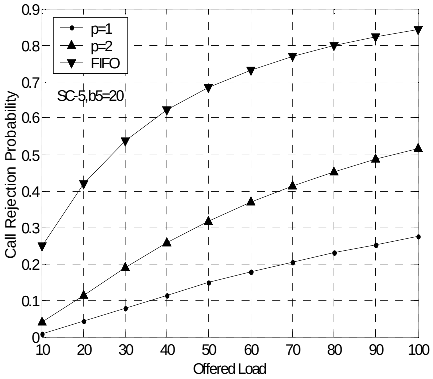


Figure 9: Connection rejection probability for different priority levels

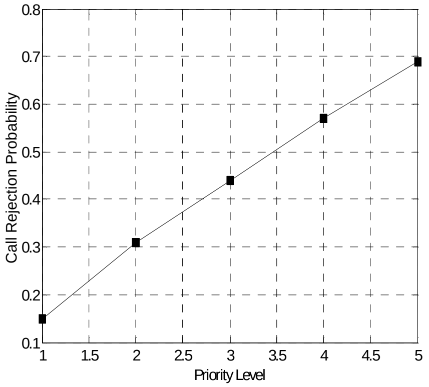


Figure 10: Connection rejection probability versus priority level

From the figures we can get the following conclusions:

1. As shown in **Figure 7**, the call rejection probability for the two algorithms are approximately the same and they are very close to the lower bound provided by the mathematical model.
2. **Figure 8** shows that the blocking probability for connections that requested large bandwidth is higher than the blocking probability for connections that requested small bandwidth.
3. **Figures 9** and **10** shows that as the priority level of the traffic class is increased as the blocking probability for connections belonging to that class is decreased, so critical traffic must be assigned a higher priority level.

3. EVALUATION MODEL ANALYSIS FOR AVERAGE PACKET DELAY OVER THE NETWORK

In this section we model a data network as a network of communication channels whose purpose is to move data messages from their origin to their destination. Each channel was modeled as a server serving a queue of data messages awaiting transmission as shown in **Figure 11.** The main metric used for the performance evaluation of the network was $E(T_{ETE})$, the average time it took for messages to move across the network.

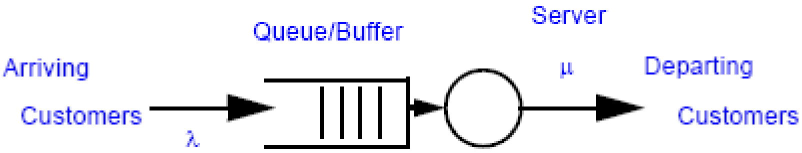


Figure 11: Basic Queueing Theory Model.

Assume that the customers are divided into M priority classes. Customers of class i are assumed to arrive according a Poisson process with intensity λ_i (packets/sec), and the mean service time is $E(T_{s_i})$ sec. The second moment of the service time distribution is denoted $E(T_{s_i}{}^2)$, and the offered traffic (utilization) is $\rho_i = \lambda_i E(T_{s_i})$. Then the total arrival rate (packets/sec) can be expressed as $\lambda = \sum_{i=1}^{M} \lambda_i$ which also is Poisson arrival process. The resulting service time distribution then becomes a weighted sum of service time distributions of individual classes and the total mean service time becomes: $E(T_s) = \sum_{i=1}^{M} \frac{\lambda_i}{\lambda} E(T_{s_i})$ and the total second moment is:

$E(T_s{}^2) = \sum_{i=1}^{M} \frac{\lambda_i}{\lambda} E(T_{s_i}{}^2)$. The total offered traffic is:

$\rho = \sum_{i=1}^{M} \rho_i = \sum_{i=1}^{M} \lambda_i E(T_{s_i}) = \lambda E(T_s)$ and the remaining mean service time at a random point of time becomes:

$E(T_R) = \sum_{i=1}^{M} \frac{\lambda_i}{2} E(T_{s_i}{}^2) = \sum_{i=1}^{M} E(T_{R_i})$, Where $E(T_{R_i}) = \frac{\lambda_i}{2}.E(T_{s_i}{}^2)$.

Based on this model, the packet delay per channel for any class can be expressed as:

$$E(T) = E(T_w) + E(T_s) + E(T_{ser}) + T_g = E(T_q) + E(T_{ser}) + T_g \qquad (10)$$

Where: $E(T_w)$ is the average waiting time per packet, $E(T_s)$ is the average transmission time (service time) per packet, $E(T_q) = E(T_w) + E(T_s)$ is the average queuing (Sojourn) time per packet, $E(T_{ser})$ is the mean searching time (lookup time) in the routing table, and T_g is the propagation delay of the packet on the channel.

3.1 Average Waiting Time $E(T_w)$ Of Any Class

For M priority classes $1,2,...,M$. The highest–priority class is taken to be 1 and the lowest is M , in descending order as labelled so that a customer with the priority P has higher priority than customers with priority $P+1$. In non-preemptive system, a service in progress is not interrupted.

Consider class $P(1 \le P \le M)$ in particular. Let a typical customer of this class arrive at arbitrary time which has mean service time $E(T_{S_P})$, arrival intensity λ_P, and the mean waiting time of this class is $E(T_{w_P})$. The mean waiting time of class P customers $E(T_{w_P})$ can be derived directory by considering the following three contributions:

(1) It must wait a random amount of time (residual service time) $E(T_R)$ until the customer currently in service completes service. For the work-conserving non-preemptive queuing system under discussion here (the server always serves a customer if one waiting to be served), this is independent of queue discipline:

$$E(T_R) = \sum_{i=1}^{M} \frac{\lambda_i}{2} E(T_{s_i}{}^2) \tag{11}$$

(2) It must wait a waiting time, due to the customers in the queue with priority P or higher, which are already in the queue at the arrival time. We can derive this waiting time using littel´s theorem.

$$\sum_{i=1}^{P} E\ (T_{s_i}).\lambda_i.E\ (T_{w_i}) \tag{12}$$

(3) It must wait a waiting time to service the customers with higher priority than p arriving during the wait time $E(T_{w_P})$we can consider this waiting time as follows:

$$\sum_{i=1}^{P-1} E\ (T_{s_i}).\lambda_i.E\ (T_{w_P}) \tag{13}$$

Putting these three observations together, we write:

$$E(T_{w_P}) = E(T_R) + \sum_{i=1}^{P} E(T_{s_i}).\lambda_i.E(T_{w_i}) + \sum_{i=1}^{P-1} E(T_{s_i}).\lambda_i.E(T_{w_P}) \tag{14}$$

We can solve equation (14) for the wait time of each class recursively, starting with the highest priority class 1 to get a general formula of the mean waiting time of any service class with priority P as follows:

$$E(T_{w_P}) = \frac{E(T_R)}{(1-\bar{\rho}_P)(1-\bar{\rho}_{P-1})} = \frac{\sum_{j=1}^{M} \frac{\lambda_j}{2} E(T_{s_j}{}^2)}{(1-\bar{\rho}_P)(1-\bar{\rho}_{P-1})} \tag{15}$$

Where, $\bar{\rho}_P = \sum_{j=1}^{P} \rho_j = \sum_{i=1}^{P} \lambda_j . E(T_{s_j})$

3.2 Sojourn (Queuing) Time $E(T_q)$ Of Any Class

$$E(T_{q_P}) = E(T_{w_P}) + E(T_{s_P}) = \frac{\sum_{j=1}^{M} \frac{\lambda_j}{2} E(T_{s_j}{}^2)}{(1-\bar{\rho}_P)(1-\bar{\rho}_{P-1})} + E(T_{s_P}) \tag{16}$$

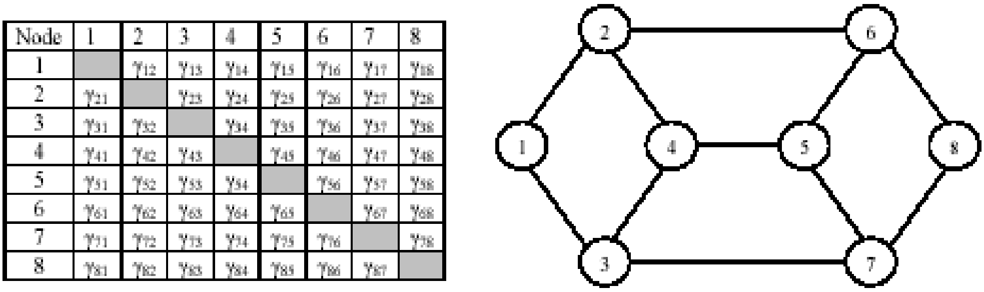
$$E(T_{q_P}) \text{ For channel } i = E(T_{q_{P_i}}) = \frac{\sum_{j=1}^{M} \frac{\lambda_{j_i}}{2} E(T_{s_{j_i}}{}^2)}{(1-\bar{\rho}_{P_i})(1-\bar{\rho}_{(P-1)_i})} + E(T_{s_{P_i}}) \tag{17}$$

Where $E(T_{q_P})$ is the average delay in passing through node i (i.e. channel i) for class P.

3.3 Average Packet Delay Across The Network $E(T_{ETE})$ Of Any Class

Average packet delay across the network of any class is the mean end-to-end delay $E(T_{ETE})$, which can be calculated as follows: Given

- The network topology with n- nodes and m-links as in **Figure 12**.
- Traffic matrix for each class as in **Figure 12**.
- Routing strategy (Routing table): distribution of traffic across the links (channels).



Node	1	2	3	4	5	6	7	8
1		γ_{12}	γ_{13}	γ_{14}	γ_{15}	γ_{16}	γ_{17}	γ_{18}
2	γ_{21}		γ_{23}	γ_{24}	γ_{25}	γ_{26}	γ_{27}	γ_{28}
3	γ_{31}	γ_{32}		γ_{34}	γ_{35}	γ_{36}	γ_{37}	γ_{38}
4	γ_{41}	γ_{42}	γ_{43}		γ_{45}	γ_{46}	γ_{47}	γ_{48}
5	γ_{51}	γ_{52}	γ_{53}	γ_{54}		γ_{56}	γ_{57}	γ_{58}
6	γ_{61}	γ_{62}	γ_{63}	γ_{64}	γ_{65}		γ_{67}	γ_{68}
7	γ_{71}	γ_{72}	γ_{73}	γ_{74}	γ_{75}	γ_{76}		γ_{78}
8	γ_{81}	γ_{82}	γ_{83}	γ_{84}	γ_{85}	γ_{86}	γ_{87}	

Figure 12. Network topology and traffic matrix of class P.

$\gamma_P = \sum_{i=1}^{n}\sum_{j=1}^{n}\gamma_{ij}$, Where γ_{ij} is the source destination traffic and γ_P is the network throughput for class P in message/sec. $\lambda_P = \sum_{i=1}^{m}\lambda_{P_i}$ is the total arrival intensity on all links (channels) for class P , λ_{P_i} is the average arrival intensity on ith channel for class P as determined from the traffic matrix and the routing strategy and accounting for the effect of acknowledgement traffic, and thus, the average path length for messages (average number of hops/message) = $\bar{n} = \frac{\lambda_P}{\gamma_P}$.

Mean per link delay of class P = $E(T_P) = \frac{1}{\lambda_P}\sum_{i=1}^{m}\lambda_{P_i}E(T_{q_P}) + E(T_{ser}) + \bar{T}_g$

$$= E(T_P) = \frac{1}{\lambda_P}\sum_{i=1}^{m}\lambda_{P_i}\left(E(T_{q_P}) + T_{g_i}\right) + E(T_{ser}) \tag{18}$$

Where $\bar{T}_g = \frac{1}{\lambda_P}\sum_{i=1}^{m}\lambda_{P_i}T_{g_i}$ is the average propagation delay over m-links for class P and T_{g_i} is the propagation delay for link number i.

Thus, the average packet delay across the network of class P = $E_P(T_{ETE}) =$ mean per link delay of class P * average path length.

$$E_P(T_{ETE}) = \bar{n}\left[\frac{1}{\lambda_P}\sum_{i=1}^{m}\lambda_{P_i}\left(E(T_{q_P}) + T_{g_i}\right)\right] + E_t(T_{ser}) \tag{19}$$

$$E_P(T_{ETE}) = \frac{1}{\gamma_P}\sum_{i=1}^{m}\lambda_{P_i}\left(E(T_{q_P}) + T_{g_i}\right) + E_t(T_{ser})$$

$$= \frac{1}{\gamma_P}\sum_{i=1}^{m}\lambda_{P_i}\cdot\left[\frac{\sum_{j=1}^{M}\frac{\lambda_{j_i}}{2}E(T_{s_{ji}}{}^{2})}{(1-\bar{\rho}_{P_i})(1-\bar{\rho}_{(P-1)_i})} + E(T_{s_{P_i}}) + T_{g_i}\right] + E_t(T_{ser}) \tag{20}$$

Where $E_t(T_{ser})$ is the total mean searching (Look up) time over the path (End-To-End).

3.4 Total Mean Searching Time $E_t(T_{ser})$ For IP And MPLS/Diffserv Domains

As described in [2], the shortest path in IP domain consists of $\bar{n}$ nodes with the longest matching search. So the total mean searching time over the path (E-T-E) is:

$$E_t(T_{ser)} = \bar{n}.E_{IP}(T_{ser}) \tag{21}$$

$$E_t(T_{ser}) = \bar{n}\left[c_0 + c_1\log(N) + c_2(\frac{4-F}{4})\log(N) + 2c_3\right] \tag{22}$$

Where $E_{IP}(T_{ser})$ is the mean searching time per node, N denotes the routing table size, M denotes the number of DSCPs (Number of classes), c_0 denotes the constant part of search time, c_1 denotes the time spent at each node on the forward path, c_2 denotes the time spent backtracking each node and c_3 denotes the time it takes for a key comparison.

As also described in [2], the shortest path in MPLS/DiffServ domain consists of $\bar{n}$-nodes, one of them has exact and longest matching search (Engress node) and the remaining $(\bar{n}-1)$ nodes have exact matching search. In the engress node of the MPLS/DiffServ domain we first look for the DSCP (Exact matching search) and then look for the destination network prefix (Longest matching search), So the total mean searching time over the path (E-T-E) resulting in an F byte match is:

$$E_t(T_{ser)} = E_{IP-Diff}(T_{ser}) + (\bar{n}-1)E_{MPLS}(T_{ser}) \tag{23}$$

$$E_t(T_{ser}) = \left[e_0 + e_1\log(M) + e_3 + c_0 + c_1\log(N) + c_2(\frac{4-F}{4})\log(N) + 2c\right]$$
$$+ (\bar{n}-1)\left[d_0 + d_1\log(N) + d_3\right] \tag{24}$$

d_0, d_1, d_3 Have the same meaning with c_0, c_1, c_3, but they should be smaller, for labels are shorter than IP prefixes and MPLS forwarding table lookup is mainly carried out by hardware and e_0, e_1, e_3 have the same meaning with d_0, d_1, d_3 but should be smaller, for DSCPs (8-bit) are shorter than labels (20-bit).

By using equation (20) and the total mean searching time $E_t(T_{ser})$, we can write an expression for the total mean end-to-end delay over the network for class P in IP and MPLS/DiffServ domains as follows:

- For IP domain:

$$E_P(T_{ETE}) = \frac{1}{\gamma_P}\sum_{i=1}^{m}\lambda_{P_i}.\left[\frac{\sum_{j=1}^{M}\frac{\lambda_{j_i}}{2}E(T_{s_{j_i}}{}^2)}{(1-\bar{\rho}_{P_i})(1-\bar{\rho}_{(P-1)_i})} + E(T_{s_{P_i}}) + T_{g_i}\right] + \bar{n}\left[c_0 + c_1\log(N)\right.$$
$$\left. + c_2(\frac{4-F}{4})\log(N) + 2c_3\right] \tag{25}$$

- For MPLS/DiffServ domain:

$$E_P(T_{ETE}) = \frac{1}{\gamma_P}\sum_{i=1}^{m}\lambda_{P_i}\cdot\left[\frac{\sum_{j=1}^{M}\frac{\lambda_{j_i}}{2}E(T_{s_{j_i}}{}^{2})}{(1-\bar{\rho}_{P_i})(1-\bar{\rho}_{(P-1)_i})} + E(T_{s_{P_i}}) + T_{g_i}\right] + \Big[e_0 + e_1\log(M)$$

$$+ e_3 + c_0 + c_1\log(N) + c_2\left(\frac{4-F}{4}\right)\log(N) + 2c_3\Big] + (\bar{n}-1)\left[d_0 + d_1\log(N) + d_3\right]$$

(26)

Also we can write the total mean end-to-end delay over the network for IP, MPLS and MPLS/DiffServ domains without priority queuing (FIFO) as described in [2] as follows:

$$E_{IP}(T_{ETE}) = \frac{1}{\gamma}\sum_{i=1}^{m}\left[\frac{2\lambda_i - \lambda_i\rho_i(1-\mu^2 c^2{}_i\sigma_s{}^2)}{2(\mu c_i - \lambda_i)} + \lambda_i T_{g_i}\right] + \bar{n}\Big[c_0 + c_1\log(N)$$

$$+ c_2\left(\frac{4-F}{4}\right)\log(N) + 2c_3\Big] \qquad (27)$$

$$E_{MPLS}(T_{ETE}) = \frac{1}{\gamma}\sum_{i=1}^{m}\left[\frac{2\lambda_i - \lambda_i\rho_i(1-\mu^2 c^2{}_i\sigma_s{}^2)}{2(\mu c_i - \lambda_i)} + \lambda_i T_{g_i}\right] + \Big[c_0 + c_1\log(N)$$

$$+ c_2\left(\frac{4-F}{4}\right)\log(N) + 2c_3\Big] + (\bar{n}-1)\left[d_0 + d_1\log(N) + d_3\right] \qquad (28)$$

$$E_{MPLS-Diff}(T_{ETE}) = \frac{1}{\gamma}\sum_{i=1}^{m}\left[\frac{2\lambda_i - \lambda_i\rho_i(1-\mu^2 c^2{}_i\sigma_s{}^2)}{2(\mu c_i - \lambda_i)} + \lambda_i T_{g_i}\right] + e_0$$

$$+ e_3 + e_1\log(M) + c_0 + c_1\log(N) + c_2\left(\frac{4-F}{4}\right)\log(N) + 2c_3$$

$$+ (\bar{n}-1)\left[d_0 + d_1\log(N) + d_3\right] \qquad (29)$$

Where $\frac{1}{\mu}$ is the mean packet (Message) length, σ_i^2 is the variance, c_i, λ_i are the capacity and arrival intensity of channel i respectively, and γ is the network throughput in message/sec.

4. PERFORMANCE EVALUATION OF THE PROPOSED ARCHITECTURE

In this section we will use our model described in section 3 to evaluate the performance of the proposed architecture in [1]. Our evaluation aims to show the improvement in the average packet delay through the use of priority queuing system, so a relative performance of MPLS/DiffServ with Priority Queuing compared to MPLS/DiffServ and IP without priority queuing is evaluated.

To achieve this, we have to get parameters in our model. However these parameters are closely related to specific hardware and software, such as c_0, c_1, c_2, c_3, and we can not obtain them [4]. Therefore we do not intend to evaluate the absolute performance of MPLS/DiffServ, which is also inaccurate and less persuasive. What we evaluate here is the relative performance.

4.1 Parameters Determination

In our model there are various parameters and they must be determined to carry out evaluation. First, is to determine c_i, d_i and e_i. As our goal is to evaluate relative performance, the absolute values of these parameters are not important. We define $R_i = \frac{d_i}{c_i}$ to indicate the ratio that MPLS is faster than IP. The values of d_i and c_i depends on the searching process (Hardware or Software) and the IP prefix or label length. Since the searching in IP routing table is carried out by software and the searching in MPLS forwarding table is mainly carried out by hardware, we can write the following relations: $d_i \propto L_d H_d$, $c_i \propto L_c S_c$, and $R_i = \frac{d_i}{c_i} = \frac{L_d H_d}{L_c S_c} = \frac{L_d}{L_c} SR$, where L_d and L_c indicates the MPLS label length and IP prefix length respectively, $SR = \frac{H_d}{S_c} < 1$is the searching ratio to indicate that searching carried by hardware (H_d) is faster than searching carried by software (S_c). We also define $\bar{R}_i = \frac{e_i}{c_i} = \frac{L_e S_e}{L_c S_c}$to indicate the ratio that searching on specific DSCP is faster than searching on IP prefix in the routing table where L_e: denotes the length of the DSCP (8-bit). Since the IP prefix lookup and DSCP lookup processes are carried out in the same node (Engress node), we can put $S_e = S_c$and thus $\bar{R}$ reduces to $\frac{L_e}{L_c}$. For simplicity we assume that $R_i = R = \frac{d_i}{c_i} = \frac{L_d H_d}{L_c S_c} = \frac{L_d}{L_c} SR$ and $\bar{R}_i = \bar{R} = \frac{e_i}{c_i} = \frac{L_e}{L_c}$. For a real BGP routing table that contains 21218 prefix, the prefix length statistics result is shown in **Figure 18** [4], [7]. From this distribution we can assume that the average IP prefix length$L_c = 24 - bits$. By putting $L_d = 20 - bits$and $L_e = 8 - bits$we can write the following relations: $R = \frac{d_i}{c_i} = \frac{L_d H_d}{L_c S_c} = \frac{5}{6} SR$ and

$\bar{R} = \frac{e_i}{c_i} = \frac{1}{3}$. From these two relations, we can get the values of d_i and e_i as a function of SR for given values of c_i.

4.2 Performance Evaluation

Figures 13-17 shows the results of our performance evaluation. In the evaluation we set c_0, c_1, c_2, c_3 respectively to be 5ns, 1ns, 1ns, 1ns. **Figures 3-5** shows the average packet delay as a function of lookup table size (N) under M=8 which is the maximum number of service classes as described in [1]. **Figure 6** shows the MPLS/DiffServ performance gain as a function of N. We also adjust the value of SR to be 0.8, 0.9, 1 and results are displayed in figures.

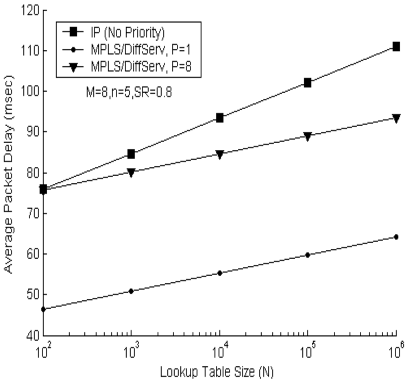


Figure 13: Average Packet Delay for different pripriity levels.

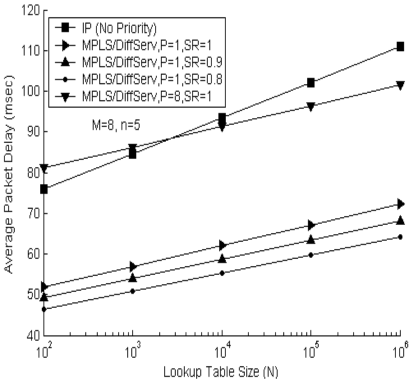


Figure 14: Average Packet Delay for different SR.

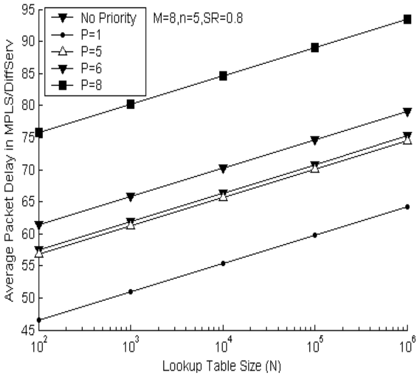


Figure 15: Average Packet Delay for MPLS/DiffServ with different priority levels.

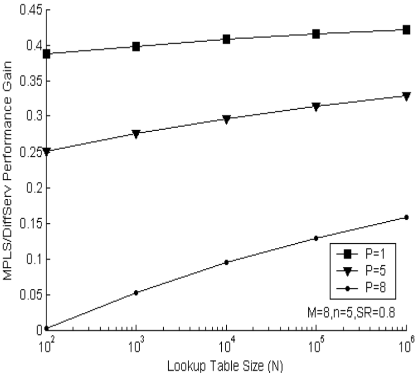


Figure 16: Relative MPLS/DiffSeve performance gain as function of N.

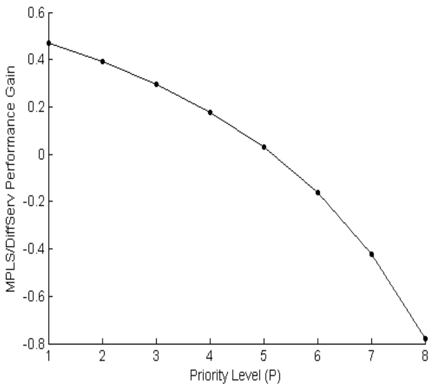


Figure 17: Relative MPLS/DiffServ performance gain for different priority levels.

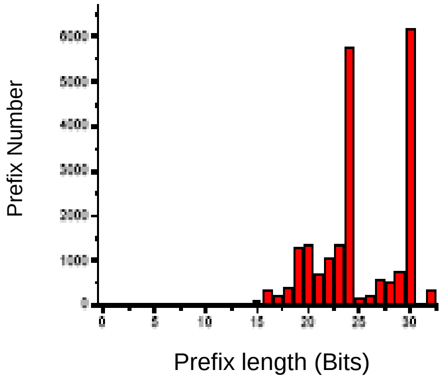


Figure 18: Prefix Length Distribution.

From the figures we can get the following conclusions

1. Packet delay increases linearly as a function of Log (N) in the three cases, IP (No priority), MPLS/DiffServ (P=1, P=8). With the same lookup table size, The average packet delay of any class (P=1 to P=8) is smaller than the original value in IP domain with no priority used, which this demonstrates the improvement possible through the use of MPLS and priority queueing.
2. **Figure 14** shows that the average packet delay for MPLS/DiffServ with different priority is greatly influenced by the parameter SR, which as SR decreases as we get small packet delay.
3. **Figure 15** shows that the average delay of the higher priority packets (P=1) has thus dropped to almost half of the original value in MPLS/DiffServ with no priority used, while the lower priority packets (P=6 to P=8) have larger packet delay than the original value. This demonstrates that we use the classes from (P=1 to P=5) for applications that required high QoS (Small delay) and control packets while the other classes is used for best effort traffic.
4. To evaluate the relative performance of MPLS/DiffServ with priority queueing compared to IP, we define MPLS/DiffServ performance gain to be the ratio of the difference between IP packet delay and MPLS/DiffServ packet delay of specific class divided by IP packet delay. From **Figure 16** we see that the MPLS/DiffServ performance gain increases as the lookup table size grows which indicates that the proposed architecture is more suitable for large size networks, which have large lookup table. We also note that the performance gain is greatly influenced by the priority level. For (P=1), MPLS/DiffServ performance gain is around 42%, for (P=5), it is around 30%, for (P=8), it is around 10%.
5. We also define the MPLS/DiffServ performance gain to be the ratio of the difference between MPLS/DiffServ packet delay with no priority used and MPLS/DiffServ packet delay of specific priority class divided by MPLS/DiffServ

packet delay with no priority used. From **Figure 17** we see that the MPLS/DiffServ performance gain decreases as the priority level decrease from (P=1 to P=8).

5. CONCLUSIONS

In this paper, we reported on a detailed evaluation of the proposed architucre in [1]. Firstly we discuss the proposed QoS routing algorithms, We showed that the overall call rejection probability for the two proposed algorithms are very close to the lower bound provided by the mathematical model. Secondly we inroduce a mathematical model based on M/G/1 with Non-Preemptive Priority Queuing system to measure the average packet delay in IP and MPLS/DiffServ domains. The average packet delay is an estimated value for many parameters in our model are hard to determine, so we depend on the relative evaluation.

Our relative evaluation shows that the average packet delay of high priority classes is much smaller than the original value in IP and MPLS/DiffServ domains with MPLS/DiffServ performance gain equal to 42% or more for higher priority class.

Finally this paper shows that, the proposed architecture in [1] can be used to support multiple number of service classes with reduction in the original IP packet delay by 42% or more with guaranteed end-to-end QoS.

6. REFERENCES

[1] Mahmoud M. Elmesalawy, Ihab A. Ali, Ibrahim I. Ibrahim and Abd Elrahman H. Elsawy, “QoS Enhancement in MPLS-Based IP Networks”, Journal of Engineering Sciences (JES), Faculty of Engineering, Assiut University, PP.553-581, March 2005.

[2] Ibrahim I. Ibrahim, Ihab A. Ali, Abd Elrahman H. Elsawy and Mahmoud M. Elmesalawy, “Performance analysis for quality of service provisining in MPLS/DiffServ-based IP networks”, Journal of Engineering Sciences (JES), Faculty of Engineering, Assiut University, PP.911-928, May 2005.

[3] Shogo Nakazawa, Hitomi Tamura, Kenji Kawahara, Yuji Oie, Performance Analysis for QoS provisining in MPLS Networks.

[4] Guangyi Liu, XioKang Lin, “MPLS Performance Evaluation in Backbone Networks”.

[5] Willibald Doeringer, Giinter Karjoth, and Mehdi Nassehi, “Routing on Longest-Matching Prefixes,”*IEEE/ACM Trans. Networking*, vol. 4, no. 1, pp. 86-97, February 1996.

[6] Hideaki Takagi, *Queueing: A Foundation of Performance Evaluation*, vol. 2. North-Holland, 1993, pp.197-213.

[7] route-server.gblx.net, http://www.traceroute.org.

[8] F. Le Faucheur et al., “Requirements for Support of Differentiated Services-aware MPLS TrafficEngineering” RFC 3564, July 2003.

[9] A. Mankin, F. Baker, R. Braden, S. Bradner, M. O’Dell, A. Romanow, A. Weinrib, and L. Zhang “Resource ReSerVation Protocol (RSVP) Version 1 Applicability Statement – Some Guidelines on Deployment”, Informational RFC 2208, September 1997.

تقييم كفاءة جودة الخدمة في شبكات بروتوكول الإنترنت المعتمدة على نظام تحويل عنونة البروتوكولات المتعددة ونظام الخدمات الفضلة في حالة استخدام الذاكرة التي تعمل بنظام الأولوية

بروتوكول الإنترنت يقدم بعض الخدمات مثل العنونة والتقسيم والتجميع ولكنه يتعامل مع كل التطبيقات بنفس الطريقة حيث أنه يقدم نوع خدمة واحدة "خدمة القدر المستطاع" حيث أنها لا تضمن آي جودة خدمة للتطبيقات الجديدة التي تحتاج إلي ضمان جودة الخدمة. ولذلك نحتاج إلي تقديم أنواع / مستويات مختلفة من الخدمة لإرسال أنواع مختلفة من التطبيقات على شبكة بروتوكول الإنترنت بجودة خدمة مضمونة.

يعتبر نظام تحويل عنونة البروتوكولات المتعددة خطوه انتقالية من بروتوكول الإنترنت ذو السرعة المنخفضة نتيجة تنفيذه في صوره برمجيه إلي التسليم عالي السرعة للبيانات.ومن الممكن ترقيه محول مسارات يعمل بنظام بروتوكول الإنترنت ليتمكن من تعيين مسارات محدده من خلال مفهوم مسار تحويل العنونة المدعم من نظام تحويل عنونة البروتوكولات المتعددة وبالتالي يؤدى إلي تحويل المسارات المحددة.ويعتبر دعم مستويات الخدمة أحد مزايا هذه الترقية.ويمكن وضع مفهوم الخدمات المفضلة المعرفة في الإصدارين الرابع والسادس من بروتوكول الإنترنت في حيز التنفيذ بمطابقته إلي اختيار مسار محدد بين أي نهايتين لكل مستوى خدمه.

ولدراسة التحسن في الأداء الناشئ عن هذه الترقية المقترحة في البحث (1) يقدم البحث (2) طريقه تستخدم أداة محاكاة و تحليل رياضي لتقييم أداء النظام المقترح و ا لوصول إلي مقارنه الأداء في الحالتين ونتائج هذا التقييم هي كما يلي:

الترقية المقترحة في البحث (1) يمكنها تقديم مستويات مختلفة من الخدمة لإرسال أنواع مختلفة من التطبيقات على شبكة بروتوكول الإنترنت بجودة خدمة مضمونة وتقليل في زمن التأخير بين الراسل والمرسل إليه بنسبه تزيد عن 39%.

هذا البحث يهدف الى استخدام الذاكرة التي تعمل بنظام الأولوية لكل مستوى خدمة بدلا من استخدام الذاكرة التي تعمل بنظام الدخل-الأول-الخرج-الأول كالمقترحه فى البحث (2) ولدراسة التحسن في الأداء الناشئ عن استخدام هذا النوع من الذاكرة يقدم هذا البحث طريقه تستخدم أداة محاكاة و تحليل رياضي لتقييم أداء الشبكة وا لوصول إلي مقارنه الأداء في الحالتين ونتائج هذا التقييم هي كما يلي:

الترقية المقترحة في البحث (1) يمكنها تقديم مستويات مختلفة من الخدمة لإرسال أنواع مختلفة من التطبيقات على شبكة بروتوكول الإنترنت بجودة خدمة مضمونة وتقليل في زمن التأخير بين الراسل والمرسل إليه بنسبه تزيد عن 39% في حالة استخدام الذاكرة التي تعمل بنظام الدخل-الأول-الخرج-الأول وأيضا يمكن التقليل في زمن التأخير بنسبه تزيد عن42% في حالة استخدام الذاكرة التي تعمل بنظام الأولوية لكل مستوى خدمة.

وبذلك عن طريق استخدام الذاكرة التي تعمل بنظام الأولوية تم التقليل في زمن التأخير بنسبه تزيد عن3% عنه في حالة استخدام الذاكرة التي تعمل بنظام الدخل-الأول-الخرج-الأول.