CAC and Routing of Self-Similar Call Traffic Based on Markov Decision Theory

Ernst Nordström

Department of Culture/Media/Computer Science, Dalarna University
SE-781 88 Borlänge, Sweden
Email: eno@du.se

Abstract

In this paper we study the call admission control (CAC) and routing issue in multi-service networks. Two categories of calls are considered: a wide-band with delayed call set up and a narrow-band operating in loss mode. The control objective is formulated as reward maximization with penalty for delay. The network is modelled as a Markov Decision Process (MDP) which allows near-optimal CAC and routing decisions in case of Poisson call arrival process and exponentially distributed call holding times. The paper provides an initial analysis of the MDP-method in the context of self-similar call arrival processes, e.g. generated by the World Wide Web Internet service, and exponentially distributed call holding times. Simulation results show that a direct application of the MDP method, based on false Poisson traffic assumptions, yields fairly good performance in terms of average reward loss and average call set up delay. We discuss a possible extension of the traditional MDP method, with potential for even better performance, that incorporates the special features of self-similar call arrival processes. For assessment of the MDP method itself we also provide performance results for the Least Loaded Routing (LLR) method.

1 INTRODUCTION

We consider the problem of Call Admission Control (CAC) and routing in multi-service networks such as ATM and STM networks, and IP networks, provided they are extended with resource reservation capabilities. The objective is to maximize the revenue from carried calls, while meeting constraints on the Quality of Service (QoS) and Grade of Service (GoS) on the packet and call level, respectively. First, the network should determine the set of feasible paths between the source and destination which offers sufficient QoS to the new and existing calls in terms of delay, jitter and data loss. Second, the network should chose to reject the call or to accept it on some path among the set of feasible paths. This choice should be consistent with GoS constraints, in terms of call blocking probabilities and call set up delays, as well as maximizing the average revenue rate for the operator.

Two categories of calls are considered: a narrow-band (NB) requesting a bandwidth of b_n Mbps, and wide-band (WB) requesting b_w Mbps $(b_n < b_w)$. The required bandwidth is represented by the call's peak bandwidth in case of deterministic multiplexing, and by the call's equivalent bandwidth in case of statistical multiplexing. Note that the equivalent bandwidth can be different on different links along the call's path. In particular, the equivalent bandwidth depends on the traffic mix on the link, buffer and link capacity as well as the target buffer overflow probability.

It is well known that when calls are set up on demand, the WB calls can suffer significantly higher rejection rates, compared to NB calls, if there is no additional mechanism to provide access fairness under overload conditions. There exists two main approaches to cope with this fairness problem: access control of NB calls or queuing of WB calls. Trunk reservation is a form of access control which reserves capacity to WB calls by rejecting NB calls when the link occupancy is over a threshold. While access control can deliver good fairness properties, this is usually achieved at the expense of bandwidth utilization.

Queuing of a WB call is done when the network is found busy by the WB call arrival. When a sufficient amount of bandwidth becomes available in the network, a waiting WB call is allowed to enter the network. This approach, if applied correctly, can provide access fairness and increased bandwidth utilization at the same time.

Modern CAC and routing mechanisms are state-dependent rather than static, which means that the decision to reject the request for a new call, or to accept it on a particular path depends on the current occupancy of the network. A state-dependent CAC and routing policy is a mapping, for every call class, from a network state space to a set of possible CAC and routing decisions, see Figure 1. State-dependent mechanisms offer advantages both in terms of achievable revenue and ability to control the QoS and GoS.

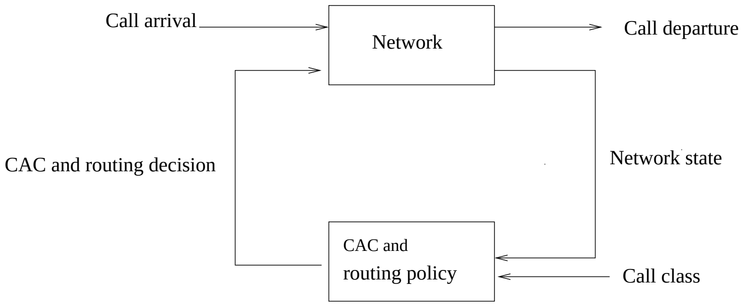


Figure 1: State-dependent CAC and routing

This paper deals with a particular form of state-dependent CAC and routing, where the behavior of the network is formulated as Markov Decision Process (MDP) [3, 13] . A MDP is a controlled Markov process, where the set of state transitions from the current Markov state to other Markov states depends on the decision or action taken by the controller in the current state. In the MDP framework, each call is described by a expected reward parameter and the objective is to maximize the reward from carried calls. The delay of WB calls is taken into account by adding a penalty term to the objective function [4, 6]. The penalty term is a function of the average delay.

Optimal state-dependent CAC and routing policies can be computed using an exact network MDP framework. However, the cardinality of the network state and policy spaces in the exact framework can be very large even for moderate-size networks. Therefore, a necessary modelling simplification is to decompose the network into a set of links assumed to have independent traffic and reward processes, respectively. When formulating the MDP framework for each link, calls with the same bandwidth requirement are aggregated into a common category, which corresponds to one dimension in the link state vector.

Traditionally, the MDP approach to CAC and routing has assumed that calls arrive to the origin-destination (OD) node pair, associated with class j, according to a Poisson process with average rate λ_j [4, 5, 6]. The call holding time is assumed to be exponentially distributed with mean $1/\mu_j$.The Poisson call arrival model is accurate for many service types, such as telephony, FTP transfers, and telnet sessions. However, for the Word Wide Web (WWW) service which contributes to a large portion of the TCP traffic in the Internet, the Poisson model is inadequate. Anja Feldmann has performed measurements on real WWW/TCP connection arrivals in the Internet, and, based on these measurements, she has proposed a certain non-Poisson renewal call arrival process model [7, 8]. This process has inter-arrival times that follow a Weibull distribution in contrast to the Poisson process which has exponentially distributed inter-arrival times. We follow the convention by Anja Feldmann and refer to this new arrival process as self-similar, since it exhibits structural similarities across a wide range of time scales. The Weibull distribution is a generalized exponential distribution. In the case of WWW/TCP connection inter-arrival times in the Internet, the complementary Weibull distribution decays slower (has a more heavy tail) than the standard exponential distribution. Measurements have also shown that the complementary distribution of holding times of WWW connections decays slower than exponentially [2].

In case of Poisson arrival process, and exponential service process, the state transition probabilities, which are part of the MDP model, become easy to formulate. The probability of having a class-j arrival or class-j departure is independent of the time since the last arrival from all the classes. This is not the case if we replace the Poisson process with a non-Poisson process such as the above self-similar process: the probability of the next event being a class-j arrival or class-j departure now becomes dependent on the time since the last arrival from each class.

This paper provides a first step towards understanding MDP-based CAC and routing of self-similar call arrival processes and exponential service times. We provide a numerical/simulation-based evaluation of the MDP control policy, derived with traditional MDP assumptions of Poisson arrivals and exponential service times. We investigate the performance when actual traffic is self-similar and CAC and routing is based on false assumptions of Poisson arrivals. The performance is measured in terms of the average reward loss and average call set up delay. For comparison, the performance of a conventional routing method, the Least Loaded Routing (LLR) method, is also evaluated.

The paper is organized as follows. Section 2 formulates the CAC and routing problem in terms of offered traffic, network and queuing model, and optimization objective. Section 3 describes the Poisson and self-similar call arrival model. Section 4 describes the network and exact link MDP models in case of Poisson call arrivals. Section 5 outlines the MDP computation procedure in case of Poisson call arrivals. Section 6 give a summary of the numerical/simulation-based evaluation of the performance of the traditional MDP and LLR method in case of Poisson and self-similar traffic. Section 7 discuss a possible extension of the traditional MDP model to a MDP model which captures the effects of the self-similar call arrivals. Finally, Section 8 concludes the paper.

2 PROBLEM FORMULATION

2.1 Traffic assumptions

The network is offered traffic from K classes which are, for sake of simplicity, assumed to be subject to deterministic multiplexing. The j-th class, $j \in J = \{1 \cdots K\}$, is characterized by the following [11]:

- origin-destination (OD) node pair,
- peak bandwidth requirement $b_j \in \{b_n, b_w\}$ [Mbps],
- general call arrival process $\{A_n\}$ with two special cases:
  - Poisson process with average arrival rate λ_j [s^{-1}],
  - Self-similar process characterized by Weibull parameters a_j and c_j,
- exponential service process $\{B_n\}$ with mean 1/μ_j [s],
- set of alternative routes, W_j, and
- reward parameter $r_j \in (0, \infty)$

The classes are classified into $G = 2$ bandwidth categories. The i-th category, $i \in I = \{1, \cdots, G\}$, is characterized by:

- peak bandwidth requirement $b_i \in \{b_n, b_w\}$, [Mbps]

- average mean call holding time $1/\overline{\mu}_i$ [s]
- average reward parameter $\overline{r}_i$

2.2 Network and queuing model

The network is assumed to consist of a set of switching nodes, interconnected by bi-directional links according to some network topology. Each bi-directional link consists of two uni-directional links, carrying traffic in opposite directions. Each uni-directional link has one finite FIFO queue for WB call requests. The queues operates as follows [12]. When the path chosen by the CAC and routing algorithm has sufficient available capacity for the new WB call, the call is set up between the considered origin-destination node. Otherwise, at least some link along the path is not able to directly accept the new WB call. At those links, join the new WB call request at the tail of the queue, provided the queue is not held at its maximum. We assume that the path would not be chosen when some of its links has insufficient capacity on both the link and in the queue. At links with sufficient capacity, reserve bandwidth for the new WB call while waiting for all links to be ready to accept the call. A link queue is served when a sufficient number or bandwidth units becomes available on the link. In this case, bandwidth for the WB call at the head of the queue is reserved on the link. When bandwidth has been reserved on every link along the path for a given WB call, the call is set up between the considered origin-destination node.

The advantage of this scheme is its simplicity but its performance may have some drawbacks. One is that the ”reservation” traffic caused by multilink calls due to bandwidth reservation on some links while the call request is in the queue of other links. Although this ”reservation” traffic is likely to be negligible under nominal conditions, it can be significant in case of overloads. Another, less critical, drawback of this solution is the lack of access fairness since the average waiting time of calls offered to multi-link paths is a sum of average waiting times of all queues.

2.3 Objective function

In our loss-delay type of systems we have to deal with bi-objective type function [12]. Let us first define each objective separately. To take into account traffic losses due to rejection of

NB and WB calls we apply reward formulation. In this case the reward from a carried call is defined by the reward rate $q_j = r_j \mu_j$ where r_j, μ_j denotes the reward parameter and the departure rate of the j-th type call, respectively. Now we can define the objective function, $\overline{R}$, as average reward from the network given by

$$\overline{R} = \sum_j r_j \overline{\lambda}_j \tag{1}$$

where $\overline{\lambda}_j$ denotes the j-th class call acceptance rate (the process is assumed to be stationary). The obvious goal of the CAC and routing algorithm is to maximize the objective function. This approach was already applied in state-dependent routing schemes for loss systems presented in [5, 3]. The reward maximization has several advantages from the management point of view since by controlling the reward parameters one can almost independently control the GoS of individual streams (cf. [5, 3]). Another advantage is that by using the control model presented in [5, 3], the objective function can be decomposed as:

$$\overline{R} = \sum_s \overline{R}^s \tag{2}$$

where $\overline{R}^s$ denotes the average reward from the s-th link. Since in our system the NB and WB calls are treated differently it may be convenient to separate the corresponding rewards:

$$\overline{R} = \overline{R}_n + \overline{R}_w = \sum_s [\overline{R}_n^s + \overline{R}_w^s] \tag{3}$$

If we would be concerned only with the delay of WB calls, the natural objective of the CAC and routing algorithm would be to minimize the average delay of calls, $\overline{D}$. For the proposed management of queuing system such an objective function is given by

$$\overline{D} = \sum_s \overline{D}^s \frac{\lambda_w^s}{\lambda_w} \tag{4}$$

where $\overline{D}^s$ denotes the average delay of calls in the s-th link queue and λ_w^s, and λ_w denotes the arrival rate of WB calls offered to the s-th link and the network, respectively.

We select for the objective function a linear combination of the two objective functions, which can also be interpreted as reward maximization with penalty for delay of WB calls:

$$\overline{R}_D = \overline{R} - \alpha \overline{D} \tag{5}$$

where α is the delay penalty weight which determines the trade-off value between the reward and average delay. By using (2),(4) in (5) we arrive at

$$\overline{R}_D = \sum_s [\overline{R}^s - \alpha \overline{D}^s \frac{\lambda_w^s}{\lambda_w}] \tag{6}$$

This form of the objective function illustrates the desired separability of the objective function.

Note that the form of (6) suggests that the delay penalty weight can be also link dependent:

$$\overline{R}_D = \sum_s [\overline{R}^s - \alpha^s \overline{D}^s \frac{\lambda_w^s}{\lambda_w}] \tag{7}$$

This feature gives additional freedom of distributing the delay among the links which may be of importance in practical problems.

3 MODELLING OF CALL ARRIVALS

Since the days of Erlang the Poisson model has commonly been used to describe the random arrivals of call requests to the OD pairs of a telephone network. Although the Poisson model serves its purpose in telephone networks, it lacks descriptive power in the case of Internet where a substantial portion of traffic is World Wide Web (WWW) connections transported by TCP. The nature of the WWW service is different from the telephone service; A person using the WWW service is more likely to initiate additional downloads after the first download. A person using the telephone service is more likely to initiate independent calls.

Measurements on real WWW connection arrivals in the Internet has revealed that the arrival process shows burstiness over many time scales, ranging from seconds to hours. Anja Feldmann [8] argues that this traffic is self-similar. The degree of burstiness over different time scales or the extend of self-similarity can be expressed with just one single parameter, the Hurst parameter. For self-similar processes its value is between 0.5 and 1 and the degree of self-similarity increases as the Hurst parameter approaches 1. More formal, a covariance-stationary process $X = \{X_k : k \geq 1\}$ is called *asymptotically self-similar* (with self-similarity parameter H, $0 < H < 1$), for a large enough m,

$$X_k \overset{d}{=} m^{1-H} X_k^{(m)} \tag{8}$$

where $X^{(m)} = (X_k^{(m)} : k \geq 1)$ is the *aggregated* process of order (time scale) m, given by

$$X_k^{(m)} = \frac{1}{m}(X_{(k-1)m+1} + \cdots + X_{km}), k \geq 1 \tag{9}$$

The process under consideration is the number of call arrivals per time unit.

Anja Feldmann has observed that the WWW connection arrivals can be accurately modeled by a renewal process with inter-arrival times that follow a Weibull distribution, $P(A_n \leq t) = 1 - e^{-\left(\frac{t}{a}\right)^c}$. Recall that the Poisson process has exponentially distributed inter-arrival times: $P(A_n \leq t) = 1 - e^{-\lambda t}$. Hence, the Weibull distribution can be seen as a generalized exponential distribution. When $c = 1$ the Weibull distribution becomes identical to the standard exponential distribution.

The exponential probability density function (pdf) is

$$a(t) = \frac{dP(A_n \leq t)}{dt} = \frac{d}{dt}(1 - e^{-\lambda t}) = \lambda e^{-\lambda t} \tag{10}$$

The mean inter-arrival time for the Poisson process is:

$$E[A_n] = \int_0^\infty ta(t)dt = \int_0^\infty t\lambda e^{-\lambda t}dt = \frac{1}{\lambda} \tag{11}$$

The Weibull pdf is

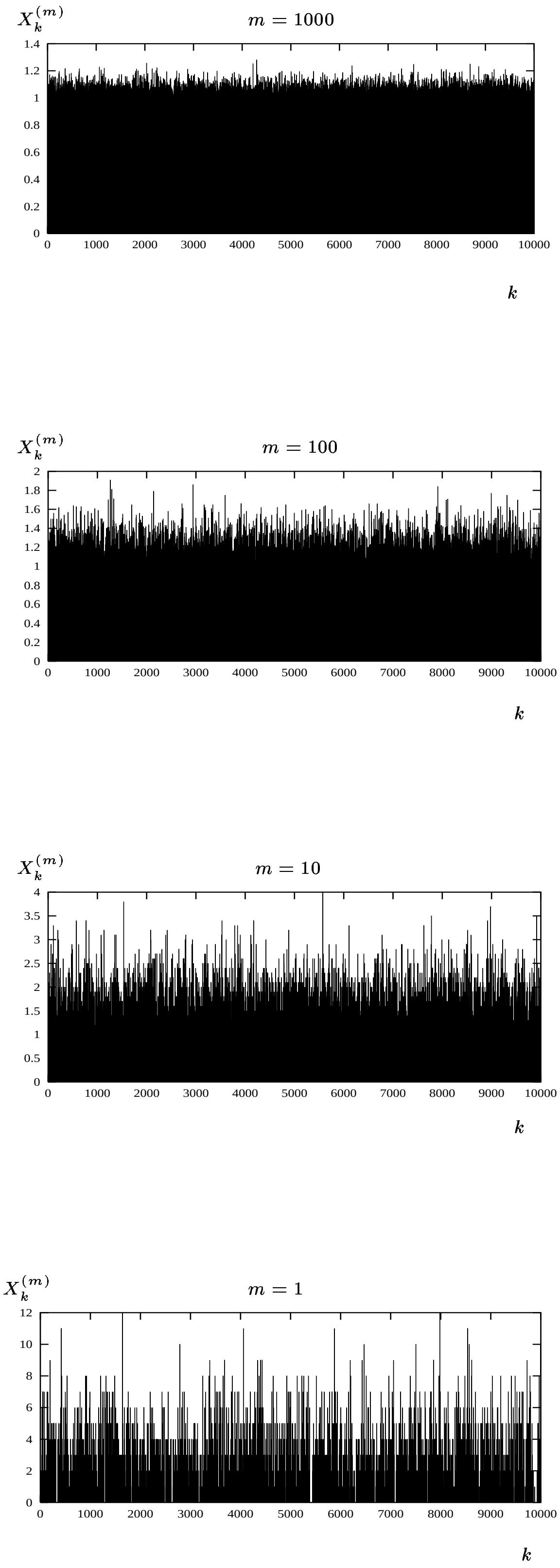
$$a(t) = \frac{dP(A_n \leq t)}{dt} = \frac{d}{dt}(1 - e^{-\left(\frac{t}{a}\right)^c}) = \frac{c}{a}\left(\frac{t}{a}\right)^{c-1} e^{-\left(\frac{t}{a}\right)^c} \tag{12}$$

The mean inter-arrival time for the Weibull-based process is:

$$E[A_n] = \int_0^\infty ta(t)dt = \int_0^\infty t\frac{c}{a}\left(\frac{t}{a}\right)^{c-1} e^{-\left(\frac{t}{a}\right)^c} dt \tag{13}$$

$$\left\{\begin{array}{l} s = \left(\frac{t}{a}\right)^c, t = as^{1/c} \\ ds = c\left(\frac{t}{a}\right)^{c-1} dt \\ dt = \frac{a}{c}s^{\frac{1-c}{c}}ds \end{array}\right\} = c\int_0^\infty se^{-s}\frac{a}{c}s^{\frac{1-c}{c}}ds \tag{14}$$

$$= a\int_0^\infty s^{1/c}e^{-s}ds = a\Gamma(1 + \frac{1}{c}) \tag{15}$$



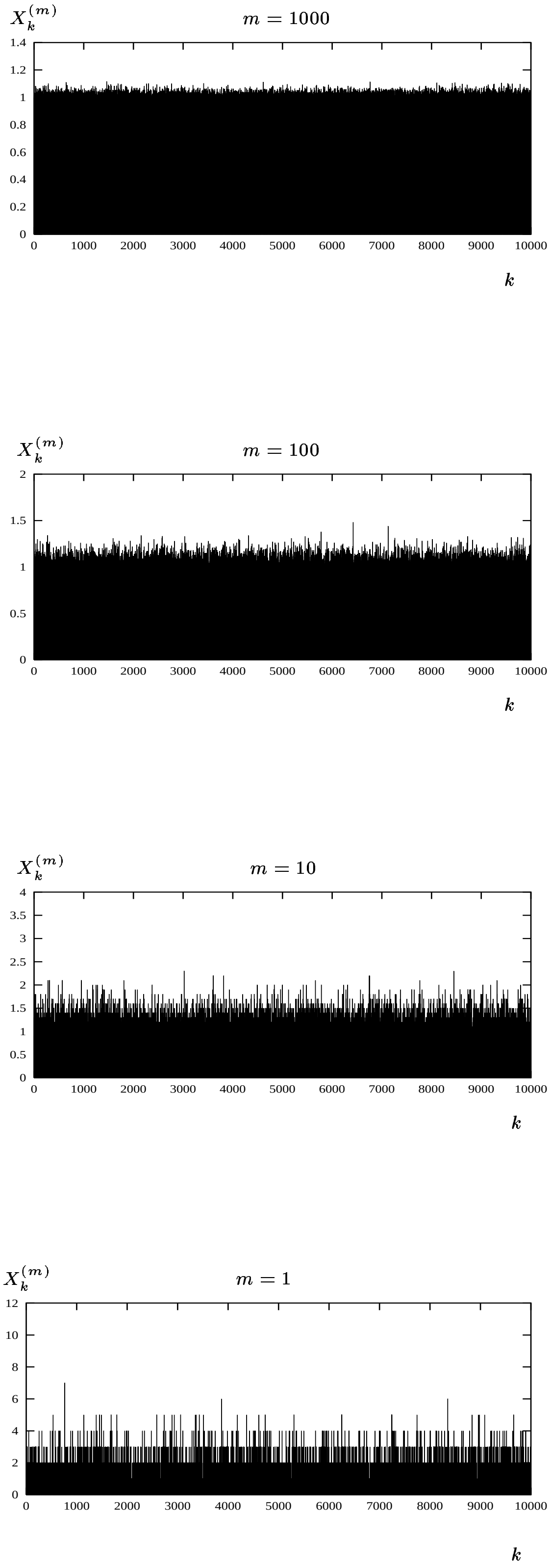


Figure 2: Self-similar call arrivals

Figure 3: Poisson call arrivals

Figure 2 and Figure 3 shows the aggregate arrival process $X_k^{(m)}$ for different time scales m for the self-similar process and the Poisson process. In the Figures, the mean arrival rate λ of the Poisson process where chosen to be 1 [s^{-1}], and the Weibull parameters were $c = 0.5$ and $a = \frac{1}{\Gamma(1+\frac{1}{c})\lambda}$. Obviously, the variability of the aggregate process decreases faster for the Poisson process than the self-similar process when the time scale is increasing.

For a self-similar process the variance of the aggregated process decays like

$$Var[X_k^{(m)}] \backsim m^{-\beta} \tag{16}$$

where $\beta = 2(1 - H)$. The variance-time plot is a popular method for determining β and thus the Hurst parameter $H = 1 - \beta/2$. One simply plots $\log_{10} Var[X_k^{(m)}]$ against $\log_{10}(m)$ and then determines the slope $-\beta$. Figure 4 shows the variance-time plot for the studied Poisson process and the self-similar process. Note that the logarithm of the variance drops with a slope of -1 for Poisson process and with a slope $-\beta, 0 < \beta < 1$, for the self-similar process.

Finally, we show in Figure 5, for the studied self-similar process, the Hurst parameter H as function of the Weibull parameter c. Note that the Hurst parameter does not depend on the Weibull parameter a.

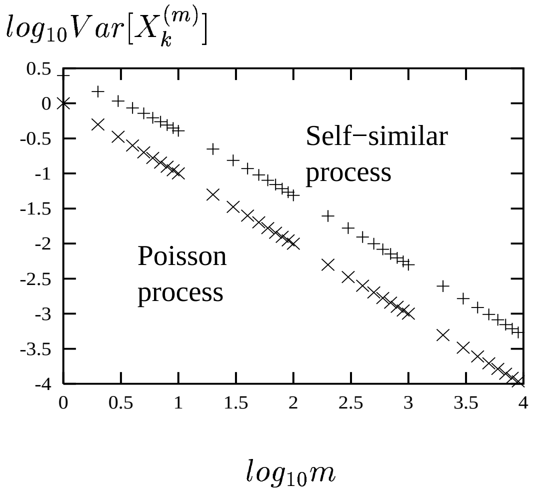


Figure 4: Variance-time plot

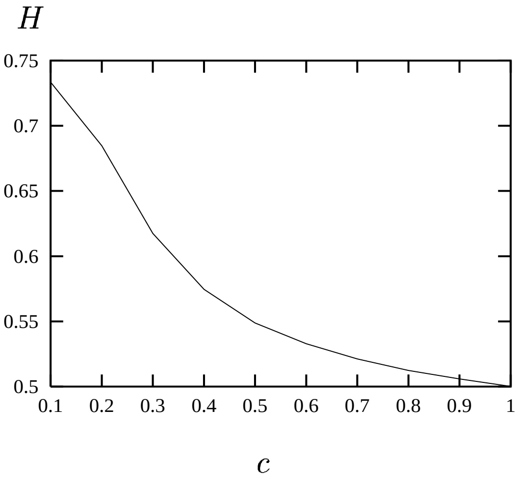


Figure 5: Hurst parameter vs. c

4 MDP MODELLING FOR POISSON CALL ARRIVALS

4.1 Network decomposition

In the exact network MDP model, the network state is given by the matrix $\mathbf{y} = \{x^s\}, s \in S$, where x^s denotes the state of link s and S denotes the set of all link indices in the network. The action space is given by

$$A = \{a = \{a_j\} : a_j \in \{0\} \cup W_j, j \in J\} \tag{17}$$

where $a_j = 0$ denotes call rejection and the set W_j contains the indices of the alternative routes possible for an accepted class j call.

In the exact MDP framework, the network state and the action spaces can be very large, even for moderate-size networks. We therefore decompose the network into a set of links assumed to have independent traffic and reward processes, respectively [5].

The network Markov process is decomposed into a set of independent link Markov processes, driven by state-dependent Poisson call arrival processes with rate $\lambda_j^s(\mathbf{x}, \pi)$, where π denotes the CAC and routing policy. In particular, a call connected on a path consisting of l links is decomposed into l independent link calls characterized by the same mean call holding time as the original call.

The network reward process is decomposed into a set of of separable link reward processes. The link call reward parameters $r_j^s(\pi)$ fulfill the obvious condition that

$$r_j = \sum_{s \in S_k} r_j^s(\pi) \tag{18}$$

where S_k denotes the set of links constituting path k, specified by the routing policy π. Different models for computing link reward parameters are possible [5]. In this paper we use a simple rule: the link reward the call reward is distributed uniformly among the path's links, resulting in the formula $r_j^s(\pi) = r_j/l$, where l denotes the number links in the call's path.

Even in the decomposed model, the state space can be quite large when many call classes share the links. One way to reduce the state space is to construct a modified link reward process in which the link call classes with the same bandwidth requirement are aggregated into one category $i \in I$ with average reward parameter defined as [5]:

$$\overline{r}_i^s(\pi) = \frac{\sum_{j \in J_i} r_j^s(\pi)\overline{\lambda_j^s}(\pi)}{\sum_{j \in J_i} \overline{\lambda_j^s}(\pi)} \tag{19}$$

where J_i denotes the set of classes that belongs to the i-th category, and $\overline{\lambda}_j^s(\pi)$ denotes the average rate of class-j calls accepted on link s. In the following, this simplification is adopted, which reduces the number of effective classes to the number of classes with unique bandwidth requirement.

4.2 Exact link MDP model for Poisson call arrivals

This section describes the exact link MDP model which provides the basis for the MDP computational procedure presented in the next section [11]. In order to simplify notation we assume that alternative routes for category i have no common links – this is not a limitation of the approach. The state in the exact link model is given by $\mathbf{x} = (x_n, x'_w)$, where x_n denotes the number of NB calls on the link, and x'_w denotes the number of WB calls in the system (on the link and in the queue). The state space X for the exact link model is given by:

$$\begin{aligned} X = \{\mathbf{x} = (x_n, x'_w) : 0 \leq x_n \leq N_n^s, 0 \leq x'_w \leq N_w^s + L^s, \\ x_l = f(\mathbf{x}), x_n b_n + (x'_w - x_l) b_w \leq C^s\} \end{aligned} \tag{20}$$

where $N_n^s = \lfloor C^s / b_n \rfloor$, $N_w^s = \lfloor C^s / b_w \rfloor$, and C^s, L^s, denotes the capacity and maximal size of link and queue s, respectively. The number of WB calls in the queue, x_l, is obtained from state $\mathbf{x}$ as follows:

$$x_l = f(\mathbf{x}) := \inf\{x_l : C^s - x_n b_n \geq (x'_w - x_l) b_w\} \tag{21}$$

The Markov decision action a is represented by a vector $a = \{a_i\}, i \in I$, corresponding to admission decisions for presumptive call requests. The action space is given by:

$$A = \{a = \{a_i\} : a_i \in \{0, 1\}, i \in I\} \tag{22}$$

where $a_i = 0$ denotes call rejection and $a_i = 1$ denotes call acceptance. The permissible action space is a state-dependent subset of A:

$$A(x) = \{a \in A : a_i = 0 \text{ if } \mathbf{x} + \delta_i \notin X, i \in I\} \tag{23}$$

where δ_i denotes a vector of zeros except a one in position $i \in I$.

The Markov chain is characterized by state transition probabilities $p_{xy}(a)$ which express the probability that the next state is $\mathbf{y}$, given that action a is taken in state $\mathbf{x}$. In our case, the state transition probabilities become:

$$p_{xy}(a) = \begin{cases} \lambda_i^s(\mathbf{x}, \pi) a_i \tau(\mathbf{x}, a), & \mathbf{y} = \mathbf{x} + \delta_i \in X, i \in I \\ x_n \overline{\mu}_n \tau(\mathbf{x}, a), & \mathbf{y} = \mathbf{x} - \delta_n \in X, \\ (x'_w - x_l) \quad \overline{\mu}_w \tau(\mathbf{x}, a), & \mathbf{y} = \mathbf{x} - \delta_w \in X, \\ 0, & \text{otherwise} \end{cases} \tag{24}$$

where $\lambda_i^s(\mathbf{x}, \pi)$ denotes the i-th category arrival rate to the link in state $\mathbf{x}$ under routing policy π. The parameters $\overline{\mu}_n$, $\overline{\mu}_w$ denotes the departure rate of NB and WB calls, respectively, and $\tau(\mathbf{x}, a)$ denotes the average sojourn time in state $\mathbf{x}$. The link call arrival rates, $\lambda_i^s(\mathbf{x}, \pi)$, are given by:

$$\lambda_i^s(\mathbf{x}, \pi) = \sum_{j \in J_i} \lambda_j^k(\pi) \phi_j^s(\mathbf{x}, \pi) \prod_{c \in S_k \setminus \{s\}} (1 - B_j^c(\pi)), \tag{25}$$

where $s \in S_k$, $B_j^c(\pi)$ denotes the probability that link c has not enough capacity to accept a class j call, and $\phi_j^s(\mathbf{x}, \pi)$ denotes a filtering probability defined as:

$$\phi_j^s(\mathbf{x}, \pi) = P \left\{ \sum_{c \in S_k \setminus \{s\}} p_j^c(\mathbf{x}, \pi) < r_j - p_j^s(\mathbf{x}, \pi) | \overline{B}_j \right\} \tag{26}$$

where $\overline{B}_j$ denotes the condition that no link on path k is in the blocking state (note that $p_j^s(\mathbf{x}, \pi)$ is constant in (26)). In other words $\phi_j^s(\mathbf{x}, \pi)$ is the probability that the path net-gain is positive (on condition that there is enough path capacity to carry the call). The filtering probability can be computed using link state distributions, or approximated with one according to experiments in [5]. The $\lambda_j^k(\pi)$ denote the arrival rate of class j to path $k \in W_j$, and is given the following load sharing model [5]:

$$\lambda_j^k(\pi) = \lambda_j \frac{\overline{\lambda}_j^k(\pi)}{\sum_{h \in W_j} \overline{\lambda}_j^h(\pi)} \tag{27}$$

where the $\overline{\lambda}_j^k(\pi)$ denotes the average rate of accepted class j calls on path k, and λ_j denotes the arrival rate of class j.

The average departure rate for the NB category is computed as:

$$\overline{\mu}_n = \left[\sum_{j\in J_n} p_j^n \mu_j^{-1}\right]^{-1} \tag{28}$$

where p_j^n denotes the probability that an arbitrary NB call found on the link is from class $j \in J_n$:

$$p_j^n = \frac{\overline{\lambda}_j^s(\pi)}{\sum_{c\in J_n} \overline{\lambda}_c^s(\pi)} \tag{29}$$

where $\overline{\lambda}_j^s(\pi)$ denotes the average rate of accepted class j calls on link s.

The average departure rate for the WB category is computed as:

$$\overline{\mu}_w = \left[\sum_{j\in J_w} p_j^w \mu_j^{-1} + T^s\right]^{-1} \tag{30}$$

where T^s denotes the average reservation time for WB calls on link s (T^s can be obtained from measurements), and p_j^w denotes the probability that an arbitrary WB call found on the link is from class $j \in J_w$.

The average sojourn time $\tau(\mathbf{x}, a)$ in state $\mathbf{x}$ is given by:

$$\tau(\mathbf{x}, a) = \left\{\sum_{i\in I} x_i \overline{\mu}_i + a_i \lambda_i^s(\mathbf{x}, \pi)\right\}^{-1} \tag{31}$$

The expected reward in state $\mathbf{x}$ is given by $R_D^s(\mathbf{x}, a) = q^s(\mathbf{x})\tau(\mathbf{x}, a)$, where $q^s(\mathbf{x})$ is obtained from

$$q^s(\mathbf{x}) = \overline{r}_n^s x_n \overline{\mu}_n + \overline{r}_w^s (x_w' - x_l)\overline{\mu}_w - \alpha^s \frac{x_l}{\lambda_w} \tag{32}$$

5 MDP COMPUTATIONAL PROCEDURE FOR POISSON CALL ARRIVALS

This section outlines the MDP computational procedure for determining a near-optimal CAC and routing policy using the exact link model [11]. The central idea is to compute *path net-gain* functions, $g_j^k(\mathbf{y}, \pi)$, which estimate the increase in long-term reward due to admission of a class j call on path k in network state $\mathbf{y}$. The CAC and routing rule is simply to chose, given the state of the network and the class of the call request, a path which offer maximal positive path net-gain among the paths with sufficient QoS (See Figure 6). The call is rejected if the path net-gain is negative, or if no path would offer sufficient QoS.

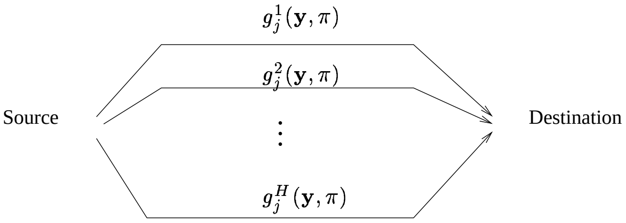


Figure 6: The call is offered to a path which has sufficient QoS and maximal positive path net-gain among the $H = |W_j|$ alternative paths.

The path net-gain is defined as:

$$g_j^k(\mathbf{y},\pi) = r_j - \sum_{s \in S_k} p_i^s(\mathbf{x},\pi) \tag{33}$$

where $\mathbf{y} = \{\mathbf{x}\}$ denotes the network state in the exact network model, and $p_i^s(\mathbf{x},\pi)$ denotes the state-dependent *link shadow price* for category i on link s. The link shadow price can be interpreted as the expected cost for accepting an i-th category call in state $\mathbf{x}$ and is defined as follows:

$$p_i^s(\mathbf{x},\pi) = \overline{r}_i^s(\pi) - g_i^s(\mathbf{x},\pi) \tag{34}$$

where $g_i^s(\mathbf{x},\pi)$ denotes the *link net-gain* for admission of a category-i call in state $\mathbf{x}$. The link net-gain expresses the increase in long-term reward due to admission of a category-i call in link state $\mathbf{x}$ and is defined as follows:

$$g_i^s(\mathbf{x},\pi) = v^s(\mathbf{x}+\delta_i,\pi) - v^s(\mathbf{x},\pi) \tag{35}$$

where $v^s(\mathbf{x},\pi)$ denotes the *relative value* for category i in state $\mathbf{x}$ and δ_i denotes a vector of zeros except for a one in position i.

To give more insight into the definition of relative values, let us define the expected link reward, $R_D^s(x_0,\pi,T)$, obtained in a interval (t_0, t_0+T) of length T, assuming state x_0 at time t_0:

$$R_D^s(x_0, \pi, T) = E\left[\int_{t_0}^{t_0+T} q^s(\mathbf{x}(t))dt\right] \qquad (36)$$

where $q^s(\mathbf{x}(t))$ denotes the expected reward accumulation rate in state $\mathbf{x}(t)$. The process $\{\mathbf{x}(t)\}$ is driven by a probabilistic law of motion specified by certain state transition probabilities.

The relative value can now be written as:

$$v^s(\mathbf{x}_0, \pi) = \lim_{T\to\infty} [R_D^s(\mathbf{x}_0, \pi, T) - R_D^s(\mathbf{x}_r, \pi, T)] \qquad (37)$$

That is, the relative value in state $\mathbf{x}_0$ is defined as the difference in future reward earnings when starting in the given state, compared to a reference state, $\mathbf{x_r}$. In practice, the relative value function is obtained by solving a set of linear equations (see below).

In the context of the exact link MDP model outlined in the previous section, the algorithm for determining the near-optimal CAC and routing policy π can be summarized as follows:

1. **Startup**: Initialize the relative values $v^s(\mathbf{x}, \pi)$ in a way that make all link net-gains with permissible admission positive.

2. **On-line operation phase**: measure per-path call acceptance rates $\overline{\lambda}_j^k(\pi)$ and per-link blocking probabilities $B_j^c(\pi)$ while employing the maximum path net-gain routing rule. Perform the measurements for a sufficiently long period for the system to attain statistical equilibrium.

3. **Policy iteration cycle**: At the end of the measurement period, perform the following steps for all links s in the network:

   (a) **Identify the link MDP model**: Determine per-category reward parameters $\overline{r}_i^s(\pi)$ and link call arrival rates $\lambda_i^s(\mathbf{x}, \pi)$

   (b) **Value determination**: Find the relative values $v^s(\mathbf{x}, \pi)$ for the current routing policy π

   (c) **Policy improvement**: Improve the link CAC policies π_s based on the new relative values

4. **Convergence test**: Repeat from 2 until average reward per time unit converges.

According to MDP theory an optimal policy is found after a finite number of policy iterations in case of a finite state and policy space [13].

The *value determination step* for link s determines the relative values $v^s(\mathbf{x},\pi)$ for all states $\mathbf{x} \in X$ by solving a sparse system of linear equations:

$$\begin{cases} v^s(\mathbf{x},\pi) = R_D^s(\mathbf{x},a) - \overline{R}_D^s(\pi)\tau(\mathbf{x},a) + \sum_{\mathbf{y}\in X} p_{xy}(a)v^s(\mathbf{y},\pi) \\ v^s(\mathbf{x}_r,\pi) = 0; \qquad \mathbf{x} \in X \setminus \{\mathbf{x}_r\}, \end{cases} \tag{38}$$

where the following quantities need to be specified:

- X: the state space, i.e. the set of possible states
- $a = \pi_s(\mathbf{x})$: the control action in state $\mathbf{x}$
- $\tau(\mathbf{x},a)$: the expected sojourn time in state $\mathbf{x}$
- $R_D^s(\mathbf{x},a)$: the expected link reward when leaving state $\mathbf{x}$
- $p_{xy}(a)$: the transition probability from state $\mathbf{x}$ to state state $\mathbf{y}$, given that action a is taken in state state $\mathbf{x}$
- $\mathbf{x}_r$: the reference state (e.g. the empty state)

in order to compute the unknowns:

- $v^s(\mathbf{x},\pi)$: the relative value in state $\mathbf{x}$ under routing policy π
- $\overline{R}_D^s(\pi)$: the average rate of link reward under policy π.

The computation (time) complexity of the value determination step of policy iteration is a function of the size, S, of the state space. Traditional Gauss elimination has complexity O(S^3). This can be seen as an upper limit of the actual complexity since the system is sparse and more efficient iterative algorithms can be used.

The *policy improvement step* for link s consists of finding the action that maximizes the relative value in each state $\mathbf{x} \in X$:

$$a = \underset{a\in A(\mathbf{x})}{\mathsf{argmax}} \left\{ R_D^s(\mathbf{x},a) - \overline{R}_D{}^s(\pi)\tau(\mathbf{x},a) + \sum_{y\in X} p_{xy}(a)v^s(\mathbf{y},\pi) \right\} \tag{39}$$

where $A(\mathbf{x})$ denotes the set of possible actions in state $\mathbf{x}$. The set of actions which yields the maximum improvement of relative values constitute an improved policy π_s' to be used again in the first step. After some simplification, the policy improvement step can be transformed into:

$$a = \underset{u \in A(\mathbf{x})}{\mathsf{argmax}} \left\{ \sum_{i \in I} u_i g_i^s(\mathbf{x}, \pi) \right\}, \mathbf{x} \in X \tag{40}$$

where $g_i^s(\mathbf{x}, \pi)$ is the link net-gain for category i in state $\mathbf{x}$. The policy improvement step has complexity O$(4GS)$, where G denotes the number of unique bandwidth categories.

6 NUMERICAL RESULTS

6.1 Considered routing algorithms

The routing algorithms that have been considered in the numerical experiments is the MDP routing algorithm [6], and the Least Loaded Routing (LLR) algorithm [3]. The reason for choosing LLR is that it is among the routing methods with best performance [1, 3]. The LLR routing method is implemented in many countries, including USA, Canada and Norway. The MDP routing method is less widely employed; We are only aware of one implementation, namely in the Bell Canada network.

The LLR algorithm works as follows. When a class-j call request is received the set of shortest paths W_j^n with n links is considered first. Among this set, a path with largest free capacity of the bottleneck link greater or equal than the bandwidth requirement b_j and greater than the trunk reservation value θ_j^n is searched for. The bottleneck link is the link with least free capacity along the path. Note that a unique trunk reservation value is used for every call from class j that is offered to the set of shortest paths of length n. In case we have a WB call request, and all paths have insufficient free bandwidth, we offer the WB call to the path with smallest maximal queue length along the path. If all paths have insufficient free bandwidth (and insufficient queue space for a WB call), the call is offered the next set of (longer) shortest paths, and the routing procedure is repeated. The procedure stops when a feasible least loaded path among the set of shortest paths is found, or when all paths between the OD pair block the call.

6.2 Examples and results

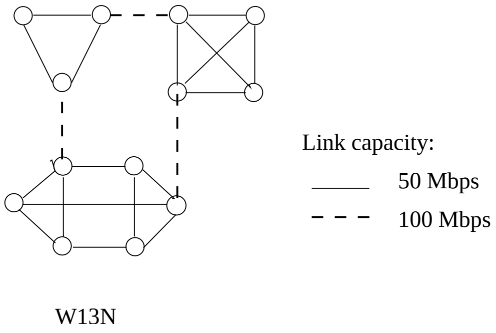


Figure 7: Network example W13N

The performance analysis is performed for the network example W13N described in Table 1. The topology of W13N, shown in Figure 7, is based on an example in [9]. The algorithm specific parameter settings, presented in Table 2, were determined heuristically based on simulation experience. The network is offered the same amount of traffic in Mbps*Erlang for each OD pair.

Two types of call arrival models are used the the simulations: the Poisson model (po) and the self-similar model (ss). The mean Poisson arrival rate λ_j and the mean holding time $1/\mu_j$ are first chosen for every class $j \in J$. The self-similar call arrival process is chosen as follows. First, the Weibull parameter c_j for $j \in J$ is set to $c_j = 0.5$ which is plausible value for real WWW connection arrivals [8]. The choice $c_j = 0.5$ gives a Hurst parameter of $H = 0.55$, see Figure 5. Second, the Weibull parameter a_j for $j \in J$ is set to $a_j = \frac{1}{\Gamma(1+\frac{1}{c_j})\lambda_j}$ which gives equal mean inter-arrival time for the Poisson and self-similar process.

Each curve in the diagrams contains N simulation points, $\overline{x}_k, k = 1, \cdots, N$, which are obtained as averages over M simulation runs per point: $\overline{x}_k = \frac{1}{M}\sum_{i=1}^{M} x_{ik}$. For assessment of the accuracy of the simulation results we present values of the pooled standard deviation of simulation results in Tables 3, 4, and 5, respectively. We compute the pooled standard deviation over the N simulation points as:

$$s = \sqrt{\frac{1}{N}\sum_{k=1}^{N} s_k^2} \tag{41}$$

where s_k^2 denotes the sample variance of the value of point k:

	W13N
symmetrical	no
#nodes	13
#bi-directional links	21
#OD pairs	156
#routes per OD pair	1-7
link capacity	50, 100
network capacity	2400
maximal queue size L^s	3
max #links in path	4
#traffic categories	2
mean holding time	1, 10
bandwidth b_i	1, 6
network traffic [Mbps*Erlang]	624
traffic per OD pair [Mbps*Erlang]	4.0
$r'_j = r_j \mu_j / b_j$	1

Table 1: Description of network example

	W13N
MDP adaptation epochs	6
call events in adaptation period	500 000
call events in measurement period	1000 000
#simulation points per curve N	4,19
#simulation runs per point M	20
delay penalty weight α^s	200
trunk reservation parameter θ^n_j	0

Table 2: Algorithm specific parameters

	Pooled reward loss sdev (%) ($L^s = 0$)	Pooled reward loss sdev (%) ($L^s = 3$)	Pooled average delay sdev (ms) ($L^s = 3$)	Pooled objective reward loss sdev (%) ($L^s = 3$)
MDP po	0.02	0.03	0.2	0.04
MDP ss	0.07	0.05	0.3	0.05
LLR po	0.03	0.02	0.3	0.03
LLR ss	0.07	0.06	0.3	0.08

Table 3: Pooled standard deviation in simulations with variable traffic mix

	Pooled reward loss sdev (%)	Pooled average delay sdev (ms)	Pooled objective reward loss sdev (%)	Pooled NB blocking sdev (%)	Pooled WB blocking sdev (%)
MDP po	0.10	1.2	0.13	0.18	0.02
MDP ss	0.15	1.7	0.21	0.26	0.08
LLR po	0.13	5.8	0.30	0.26	0.01
LLR ss	0.23	11.7	0.56	0.48	0.04

Table 4: Pooled standard deviation in simulations with variable maximal queue length

$$s_k^2 = \frac{1}{M-1}\sum_{i=1}^{M}(x_{ik} - \overline{x}_k)^2 \qquad (42)$$

The first set of figures shows the routing performance in network example W13N as a function of the traffic mix. We consider both the loss network case and the mixed loss-delay network case. In the latter case, we assume a delay penalty weight α^s of 200 and a maximal queue size L^s of 3. The mix of OD-pair traffic is measured by the ratio $b_n\lambda_n\mu_n^{-1}/b_w\lambda_w\mu_w^{-1}$. Different mixes are obtained by varying the per-category call arrival rate to the OD pairs between the simulations, while keeping the amount of traffic per OD pair constant. All OD pairs were offered the same per-category call arrival rates within a simulation. Figure 8-9, 10, 11 and show the reward loss, average call set up delay, objective reward loss, respectively, as a function of the traffic mix.

The second set of figures shows the routing performance in network example W13N as

	Pooled reward loss sdev (%)	Pooled average delay sdev (ms)	Pooled objective reward loss sdev (%)	Pooled NB blocking sdev (%)	Pooled WB blocking sdev (%)
MDP po	0.12	1.5	0.21	0.15	0.09
MDP ss	0.08	0.6	0.11	0.11	0.08
LLR po	0.09	6.4	0.50	0.17	0.01
LLR ss	0.09	7.3	0.54	0.17	0.02

Table 5: Pooled standard deviation in simulations with variable delay penalty weight

function of the maximal queue size, assuming a traffic mix of 1 for each OD pair, and a delay penalty weight α^s of 200. Figure 12, 13, 14 and 15 show the reward loss, average call set up delay, objective reward loss, per-category call blocking probability, respectively, as a function of the maximal queue size.

The third set of figures shows the routing performance in network example W13N as a function of the delay penalty weight α^s, assuming a maximal queue size L^s dvipof 3 and a traffic mix of 1. Figure 16, 17, 18 and 19 show the reward loss, average call set up delay, objective reward loss and per-category call blocking probability,respectively, as a function of the delay penalty weight.

The reward loss, average call set up delay and objective reward loss, respectively, are given by:

$$L = 1 - \overline{R}/R, \tag{43}$$

$$\overline{D} = \sum_s \overline{D}_s \frac{\lambda_w^s}{\lambda_w}, \tag{44}$$

$$L_D = 1 - \overline{R}_D/R. \tag{45}$$

6.3 Results Analysis

The first set of figures shows the routing performance for different traffic mixes, both for a loss network and a mixed loss-delay network. The following conclusions are drawn:

- The best reward loss, in both the pure loss network and the mixed loss-delay network case, in both the Poisson and self-similar traffic case, is obtained for the MDP algorithm (Figure 8 and 9).

- The reward loss is higher in the self-similar case can than in the Poisson case (Figure 8 and 9).

- The average delay can be quite large for the LLR algorithm when WB traffic dominates (Figure 10).

- The best overall behavior for the mixed loss-delay network, in terms of the objective function, is obtained for MDP method for both the Poisson and self-similar traffic case (Figure 11).

The second set of figures show the routing performance for different maximal queue sizes. The following conclusions are drawn:

- Call queuing reduces the reward loss (Figure 12).

- The average call set up delay increases when the maximal queue size increases (Figure 13).

- The best overall behavior in terms of the objective function is obtained for MDP algorithm for both the Poisson and self-similar traffic case (Figure 14).

- The NB blocking probability increases, and the WB blocking probability decreases, with increasing maximal queue size (Figure 15).

The third set of figures show the routing performance as a function of the delay penalty weight. The following conclusions are drawn for MDP routing:

- The reward loss increases with the delay penalty weight α^s (Figure 16).

- The average call set up delay decreases with the delay penalty weight α^s (Figure 17).

- The objective reward loss increases with the delay penalty weight α^s (Figure 18).

- The NB call blocking probability is not much changed by increasing delay penalty weight α^s (Figure 19).

- The WB call blocking probability increases with increasing delay penalty weight α^s (Figure 19).

The following conclusions are drawn for LLR routing:

- The reward loss is independent of the delay penalty weight α^s (Figure 16).
- The average call set up delay is independent of the delay penalty weight α^s (Figure 17).
- The objective reward loss increases with the delay penalty weight α^s (Figure 18).

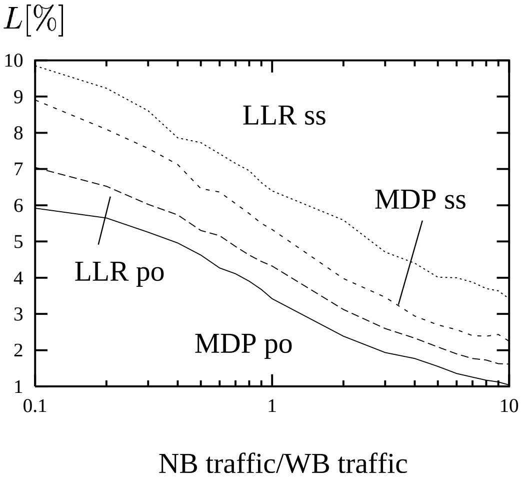


Figure 8: Reward loss versus traffic mix for network W13N with $L^s = 0$

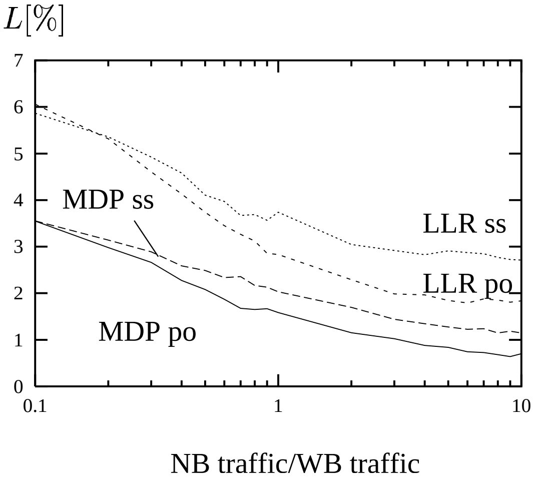
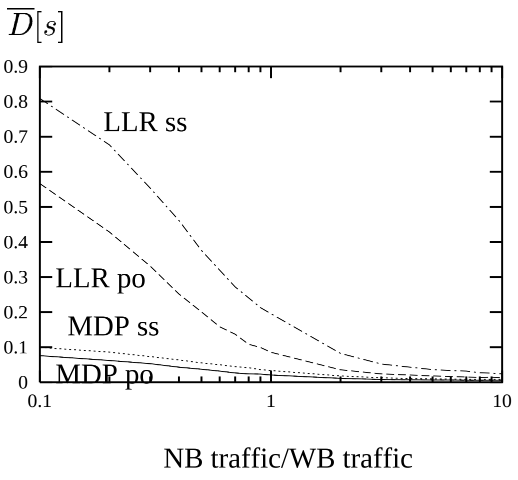


Figure 9: Reward loss versus traffic mix for network W13N with $L^s = 3$



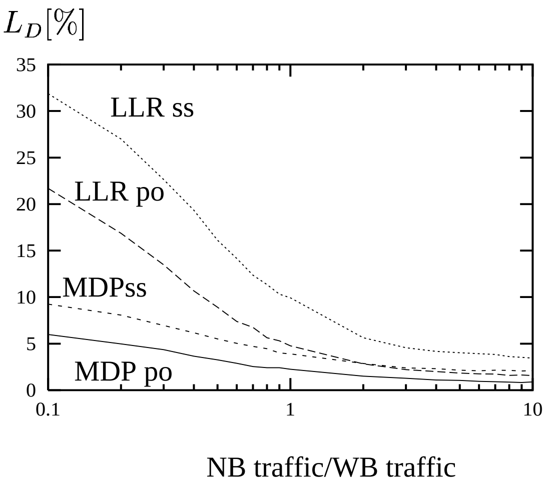
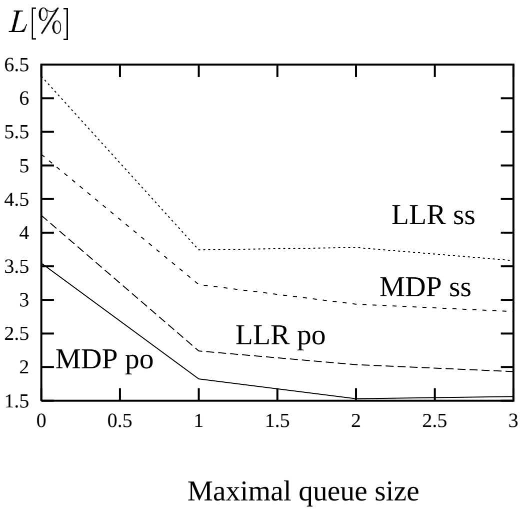


Figure 10: Average call set up delay versus traffic mix for network W13N with $L^s = 3$

Figure 11: Objective reward loss versus traffic mix for network W13N with $L^s = 3$



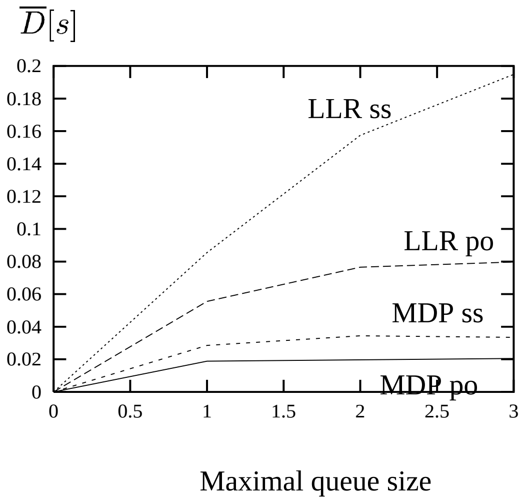
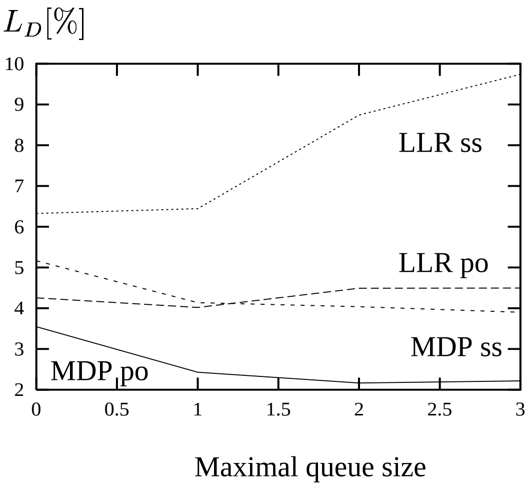


Figure 12: Reward loss as versus maximal queue size for network W13N

Figure 13: Average delay versus maximal queue size for network W13N



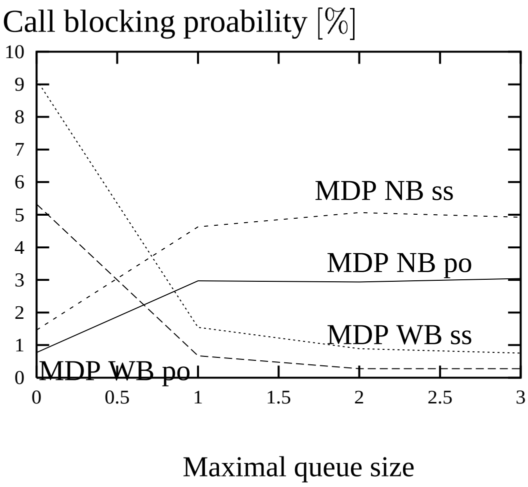
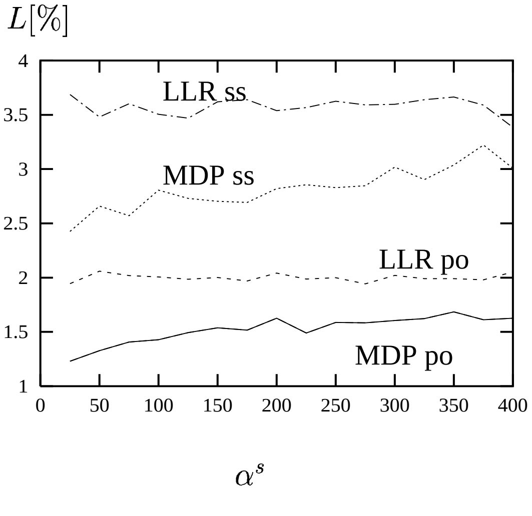


Figure 14: Objective reward loss versus maximal queue size for network W13N

Figure 15: NB and WB call blocking probability for network W13N



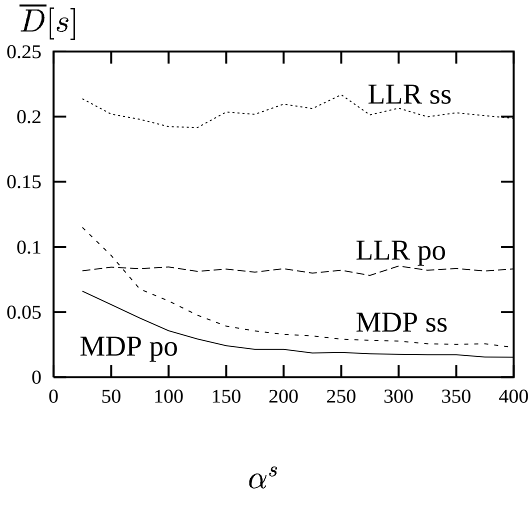
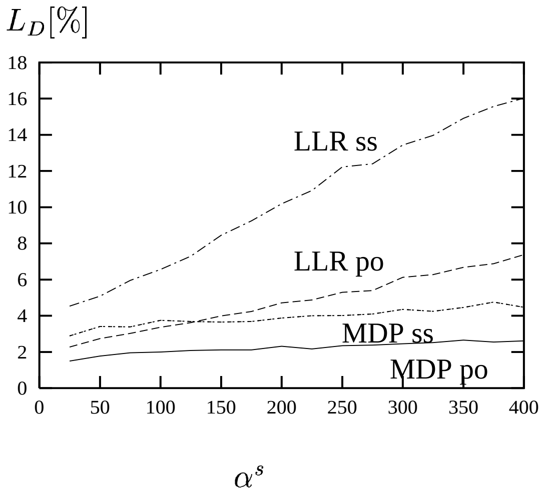


Figure 16: Reward loss versus delay penalty weight for network W13N

Figure 17: Average call set up delay versus delay penalty weight for network W13N



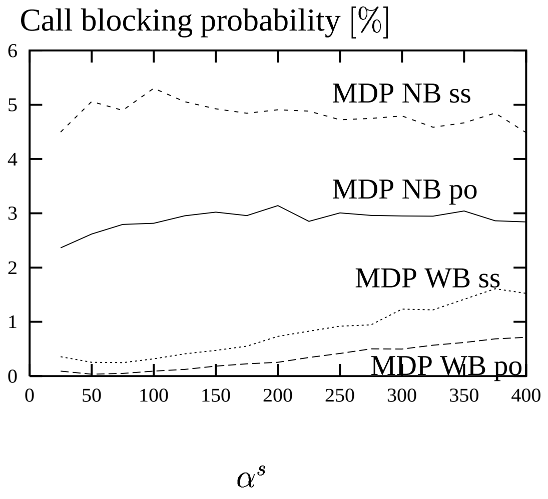


Figure 18: Objective reward loss versus delay penalty weight for network W13N

Figure 19: NB and WB call blocking probability for network W13N

7 DISCUSSION

In this section we discuss the requirements on a MDP model which captures the special features of the self-similar call arrival process. For simplicity, lets consider a single link which is offered one call class with self-similar arrival process and exponential call holding times. We observe that only when the inter-arrival times are exponentially distributed we have a memoryless arrival process (Poisson). Memoryless means that the probability of the next event being an arrival, for example, is independent of the time since the last arrival.

In all other cases, including the self-similar Weibullian case, the process has memory. That is, the probability of the next event being an arrival depends on the time since the last arrival.

It therefore seems reasonable that the MDP state should contain both the number of calls n on the link, and the time t since the last arrival. The choice of the augmented MDP state $x = (n, t)$ implies a new MDP model (state transition probabilities and expected reward), and a new set of time-dependent relative value functions, $v(n, t, \pi)$. In a future publication we will derive the exact formulas for the MDP model and relative value functions. Once we have determined the relative value functions, we obtain the link admission gain function $g(n, t, \pi)$ as:

$$g(n,t,\pi) = v(n+1,0,\pi) - v(n,t,\pi) \tag{46}$$

which is the difference between the relative value of the state $y = (n+1, 0)$ entered after admission, and the relative value in the current state $x = (n, t)$ before admission.

8 CONCLUSION

In this paper we formulated the CAC and routing problem in mixed loss-delay mode multi-service networks as a reward maximization problem with penalty for WB call set up delay. In this formulation each call class is characterized by its reward parameter defining the expected reward for carrying a call from this class. Such a formulation allows to apply Markov Decision Process (MDP) theory to solve the problem. To make the solution feasible we decomposed the network into a set of links assumed to have independent traffic and reward processes, respectively.

Traditionally, the MDP approach to CAC and routing has assumed Poisson call arrivals and exponentially distributed call holding times. These assumptions are valid for services like telephony, FTP and telnet. However, they become inaccurate for services like World Wide Web (WWW), which contributes to a large portion of the TCP traffic in the Internet.

In particular, the measurements on real WWW/TCP connection arrivals in the Internet have revealed that the arrival process is self-similar and that call holding time distribution is more heavy tailed than the standard exponential distribution.

In this paper have investigated the performance of the traditional MDP method for CAC and routing [6, 11] for two call arrival models: the Poisson model and the self-similar model. For simplicity, we have only considered exponentially distributed call holding times. The self-similar model of choice is a renewal process with inter-arrival times that follow a Weibull distribution [8].

The simulation results show that the CAC and routing performance is better for the MDP method than for the Least Loaded Routing (LLR) method (e.g. up to 80% better in case of reward loss), even in the case when the network is offered self-similar call traffic, and routing decisions are based on false assumptions of Poisson call arrivals.

We believe that the performance of the MDP method can be improved further by incorporating the special features of the self-similar call arrival process into the MDP model. An extended MDP model with this capability is part of future work.

References

[1] Ash G., *Dynamic Routing in Telecommunication Networks*, McGraw-Hill, 1998.

[2] Crovella M., Bestavros A., “Self-similarity in world wide web traffic – evidence and possible causes”, *IEEE/ACM Transactions on Networking*, Vol. 5, No. 6, pp. 835-846, 1997.

[3] Dziong Z., *ATM Network Resource Management*. McGraw-Hill (ISBN 0-07-018546-8), 1997.

[4] Dziong Z., Liao K-Q., Mason L.G., “Flow Control Models for Multi-Service Networks with Delayed Call Set Up”, In *Proceedings of IEEE INFOCOM’90,*, pp. 39-46, IEEE Computer Society Press, 1990.

[5] Dziong Z., Mason L., “Call Admission and Routing in Multi-Service Loss Networks”, *IEEE Trans. on Commun.*, Vol. 42, No. 2, 1994.

[6] Dziong Z., Nordström E., “CAC and Routing for Multi-Service Networks with Delayed Set Up of Wide-band Calls – Markov Decision Theory Framework”, submitted, 2002.

[7] Feldmann A., “Impact of non-poisson arrival sequences for call admission control algorithms with and without delay”, In *Proceedings of IEEE GLOBECOM*, pp. 617-622, Nov. 1996.

[8] Feldmann A., “Characteristics of TCP connection arrivals”, Tech. Rep., AT&T Labs Research, 1998.

[9] Ma Q., Steenkiste P., Zhang H., “Routing High-bandwidth Traffic in Max-min Fair Share Networks, In *Proceeding of ACM SIGCOMM’96*, pp. 206-217, Stanford, CA, USA, 1996.

[10] Nordström E., “Near-Optimal Link Allocation of Blockable Narrow-band and Queueable Wide-Band Call Traffic in ATM Networks”, In *Proceedings of ITC 15*, Washington, 1997.

[11] Nordström E., “Near-Optimal CAC and routing for multi-service networks by Markov decision theory”, In *Proceedings of KK foundation conference*, Skövde, Sweden, April, 2002.

[12] Nordström E., Carlström J., “Call Admission Control and Routing in Multi-Service Networks by Markov Decision Theory with State Aggregation”, submitted, 2002.

[13] Tijms H., *Stochastic Modeling and Analysis – a Computational Approch*, Wiley, 1986.