Technical Report
Quality of Service Routing in MPLS Networks using Decentralized Learning

Fariba Heidari, Shie Mannor and Lorne G. Mason

Department of Electrical and Computer Engineering,

McGill University, Montreal, Quebec, Canada

fariba.heidari@mail.mcgill.ca, shie.mannor@mcgill.ca,

lorne.mason@mcgill.ca

Abstract. This paper presents several decentralized learning algorithms for on-line intra-domain routing of bandwidth guaranteed paths in MPLS networks when there is no a-priori knowledge of traffic demand. The presented routing algorithms use only their locally observed events and update their routing policy using learning schemes. The employed learning algorithms are either learning automata or the multi-armed bandit algorithms. We investigate the asymptotic behavior of the proposed routing algorithms and prove the convergence of one of them to the user equilibrium. Discrete event simulation results show the merit of these algorithms in terms of increasing the network admissibility compared with shortest path routing. We investigate the performance degradation due to decentralized routing as opposed to centralized optimal routing policies in practical scenarios. The system optimal and the Nash bargaining solutions are two centralized benchmarks used in this study. We provide nonlinear programming formulations of these problems along with a distributed recursive approach to compute the solutions. An on-line partially-decentralized control architecture is also proposed to achieve the system optimal and the Nash bargaining solution performances. The results of this study indicate that decentralized learning techniques provide efficient, stable and scalable approaches for routing the bandwidth guaranteed paths.

1 Introduction

On-line routing of bandwidth guaranteed paths plays an important role in on-line Quality of Service (QoS) provisioning. Other QoS metrics such as delay or loss can be converted into effective bandwidth requirements [1] and QoS constraints expressed as delay, loss or guaranteed bandwidth in Service Level Agreements (SLAs) can be provided using bandwidth guaranteed path routing schemes. This is especially motivated by the need of Internet Service Providers (ISPs) to provide on-line QoS requirements of rapidly expanding interactive and streaming services such as e-science, video-on-demand and IPTV.

Multi-Protocol Label Switching (MPLS) is one of the frameworks where the problem of on-line routing of bandwidth guaranteed paths is raised. MPLS allows explicit source routing [2] where part or all the intermediate nodes along a path are explicitly specified at the source node. This feature of MPLS can efficiently improve the network resource utilization and enhance the admissibility performance. In this study, we address the problem of on-line intra-domain bandwidth guaranteed path routing in MPLS networks.

An on-line routing algorithm should not need any a-priori knowledge about the traffic demand; it should adapt to the smooth changes in the traffic pattern and track the dynamics of the network. The on-line routing algorithm should be efficient, stable and scalable. In order to be scalable, the algorithm needs to be decentralized and impose low overhead to the network. This is one of the main drawbacks of many of the previously proposed routing techniques; many of these algorithms are either centralized or state dependent. In state dependent routing, the state information is flooded through the network and imposes flooding overhead. In this study, our focus is on employing decentralized routing and learning the routing policy using the locally observed events.

The use of decentralized learning in network routing raises a number of crucial issues including stability and asymptotic behavior. For most of the decentralized routing schemes, there is no theoretical approach to predict the asymptotic behavior of these algorithms and discrete event simulation is the only evaluation method that can be used. Another important issue with decentralized routing is the performance degradation due to the selfish behavior of end-to-end users. In decentralized routing, each user unilaterally tries to optimize its own performance regardless of its system-wide impact. In network design and analysis, it is of interest to investigate the performance gap between user optimal and centralized system optimal solutions in realistic network scenarios.

In this paper, we devise learning based, decentralized, event dependent routing techniques for source routing of bandwidth guaranteed paths in MPLS networks. We consider the case where there is no a-priori information about the traffic demand, but the topology of the network is known. The devised routing schemes use only the acceptance/rejection feedbacks to their bandwidth request attempts to update their routing policy. The performance of the algorithms are evaluated in terms of the network admissibility, that is the probability that a bandwidth request is admitted over the network. We analyze the asymptotic behavior of the proposed routing schemes and investigate their efficiency compared to centralized optimal solutions in some practical scenarios.

1.1 Related Work and Contributions

A variety of centralized and decentralized algorithms have been reported for routing in MPLS networks. Some of these algorithms assume all the point-to-point traffic demands are known. While this is a valid assumption in network design and analysis, in on-line routing this information is not always available. Many other algorithms such as [1, 3–10] are State Dependent Routing (SDR) methods. These algorithms update their routing policy based on the current state of the network. In these algorithms, state information as measured by link utilization, residual bandwidth or measured arrival rates is flooded through the network either periodically or on demand. The flooding overhead raises stability and scalability issues. Moreover, some of these algorithms such as [1] are computationally expensive and can process only one request at a time.

Event Dependent Routing (EDR) methods on the other hand, are decentralized routing schemes that update their routing policy based on the events that they observe. EDR schemes use distributed decision making and select a route for the current bandwidth request as a function of the outcome of their previous attempts. The routing algorithms presented in [11, 12] are examples of EDR schemes proposed for MPLS routing. Case studies reported in [11] for AT&T's network, indicate that network performance, as measured by end-to-end delay and loss probability, is comparable for EDR and SDR, while EDR is more scalable than SDR due to the state information flooding of SDR.

There are four major contributions of this paper. The first contribution is the introduction of reinforcement learning based, decentralized, event dependent routing techniques for routing in MPLS networks. The reinforcement learning framework [13] considers an agent with a given set of actions that interacts with a dynamic unknown environment and attempts to learn an optimal, or at least reasonable, action selection policy via a sequence of trials. We consider the reinforcement learning in the weak sense where the objective is not to solve an optimization problem; rather the goal is to perform reasonably over the sequence of trials. This is reminiscent of event dependent routing framework where the path chosen for routing the current bandwidth request is selected using the outcome of previous attempts and the outcomes depend on the network state that is assumed to be unknown. The devised learning based routing schemes have low computational complexity and track the dynamics of the network.

Our second contribution is the investigation of the asymptotic behavior of the proposed routing algorithms under different stationary/non-stationary assumptions for the network. We consider the user equilibrium where there is no incentive for any single user to unilaterally change its routing policy [14] as the natural result of routing rationally and prove the asymptotic convergence of one of the devised routing schemes to the user equilibrium. This predictable behavior is important in network design and analysis and is one of the strengths of our devised routing techniques in comparison with alternate EDR schemes such as the algorithm considered in [11]. For the EDR scheme of [11] and many other EDR and SDR schemes, there is no theoretical approach to predict the performance of the algorithm and discrete event simulation is the only method that can be used.

The third contribution of this paper addresses the investigation of the gap between the performance of the devised decentralized routing schemes and that of the system optimal solution. We provide nonlinear programming formulations of the user equilibrium and the system optimum problems of the proposed routing procedure and present a distributed recursion method for computing these solutions. This numerical method is then used to examine the quality of the user optimal solutions as opposed to system optimal routing policies in realistic network environments.

System optimal policies can lead to unfair resource allocation among the users where users with the same objective do not equally share the resources. The axioms of bargaining and the Nash bargaining solution from game theory provide the mathematical framework to address the optimality and fairness issues. In the packet switching context, these concepts have been applied to the network flow control problem in [15] and [16]. The notion of Nash bargaining is used in [17] for inter-domain traffic engineering and in [18] for bandwidth management in multimedia applications. In this study, we apply these concepts to intra-domain routing in MPLS networks and propose the Nash bargaining solution as an alternate Pareto optimal, fair, centralized benchmark routing policy. We extend the proposed distributed recursion method to compute the Nash bargaining solution and compare the performance of the user and system optimal routing policies with this fair solution.

Our fourth contribution is an on-line routing architecture that obtains the performance of the system optimal and the Nash bargaining solutions in a partially-decentralized manner. This routing mechanism has practical application in scenarios where the network capacities and traffic load are not well matched, due to abnormal traffic conditions or network failures and decentralized routing schemes can be inefficient. Centralized routing schemes are not applicable for on-line routing in these scenarios due to their computational load and scalability issues. We provide a technique for on-line implementation of this architecture.

1.2 Structure of the Paper

This paper is organized as follows. The proposed routing algorithms are described in Section 2. The asymptotic behaviors of these algorithms are studied in Section 3. The nonlinear programming formulations of the user equilibrium, system optimum and Nash bargaining problems and the numerical methods for computing their solutions are presented in Section 4. Section 5 gives an on-line routing architecture that converges to the system optimal and the Nash bargaining solutions in a partially-decentralized manner. The simulation results are discussed in Section 6. Summary and conclusions are given in Section 7.

2 Reinforcement Learning-based Load Shared Sequential Routing

We consider the Load Shared Sequential Routing (LSSR) [19] algorithm for routing the bandwidth guaranteed paths and propose the application of reinforcement learning schemes to update the load sharing factors on-line. We call these algorithms Reinforcement Learning-based Load Shared Sequential Routing (RL-based LSSR) schemes. In the following, we first describe the structure and the functionality of the LSSR algorithm. We then review the learning schemes used for updating the load sharing factors and discuss the properties that made them suitable candidates for our application. This is an extension to the results presented in [20].

2.1 Algorithm Description

Consider a general network topology with a set of alternate paths between each origin-destination pair (o, d). The set of alternate paths between each (o, d) are the K-shortest paths between these two nodes. The maximum number of considered alternate paths depends on the topology of the network and the maximum number of paths that are tried sequentially in routing the bandwidth requests as will be explained later. For each class 's' traffic between each (o, d), a set of Route-Trees (RTs) are built each with one or more alternate paths. The number and the order of the alternate paths varies from one RT to the other. The LSSR scheme assumes that for every (o, d, s), there is a load sharing vector $\underline{\alpha}^{o,d,s}$ such that:

$$\sum_{\ell=1}^{|K_{o,d,s}|} \alpha_\ell^{o,d,s} = 1, \quad \text{and} \qquad \alpha_\ell^{o,d,s} \geq 0, \tag{1}$$

where $\ell = 1, 2, \ldots, |K_{o,d,s}|$; $K_{o,d,s}$ is the set of RTs for class 's' traffic from 'o' to 'd' and $\alpha_\ell^{o,d,s}$ is the load sharing factor of the ℓ^{th} RT for (o, d, s). A pictorial representation of the LSSR model is provided in Fig. 1.

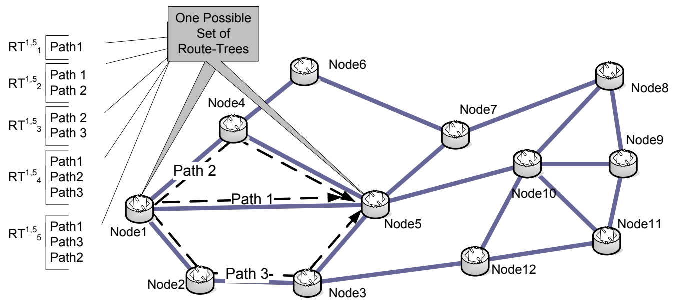

Fig. 1. LSSR model with 3-shortest paths and 15 possible RTs.

Upon arrival of a new bandwidth request for (o, d, s), one of the RTs is selected at random with $\alpha_\ell^{o,d,s}$ being the probability of selecting the ℓ^{th} RT and the alternate paths of the selected RT are tried sequentially. If there is not enough bandwidth available on at least one link of a path, a notification message is sent to the origin node and the origin node forwards the request to the next alternate path. In the MPLS context, the notification messages can be sent by constraint-based routing using the label distribution protocol [21] or the traffic engineering extensions of the resource reservation protocol [22]. This process is repeated until the requested bandwidth is allocated on one of the alternate paths of the selected RT or all the alternate paths of that RT have been tried unsuccessfully. If all the paths have been tried unsuccessfully, the bandwidth request is lost and is rejected from the network.

One of issues with alternate path routing is the bi-stable behavior that may occur as a result of routing the connections over non-shortest paths [23, 24]. We employ the trunk reservation mechanism [23] to avoid this bi-stable behavior. When the link load approaches its capacity, the trunk reservation mechanism gives the priority to those connections that are routed over the shortest paths by reserving the capacity that is above the trunk reservation threshold for these connections. The thresholds are determined using Krupp's square root rule [24].

The bandwidth constraint model used for allocating bandwidth to different traffic classes is the fully isolated Maximum Allocation Model (MAM) [25]. MAM gives an upper bound on the maximum bandwidth that can be reserved for each traffic class on each link. In fully isolated MAM, the sum of bandwidth constraints of different classes is less than or equal the link capacity. Using this bandwidth constraint model in LSSR, one can assume that the routing modules associated with different traffic classes of one (o, d) work in parallel and are isolated from each other.

In the on-line load share update procedure that we propose, there is a learning module associated with each (o, d, s) that uses the $\{0, 1\}$ rejection/acceptance feedback from the network and a learning technique to update the load sharing factors. At each bandwidth request arrival, only the rejection/acceptance feedback for the selected route-tree is revealed to the learning module and the learning module uses only this information to update its load sharing vector.

Learning automata and multi-armed bandit are two classes of reinforcement learning algorithms that naturally fit learning through trials and errors with partial information [13]. These frameworks consider an agent that interacts with a dynamic unknown environment and attempts to learn an optimal, or at least reasonable, action selection policy via a sequence of trials. At each trial n, the agent selects one of the possible actions $a(n) = a_i \in \underline{A}$. The agent then receives a reward $x(n) \in \underline{X}$ which is a measure of the desirability of the selected action from the environment. The agent uses this feedback signal to update its policy. This is called learning with partial information (in contrast with the full information case where at each trial the information about the rewards associated with all the actions is revealed to the agent).

We have used Linear Reward-Inaction (LRI) and Linear Reward-ϵPenalty (LRϵP) updating schemes from learning automata [26] and the EXP3.P algorithm from multi-armed bandit framework [27] for updating the load sharing factors. These algorithms have appropriate asymptotic behaviors as will be explained later that make them suitable candidates for updating the routing policy in the LSSR model. However, one should note that these are not the only possible candidates. The update procedure of LRI, LRϵP and EXP3.P schemes are reviewed in the following subsections.

Learning Automata Learning automata is one of the earliest frameworks of learning via trial and error [26], [28]. Formally, a learning automaton is described as a quadruple $\{\underline{A}, \underline{P}, \underline{X}, T\}$ in which $\underline{\mathrm{A}}$ is the set of actions with $\mathrm{K} = |A|$, $\underline{\mathrm{P}}$ is the probability distribution over the set of actions, $\underline{X}$ is the set of possible responses from the environment and T is the learning scheme:

$$\underline{P}(n+1) = T(\underline{P}(n), a(n), x(n)). \tag{2}$$

Both the LRI and the LRϵP schemes assume that $\underline{X} = \{0, 1\}$, with $x(n) = 1$ if the selected action is rewarded and $x(n) = 0$ otherwise. The learning schemes of these algorithms taken from [26] is given in Fig. 2.

In the LRI scheme, the parameter β, as presented in Fig. 2, is equal to zero and the probabilities are updated only when the selected action is rewarded. The LRI method is known to be ϵ-optimal if the reward stream is stationary [26]. That is, if the reward process (for each action) is stationary and if one of the actions, a_k, has the highest expected reward, $\forall \quad 0 < \epsilon < 1$, by choosing the parameter γ arbitrarily small, $Pr\{\lim_{n\to\infty} p_k(n) = 1\} > 1 - \epsilon$.

The LRϵP learning rule is derived from the above formula by choosing $\gamma \gg \beta$. The LRϵP rule is sub-optimal in comparison with the LRI algorithm in stationary environments. However, it avoids being locked in a state and is a better choice for non-stationary environments. For the LRϵP algorithm, a large value of γ will result in a large variance in the steady state distribution.

```
Procedure: The LRεP algorithm
Parameters: $\gamma \in (0,1)$ and $\beta \ll \gamma$.
1. Initialize $p_1(1), \ldots, p_K(1)$ at random at an interior point of the distribution
   set.
2. At each trial $n = 1, \ldots$:
   (a) Choose $a(n) = a_i$ randomly according to the distribution
       $p_1(n), \ldots, p_K(n)$.
   (b) Receive reward $x_i(n) \in \{0,1\}$.
   (c) For $j = 1, \ldots, K$:
       if $x_i(n) = 1$,
          $p_j(n+1) = \begin{cases} (1-\gamma)\, p_j(n) & \forall j \neq i \\ p_j(n) + \gamma\,(1 - p_j(n)) & j = i; \end{cases}$
       if $x_i(n) = 0$,
          $p_j(n+1) = \begin{cases} \frac{\beta}{K-1} + (1-\beta)\, p_j(n) & \forall j \neq i \\ (1-\beta)\, p_j(n) & j = i. \end{cases}$
```

Fig. 2. The LRεP algorithm.

The Multi-Armed Bandit Problem The multi-armed bandit literature also considers the problem of learning the action-selection policy through a sequence of trials and errors [29]. The variant of the multi-armed bandit problem we consider is the so-called adversarial multi-armed bandit problem where the reward process is non-stochastic and is assumed to be generated by Nature or even by an adversary.

Exponential weighting-type algorithms are one class of the learning algorithms proposed to solve the multi-armed bandit problem [30]. In these algorithms, there is a weight associated with each action that is an exponential function of the cumulative reward of that action and the probability of selecting an action is a function of its weight. We use the variant of the exponential weighting methods, called EXP3.P, proposed to solve the problem when the reward generation can be adversary [27]. The pseudo code of the algorithm taken from [27] for a sequence of N trials is presented in Fig. 3.

In this algorithm, the parameters κ and γ control the amount of exploration done by the algorithm initially (κ) and persistently (γ) over the sequence of N trials.

Procedure: The EXP3.P algorithm
Parameters: $\kappa > 0$ and $\gamma \in (0,1]$.

1. Initialization: For $i = 1, \ldots, K$ $\quad w_i(1) = \exp\left(\frac{\kappa\gamma}{3}\sqrt{\frac{N}{K}}\right)$.
2. At each stage $n = 1, \ldots, N$:
   (a) For $i = 1, \ldots, K$ set $\quad p_i(n) = (1-\gamma)\frac{w_i(n)}{\sum_{j=1}^{K} w_j(n)} + \frac{\gamma}{K}$.
   (b) Choose $a(n) = a_i$ randomly according to the distribution $p_1(n), \ldots, p_K(n)$.
   (c) Receive reward $x_i(n) \in [0,1]$.
   (d) For $j = 1, \ldots, K$ set:

$$\hat{x}_j(n) = \begin{cases} \frac{x_j(n)}{p_j(n)} & j = i \\ 0 & \text{otherwise,} \end{cases}$$

$$w_j(n+1) = w_j(n) \times \exp\left(\frac{\gamma}{3K}(\hat{x}_j(n) + \frac{\kappa}{p_j(n)\sqrt{KN}})\right).$$

Fig. 3. The EXP3.P algorithm.

The EXP3.P algorithm belongs to the family of the so-called "no-regret" algorithms. Here, regret is defined as the difference between the cumulative reward obtained from running the algorithm and that of the best over the hindsight action. In no-regret algorithms, the average regret converges to zero as the number of trials goes to infinity; see [29] and references therein. For the EXP3.P algorithm, by selecting the parameters appropriately, the no-regret property is held even in the worst case non-stationary environments that the reward of an action may depend on the past selected actions [27]. In the context of LSSR, this means that for each (o, d, s), the EXP3.P-based LSSR guarantees an upper bound on the difference between the number of admitted bandwidth requests of that (o, d, s) over a sequence of N trials when running the algorithm and the number of admitted requests if the best, over that period, RT was chosen constantly. This upper bound is held no matter what the routing policies of other pairs are.

The application of no-regret algorithms in network routing has been previously studied in the context of on-line shortest path routing; see, e.g., [31–33]. These studies assume a given weighted directed graph in which the edge weights can arbitrarily change over time. The weight of an edge represents the cost defined as delay or loss of that edge. No-regret algorithms give a routing policy with asymptotic average cumulative cost not much larger than that of the minimum weighted (shortest) path over the hindsight.

In comparison among three proposed routing schemes, the LRI-based LSSR is a more suitable choice when the network is (approximately) stationary. When this is not the case, the LRϵP-based LSSR and the EXP3.P-based LSSR algorithms are more suitable candidates. In the following subsections, the asymptotic behaviors of these algorithms are studied. These results are important in the network design and analysis as they predict the asymptotic behavior of the proposed event dependent routing schemes.

3 Asymptotic Behavior

The asymptotic behaviors of the proposed RL-based LSSR algorithms are investigated with two underlying assumptions: when the state of the network and consequently the route-tree admissibilities are (approximately) stationary and when the stationarity assumption is not applicable.

For the LRI-based LSSR, under stationary assumption, the user equilibrium is considered as the natural result of decentralized selfish routing. The concept of user equilibrium, also known as Wardrop equilibrium, is commonly used in non-cooperative strategic games [14]. Consider a non-cooperative game with a set of users $u \in \underline{U}$, each with a set of actions $\underline{A}(u)$. Let us assume that there is a utility function R_i^u associated with each $a_i \in \underline{A}(u), u \in \underline{U}$ and that each user rationally tries to maximize its own utility. In this setup, at the user equilibrium, there is no incentive in terms of utility increase for an individual user to unilaterally change its strategy and the user equilibrium can be viewed as the equilibrium solution of the game where all the users are selfishly trying to maximize their own utility.

In the LSSR model, each (o, d, s) is considered as a single user; the sets of actions are the sets of RTs and the utility associated with each RT is the admissibility of that RT. In this context, load sharing vectors $\underline{\alpha}^*$ are at user equilibrium if and only if, for each (o, d, s), there is no incentive in terms of increasing the admissibility, for changing the load sharing vector to $\widetilde{\underline{\alpha}}^{o,d,s} \neq \underline{\alpha}^{*o,d,s}$. In other words, in the LSSR model, at the user equilibrium, the admissibility of the used RTs ($\alpha_k^{*o,d,s} > 0$) are equal and these values are greater than or equal that of the unused RTs ($\alpha_k^{*o,d,s} = 0$). For the LRI-based LSSR under stationary assumption, we prove that the algorithm asymptotically converges to a user equilibrium:

Theorem 1. *For every network employing the LRI-based LSSR, there exists a $0 < \gamma_0 < 1$ such that by choosing the learning parameter of the LRI algorithm $\gamma < \gamma_0$, the routing policy $\underline{\alpha}(.)$ converges asymptotically to a user equilibrium ($\underline{\alpha}^*$) for every feasible interior initial set of load sharing factors $\underline{\alpha}(1)$:*

$$\exists 0 < \gamma_0 < 1 \quad \text{s.t.} \quad \forall \underline{\alpha}(1), \quad \text{if} \quad \gamma < \gamma_0, \quad \lim_{n\to\infty} \underline{\alpha}(n) = \underline{\alpha}^*.$$

Proof: In the following, we first present the proof of asymptotic convergence of the LRI-based LSSR to the user equilibrium by showing that its trajectory can be written as a set of Ordinary Differential Equations (ODEs). We then show that the equilibrium solutions of these ODEs are asymptotically stable.

3.1 Asymptotic Convergence Proof

The asymptotic convergence of the LRI-based LSSR to the user equilibrium is proved using the results of [34] and [35]. Consider a network where all the users ($u \in \underline{U}$), employ the LRI-based LSSR scheme for routing the bandwidth requests over the set of actions $\underline{A} = \times_{u \in U} \underline{A}(u)$. Let $\underline{\theta}_k$ be the set of probability vectors of all the users over the time interval between the arrival of k^{th} and $(k+1)^{th}$ bandwidth requests to the network. Note that the k^{th} and the $(k+1)^{th}$ bandwidth requests do not necessarily belong to the same user. Let the learning parameters of all the users be equal and set to γ. This system can be expressed as a stochastic approximation of the form:

$$\underline{\theta}^{\gamma}_{k+1} = \underline{\theta}^{\gamma}_{k} + \gamma Y^{\gamma}_{k}, \tag{3}$$

where Y^{γ}_{k} depends on the reward used for updating the $\underline{\theta}_k$ at the time of the k^{th} update. This system can also be represented as a finite state Markov Process (MP) $(\xi^{\gamma}_{k}, \theta^{\gamma}_{k})$ where ξ^{γ}_{k} is the state of the process at the time of k^{th} update. This process is stable for every stationary traffic distribution (for every $\underline{\theta}$), [23, 24], with the state transition probabilities of $Pr(\xi', \xi) = Pr(\xi_{k+1}(\underline{\theta}) = \xi' | \xi_k(\underline{\theta}) = \xi)$. Let $G(\xi, \underline{\theta})$ be defined as $G(\xi, \underline{\theta}) = \mathbb{E}\left(Y^{\gamma}_{k} | \xi^{\gamma}_{k} = \xi, \underline{\theta}^{\gamma}_{k} = \underline{\theta}\right)$ and $g(\underline{\theta})$ be defined as:

$$g(\underline{\theta}) = \lim_{m\to\infty} \frac{1}{m} \sum_{k=0}^{m-1} \mathbb{E}\left(G(\xi_k(\underline{\theta}), \underline{\theta})\right). \tag{4}$$

Now, consider the interpolation process of:

$$\underline{\pi}^{\gamma}(t) = \underline{\theta}^{\gamma}_{k} \quad \text{for} \quad t \in [k\gamma, (k+1)\gamma). \tag{5}$$

From the definitions of (3) and (4), for $t = k\gamma$, we have:

$$\left(\underline{\pi}^{\gamma}(t+\gamma) - \underline{\pi}^{\gamma}(t)\right)/\gamma = Y_k^{\gamma}, \tag{6}$$

and the conditional expected behavior of Y_k^{γ} with respect to the σ-algebra of the MP up to the k^{th} update, $\mathcal{F}_k^{\gamma}$, is written as:

$$\mathbb{E}\left(\frac{\underline{\pi}^{\gamma}(t+\gamma) - \underline{\pi}^{\gamma}(t)}{\gamma}|\mathcal{F}_k^{\gamma}\right) = G(\xi_k^{\gamma}, \underline{\pi}^{\gamma}(t)). \tag{7}$$

Using the results of [36], every subsequence of $\underline{\pi}^{\gamma}(.)$ as $\gamma \to 0$ has a weakly convergent subsequence; all weak limits are Lipschitz continuous a.s. and following [35], any such limit satisfies the ODEs of the form:

$$\frac{d\underline{\pi}(t)}{dt} = g[\underline{\pi}(t)]. \tag{8}$$

In a network employing the LRI-based LSSR, let $\lambda^{o,d,s}$ be the class 's' traffic from origin 'o' to destination 'd' and let $R_{\ell}^{o,d,s}(t)$ represent the admissibility of the class 's' traffic between (o,d) via the ℓ^{th} RT when $\underline{\alpha} = \underline{\pi}(t)$. From the above discussion and following the expected behavior of the LRI algorithm [37], the trajectory of each load sharing factor $\alpha_{\ell}^{o,d,s}$ satisfies an ODE of the form:

$$\frac{d\pi_{\ell}^{o,d,s}(t)}{dt} = \frac{\lambda^{o,d,s}}{\sum \lambda^{u,v,r}} \pi_{\ell}^{o,d,s}(t)\Big(R_{\ell}^{o,d,s}(t) - \sum_{r=1}^{|K_{o,d,s}|} R_r^{o,d,s}(t)\pi_r^{o,d,s}(t)\Big). \tag{9}$$

The r.h.s. of this equality is Lipschitz continuous and has a unique solution for every initial point. If it can be proved that the ODE has asymptotically stable points, then the limit points of $\lim_{t\to\infty} \underline{\pi}(t)$ converge to the set of stable points.

Note that the equilibrium solutions of (9) satisfy the necessary and sufficient conditions for being the user equilibrium. In other words, it is not possible for an initial condition to converge to an equilibrium point of (9) that is not a user equilibrium. Consequently, when the user equilibrium is unique, the LRI-based LSSR asymptotically converges to that unique user equilibrium solution for every initial starting point.

3.2 Asymptotic Stability Proof

The following proposition taken from [38] is used to obtain the sufficient conditions for the asymptotic stability of the dynamics of (9):

Proposition 1. *Suppose that there exists a continuously differentiable function* $V : \Theta \mapsto \mathbf{R}$ *such that for each* $\underline{\theta} \in \Theta$,

$$\frac{\partial V(\underline{\theta})}{\partial \theta(u, a_i)} = g_{u,a_i}(\underline{\theta}), \quad u \in \underline{U}, a_i \in \underline{A}(u), \tag{10}$$

where g_{u,a_i} *is defined as in (8). Then,* $\frac{d}{dt}V(\hat{\underline{\pi}}(t)) \geq 0$, *with the equality iff* $\hat{\underline{\pi}}(t)$ *is an equilibrium solution of (8). The isolated local maxima of* V *are asymptotically stable and* $\underline{\theta} \in \Theta$ *is a local maximum of* V *if for each* $u \in \underline{U}$, *it satisfies* $g_{u,a_i}(\underline{\theta}) = \max_{a_j \in \underline{A}(u)} g_{u,a_j}(\underline{\theta})$ *when* $\theta(u, a_i) > 0$.

In the LRI-based LSSR setup, let $B_{i,j,s}$ be the admissibility of class 's' traffic on the link (i, j) and $M_{i,j,s}$ be the class 's' traffic arrival rate on the link (i, j), where $B_{i,j,s} = B_{i,j,s}(\underline{M}_{i,j})$ and $M_{i,j,s} = M_{i,j,s}(\underline{B}, \underline{\alpha})$. Moreover, let the admissibility of each RT be a function of the admissibilities of the links along the paths of that RT, $R_k^{o,d,s} = R_k^{o,d,s}(\underline{B})$. We define the function Z^{ue} as:

$$Z^{ue} = \sum_{o,d,s,k} \Big(\int_0^{\alpha_k^{o,d,s}} \lambda^{o,d,s} R_k^{o,d,s}(\underline{B}) d\alpha + \int_0^{\alpha_k^{o,d,s}} \sum_{(i,j)\in(o,d,s,k)} \eta_{i,j,s}^{ue} \frac{\partial M_{i,j,s}}{\partial \alpha_k^{o,d,s}} d\alpha \Big), \tag{11}$$

with $\underline{\eta}^{ue}$ being $\underline{\eta}^{ue} = \left(\dot{\underline{M}}_b - \dot{\underline{B}}_m^{-1} \right)^{-1} \dot{\underline{Z}}_b^{ue}$ where $\dot{\underline{Z}}_b^{ue} = [\frac{\partial Z^{ue}}{\partial B_{i,j,s}}]$, $\dot{\underline{B}}_m = [\frac{\partial B_{i,j,s}}{\partial M_{i,j,r}}]$ and $\dot{\underline{M}}_b = [\frac{\partial M_{i,j,s}}{\partial B_{k,l,s}}]$. We then show that Z^{ue} satisfies the conditions of Proposition 1 and consequently is a Lyapunov function associated with ODEs of (9).

The partial derivative of Z^{ue} with respect to $\alpha_k^{o,d,s}$ is written as:

$$\frac{\partial Z^{ue}}{\partial \alpha_k^{o,d,s}} = \lambda^{o,d,s} R_k^{o,d,s} + \sum_{(i,j)\in(o,d,s,k)} \eta_{i,j,s}^{ue} \frac{\partial M_{i,j,s}}{\partial \alpha_k^{o,d,s}} + \sum_{p,q} \frac{\partial B_{p,q,s}}{\partial \alpha_k^{o,d,s}} \frac{\partial Z^{ue}}{\partial B_{p,q,s}}, \tag{12}$$

where for each link (p,q), $\frac{\partial B_{p,q,s}}{\partial \alpha_k^{o,d,s}}$ is written as:

$$\frac{\partial B_{p,q,s}}{\partial \alpha_k^{o,d,s}} = \frac{\partial B_{p,q,s}}{\partial M_{p,q,s}} \left(\frac{\partial M_{p,q,s}}{\partial \alpha_k^{o,d,s}} + \sum_{r,l} \frac{\partial M_{p,q,s}}{\partial B_{r,l,s}} \frac{\partial B_{r,l,s}}{\partial \alpha_k^{o,d,s}} \right). \tag{13}$$

By defining $\underline{\dot{B}}_\alpha$ and $\underline{\dot{M}}_\alpha$ as $\underline{\dot{B}}_\alpha = [\frac{\partial B_{i,j,s}}{\alpha_\ell^{u,v,r}}]$ and $\underline{\dot{M}}_\alpha = [\frac{\partial M_{i,j,s}}{\partial \alpha_\ell^{u,v,r}}]$, Equations (13) are written in compact form as:

$$\underline{\dot{B}}_\alpha = \left(\underline{\dot{B}}_m^{-1} - \underline{\dot{M}}_b \right)^{-1} \underline{\dot{M}}_\alpha. \tag{14}$$

Using the last set of equations, Equation (12) is equal to:

$$\begin{aligned} \frac{\partial Z^{ue}}{\partial \alpha_k^{o,d,s}} &= \lambda^{o,d,s} R_k^{o,d,s} + \sum_{(i,j)\in(o,d,s,k)} \eta_{i,j,s}^{ue} \frac{\partial M_{i,j,s}}{\partial \alpha_k^{o,d,s}} - \sum_{(i,j)\in(o,d,s,k)} \eta_{i,j,s}^{ue} \frac{\partial M_{i,j,s}}{\partial \alpha_k^{o,d,s}} \\ &= \lambda^{o,d,s} R_k^{o,d,s}, \end{aligned} \tag{15}$$

which satisfies the conditions of Proposition 1. This means that Z^{ue} is a Lyapunov function associated with ODEs of (9) and that for every initial point, the equilibrium solution of (9) is asymptotically stable. ■

For the EXP3.P-based LSSR, the no-regret property of the EXP3.P algorithm is used to investigate the asymptotic behavior of the algorithm. User equilibrium cannot be used as a benchmark for performance evaluation of no-regret algorithms as these algorithms do not necessarily converge to user equilibrium [29]. For these algorithms, the best over the sequence of trials, fixed action selection policy is used as the benchmark. No-regret algorithms give an upper bound on the difference between the average cumulative reward of the algorithm and that of this benchmark scenario. This upper bound goes to zero as the number of trials goes to infinity. Using this property, even when the stationary assumption for the network and consequently for the reward generation process of the learning algorithm does not hold, for each (o,d,s), the EXP3.P-based LSSR guarantees an upper bound on the difference between the number of admitted bandwidth requests over a sequence of N trials and the number of admitted requests if the best, over that period, RT was chosen constantly. The per round of this difference goes to zero as the number of trials goes to infinity.

4 Derivation of Load Sharing Factors at User Equilibrium, System Optimum and Nash Bargaining Solutions

A fundamental problem arising from fully decentralized routing is that these algorithms may lead to inefficient use of network resources. In the network design and analysis, the performance of decentralized routing schemes is compared with centralized optimal routing policies. The most commonly used centralized optimal policy, is the system optimal policy in which the sum of the utilities is maximized over the network. The Price Of Anarchy (POA) concept, as presented in [39], quantifies the degradation of the network performance due to the absence of a centralized routing policy. The POA is defined as the maximum ratio of the system optimal solution to the worst user equilibrium where the maximization is carried out over all the instances of the network topologies, traffic demands and utility functions. From a practical point of view, the POA is by definition pessimistic, as it compares the worst case performance of the potentially non unique user equilibrium, associated with decentralized control, with the system optimal solution. However, the practical scenarios are not necessarily the worst case scenarios and it is of interest to investigate the gap between the user equilibrium and the system optimal solutions in these cases.

The system optimal policy is the most efficient solution in terms of overall performance, but it can lead to unfair resource allocation among the users. In the system optimal solution, users with the same objective do not necessarily equally share the resources. The Nash bargaining solution is used as an alternate fair, Pareto optimal benchmark and the performance of the user equilibrium is also compared with this fair, efficient solution.

In the following subsections, we give the nonlinear programming formulations of the user equilibrium, system optimum and Nash bargaining problems for the LSSR model and present numerical methods for computing the solutions of these problems. These numerical methods are then used to evaluate the degradation of the performance due to decentralized routing as opposed to the centralized optimal solutions in some practical scenarios. The following subsections extend the results presented in [40].

4.1 User Equilibrium (UE)

In the context of the LSSR with $R_\ell^{o,d,s}$ being the admissibility of the ℓ^{th} RT of (o,d,s) when the load sharing vectors are equal to $\underline{\alpha}^*$, $\underline{\alpha}^*$ is a user equilibrium solution if and only if, for every (o,d,s), given the set of other load sharing vectors,

$$\sum_{\ell=1}^{|K_{o,d,s}|} \widetilde{\alpha}_\ell^{o,d,s} R_\ell^{o,d,s} \leq \sum_{\ell=1}^{|K_{o,d,s}|} \alpha_\ell^{*o,d,s} R_\ell^{o,d,s}. \tag{16}$$

In other words, at the user equilibrium, the admissibility of the used RTs, $(\alpha_k^{*o,d,s} > 0)$, are equal and these values are greater than or equal that of the unused RTs.

4.2 System Optimization (SYS)

In the system optimal solution, the load sharing factors are chosen such that the overall network utility is maximized. The system optimization problem has the following form:

$$\max_{\underline{\alpha}} Z^{sys}(\underline{\alpha}) = \sum_{o,d,s,k} \lambda^{o,d,s} \alpha_k^{o,d,s} R_k^{o,d,s} \tag{17}$$

subject to

$$\sum_{k=1}^{|K_{o,d,s}|} \alpha_k^{o,d,s} = 1$$

$$\alpha_k^{o,d,s} \geq 0.$$

Intuitively, the set of load sharing vectors $\underline{\alpha}^*$ is a local optimal solution of the system optimization problem if for any (o,d,s), the marginal gain of decreasing the load sharing factor of any RT is at most the marginal cost of increasing the load sharing factor of any other RT of that (o,d,s). In mathematical form, in the local optimal solution, for each (o,d,s), the partial derivatives of the objective function with respect to load sharing factors of the used RTs $(\alpha_k^{*o,d,s} > 0)$ are equal and these values are greater than or equal to the partial derivatives of the objective function with respect to load sharing factors of the unused RTs, $(\alpha_k^{*o,d,s} = 0)$.

4.3 Nash Bargaining (NB)

In the Nash bargaining problem, the objective of the users is to improve their performance through cooperation and to have a performance at least as good as a given disagreement outcome where there is no cooperation among the users. In the Nash bargaining solution, no single user can increase its utility without adversely affecting any other user and the users with the same objective equally share the network resources. The Nash bargaining solution is the only solution that satisfies the four axioms of bargaining: Pareto efficiency, symmetry, independence of affine transformation and independence of irrelevant alternatives. For more detailed information on the axioms of bargaining, refer to [41].

The Nash bargaining solution solves an optimization problem with the objective function being equal to the product of surplus with respect to disagreement outcome of the utility functions of all the users [41]. In the context of the LSSR, when the disagreement outcomes of all the users are set to zero, the Nash bargaining solution solves the following optimization problem:

$$\max_{\underline{\alpha}} Z^{nb}(\underline{\alpha}) = \prod_{o,d,s} \lambda^{o,d,s} \left(\sum_{k=1}^{|K_{o,d,s}|} \alpha_k^{o,d,s} R_k^{o,d,s} \right) \tag{18}$$

$$\text{subject to}$$

$$\sum_{k=1}^{|K_{o,d,s}|} \alpha_k^{o,d,s} = 1$$

$$\alpha_k^{o,d,s} \geq 0.$$

Because of the monotonicity of the log function, the solution of the above optimization problem also solves an alternate optimization problem where the objective function is replaced with:

$$\widetilde{Z}^{nb}(\underline{\alpha}) = \sum_{o,d,s} \log \Big(\lambda^{o,d,s} \sum_{k=1}^{|K_{o,d,s}|} \alpha_k^{o,d,s} R_k^{o,d,s} \Big). \tag{19}$$

As this alternate objective function simplifies the formulations, it will be used throughout the rest of the paper.

Here again, at the local optimal solution, for each (o, d, s), the partial derivative of $\widetilde{Z}^{nb}$ with respect to load sharing factors of the used RTs ($\alpha_k^{*o,d,s} > 0$)

are equal and these values are greater than or equal that of the unused RTs, $(\alpha_k^{*o,d,s} = 0)$.

4.4 Recursive Solution Algorithm

We use the ODEs associated with the expected behavior of the LRI-based LSSR algorithm to solve the three above-mentioned problems. In our proposed approach, these problems are solved by first initializing the load sharing factors with an interior point of the distribution set. The load sharing factors are then updated iteratively using the updating scheme of:

$$\alpha_k^{o,d,s}(n+1) = \alpha_k^{o,d,s}(n)\Big[1 + \overline{\gamma}\Big(s_k^{o,d,s}(n) - \sum_{r=1}^{|K_{o,d,s}|} s_r^{o,d,s}(n)\alpha_r^{o,d,s}(n)\Big)\Big], \quad (20)$$

with,

$$s_k^{o,d,s}(n) = \begin{cases} \frac{\lambda^{o,d,s}}{\sum \lambda^{u,v,r}} \left(c_1^{o,d,s} \frac{\partial Z(n)}{\partial \alpha_k^{o,d,s}} + c_2^{o,d,s} \right) & \text{SYS \& NB} \\ \frac{\lambda^{o,d,s}}{\sum \lambda^{u,v,r}} R_k^{o,d,s}(n) & \text{UE,} \end{cases}$$

where $0 < \overline{\gamma} < 1$ is the learning parameter and depending on the problem to be solved, $Z(n)$ is replaced by $Z^{sys}(n)$ or $\widetilde{Z}^{nb}(n)$. The constants $c_1^{o,d,s}$ and $c_2^{o,d,s}$ are chosen such that $0 \leq s_k^{o,d,s} \leq 1, \quad \forall k \in \{1, \ldots, |K_{o,d,s}|\}$. This iterative process is repeated until $|\alpha_k^{o,d,s}(n+1) - \alpha_k^{o,d,s}(n)|$ is sufficiently small.

For the system optimization and the Nash bargaining problems, in order to calculate the values of $s_k^{o,d,s}$ terms in each iteration, the partial derivatives of the objective function with respect to load sharing factors need to be calculated. Theses values can be calculated as follows:

System Optimization For the system optimization problem, at each iteration, the values of the partial derivatives of $Z^{sys}(n)$ with respect to load sharing factors, $\alpha_k^{o,d,s}$, are obtained by considering two sets of auxiliary variables of link admissibilities, $\underline{B} = [B_{i,j,s}]$ and link arrival rates, $\underline{M} = [M_{i,j,s}]$ with $B_{i,j,s} = B_{i,j,s}(\underline{M}_{i,j})$ and $M_{i,j,s} = M_{i,j,s}(\underline{B}, \underline{\alpha})$. Here again, we assume that the route-tree admissibilities are functions of link admissibilities, $R_k^{o,d,s} = R_k^{o,d,s}(\underline{B})$. We then have:

$$\frac{\partial Z^{sys}}{\partial \alpha_k^{o,d,s}} = \lambda^{o,d,s} R_k^{o,d,s} + \sum_{p,q} \frac{\partial B_{p,q,s}}{\partial \alpha_k^{o,d,s}} \frac{\partial Z^{sys}}{\partial B_{p,q,s}}, \tag{21}$$

where for each link (p,q), $\frac{\partial B_{p,q,s}}{\partial \alpha_k^{o,d,s}}$ is written as:

$$\frac{\partial B_{p,q,s}}{\partial \alpha_k^{o,d,s}} = \frac{\partial B_{p,q,s}}{\partial M_{p,q,s}} \left(\frac{\partial M_{p,q,s}}{\partial \alpha_k^{o,d,s}} + \sum_{k,l} \frac{\partial M_{p,q,s}}{\partial B_{k,l,s}} \frac{\partial B_{k,l,s}}{\partial \alpha_k^{o,d,s}} \right). \tag{22}$$

Using the same procedure as in the previous section, the set of Equations (22) can be written in a compact form as:

$$\dot{\underline{B}}_\alpha = \left(\dot{\underline{B}}_m^{-1} - \dot{\underline{M}}_b \right)^{-1} \dot{\underline{M}}_\alpha, \tag{23}$$

where $\dot{\underline{B}}_m = [\frac{\partial B_{i,j,s}}{\partial M_{i,j,r}}]$, $\dot{\underline{M}}_b = [\frac{\partial M_{i,j,s}}{\partial B_{k,l,s}}]$ and $\dot{\underline{M}}_\alpha = [\frac{\partial M_{i,j,s}}{\partial \alpha_\ell^{u,v,s}}]$. Let $\underline{\eta}^{sys}$ be defined as $\underline{\eta}^{sys} = \left(\dot{\underline{B}}_m^{-1} - \dot{\underline{M}}_b \right)^{-1} \dot{\underline{Z}}_b^{sys}$, where $\dot{\underline{Z}}_b^{sys} = [\frac{\partial Z^{sys}}{\partial B_{i,j,s}}]$. Equations (21) will be equivalent to:

$$\frac{\partial Z^{sys}}{\partial \alpha_k^{o,d,s}} = \lambda^{o,d,s} R_k^{o,d,s} + \sum_{(i,j)\in(o,d,s,k)} \eta_{i,j,s}^{sys} \frac{\partial M_{i,j,s}}{\partial \alpha_k^{o,d,s}}, \tag{24}$$

where, $\underline{\eta}^{sys} = \left(\dot{\underline{B}}_m^{-1} - \dot{\underline{M}}_b \right)^{-1} \dot{\underline{Z}}_b^{sys}$, $\dot{\underline{Z}}_b^{sys} = [\frac{\partial Z^{sys}}{\partial B_{i,j,s}}]$. In order to calculate the values of the partial derivatives of the objective functions with respect to each $\alpha_k^{o,d,s}$ from (24), the values of link arrival rates, link admissibilities, their partial derivatives and the route-tree admissibilities need to be calculated. We adopt the model reported in [42], which approximates the performance of alternate path routing in arbitrary network topologies, multiple traffic classes and class dependent access via trunk reservation to calculate the link arrival rates, their partial derivatives and the $R_k^{o,d,s}$ terms. In this method, given the link admissibilities, $\underline{B} = [B_{i,j,s}]$, the admissibility of each route-tree $R_k^{o,d,s}$ with $|K_k^{o,d,s}|$ alternate paths is written as the sum of $|K_k^{o,d,s}|$ independent event probabilities. The r^{th} probability is the probability that the r^{th} path is admissible while the first $r-1$ paths are not admissible. Each of these probabilities is calculated using the conditional probability of the first $r-1$ paths not being admissible given that the r^{th} path is admissible. These conditional probabilities are calculated from the binomial moments using the exclusion-inclusion principle.

Given the link arrival rates, the Bean-Gibbens-Zachary method [43] is used to derive the admissibility of each traffic class 's' on each link (i, j), $B_{i,j,s} = B_{i,j,s}(\mathbf{m_{i,j}})$. The fixed point equations of $\underline{B} \to \underline{M}$, $\underline{M} \to \underline{B}$ are solved in an iterative manner to obtain consistent link flows and admissibilities.

Nash Bargaining Using a similar approach, for the Nash bargaining problem, the partial derivative of $\widetilde{Z}^{nb}$ with respect to each $\alpha_k^{o,d,s}$ is written as:

$$\frac{\partial \widetilde{Z}^{nb}}{\partial \alpha_k^{o,d,s}} = \frac{R_k^{o,d,s}}{\sum_r \alpha_r^{o,d,s} R_r^{o,d,s}} + \sum_{p,q} \frac{\partial B_{p,q,s}}{\partial \alpha_k^{o,d,s}} \frac{\partial \widetilde{Z}^{nb}}{\partial B_{p,q,s}}, \tag{25}$$

and as a result, by defining $\underline{\eta}^{nb}$ as $\underline{\eta}^{nb} = \left(\underline{\dot{B}}_m^{-1} - \underline{\dot{M}}_b \right)^{-1} \dot{\underline{\widetilde{Z}}}_b^{nb}$, with $\dot{\underline{\widetilde{Z}}}_b^{nb} = [\frac{\partial \widetilde{Z}^{nb}}{\partial B_{i,j,s}}]$, Equation (25) is equivalent to:

$$\frac{\partial \widetilde{Z}^{nb}}{\partial \alpha_k^{o,d,s}} = \frac{R_k^{o,d,s}}{\sum_r \alpha_r^{o,d,s} R_r^{o,d,s}} + \sum_{(i,j)\in(o,d,s,k)} \eta_{i,j,s}^{nb} \frac{\partial M_{i,j,s}}{\partial \alpha_k^{o,d,s}}. \tag{26}$$

The same fixed point equations as in the system optimization problem can be employed to calculate the values of $\underline{\eta}^{nb}$ terms and the values of partial derivatives of (26).

User Equilibrium For the user equilibrium case, the expected reward of each RT used in the recursion formula of (20) is equal to the admissibility of that RT. Here again, the model of [42] as explained above can be used to calculate the route-tree admissibilities.

The proposed method for computing the user equilibrium, the system local optimal and the Nash bargaining solutions is summarized in Fig. 4.

In choosing the learning parameter $\overline{\gamma}$, there is trade off between the convergence speed of the algorithm and the amplitude of fluctuations around the equilibrium; the smaller the learning parameter, the slower the convergence speed and the smoother the convergence curve appears. By increasing the learning parameter, the algorithm converges more quickly to the equilibrium solution in the first iterations, but continues by fluctuating around the equilibrium solution with a larger amplitude. The learning parameter can also be time dependent, with its value being large at first to speed up the convergence and reduced near the equilibrium solution to avoid or reduce oscillations about the equilibrium value.

Procedure: Recursive solution approach

```
Initialize the load sharing factors $\underline{\alpha}(0)$ at an interior point of the distribution
set at random.
repeat
  a) Calculate the $s_i^{o,d,s}(n)$ terms using
     $$s_i^{o,d,s}(n) = \frac{\lambda^{o,d,s}}{\sum \lambda^{u,v,r}} \left( c_1^{o,d,s} w_i^{o,d,s} + c_2^{o,d,s} \right),$$
  where,
     $$w_k^{o,d,s} = \begin{cases} R_k^{o,d,s} + \sum_{(i,j) \in (o,d,s,k)} \eta_{i,j,s}^{sys} \frac{1}{\lambda^{o,d,s}} \frac{\partial M_{i,j,s}}{\partial \alpha_k^{o,d,s}} & \text{SYS} \\ \frac{R_k^{o,d,s}}{\sum_r \alpha_r^{o,d,s} R_r^{o,d,s}} + \sum_{(i,j) \in (o,d,s,k)} \eta_{i,j,s}^{nb} \frac{\partial M_{i,j,s}}{\partial \alpha_k^{o,d,s}} & \text{NB} \\ R_k^{o,d,s} \quad \text{UE} \end{cases}$$
  b) Update the load sharing factors using
     $$\alpha_i^{o,d,s}(n+1) = \alpha_i^{o,d,s}(n) \Big[ 1 + \overline{\gamma} \Big( s_i(n) - \sum_{r=1}^{|K_{o,d,s}|} s_r(n) \alpha_r^{o,d,s}(n) \Big) \Big].$$
until $|\alpha_i^{o,d,s}(n+1) - \alpha_i^{o,d,s}(n)| < \epsilon, \quad \forall (o,d,s,i).$
```

Fig. 4. Proposed recursive solution approach.

The selection of the optimal time dependent learning parameter is an optimal dynamic control problem outside the scope of our present research.

4.5 Asymptotic Convergence of the Recursive Solution Approach

The following theorem guarantees the asymptotic convergence of the distributed method of Fig. 4 to the locally optimal solution of the corresponding optimization problem.

Theorem 2. *For every initial point, the recursive algorithm presented in Fig. 4 converges asymptotically to the locally optimal solution of the related optimization problem.*

Proof: Using a similar argument to the proof of Theorem 1, by considering the interpolation of $\pi_k^{o,d,s}(t) = \alpha_k^{o,d,s}(n) \quad \text{for } t \in [n\overline{\gamma}, (n+1)\overline{\gamma})$, the recursive formula of (20) is written as:

$$\frac{d\pi_k^{o,d,s}(t)}{dt} = \pi_k^{o,d,s}(t)\Big(s_k^{o,d,s}(t) - \sum_{r=1}^{|K_{o,d,s}|} s_r^{o,d,s}(t)\pi_r^{o,d,s}(t)\Big). \tag{27}$$

The r.h.s. of these equations are Lipschitz continuous and have a unique solution for every starting point. The asymptotic stability of these solutions are proved following a similar approach as the proof of Theorem 1. We consider Z^{ue}, Z^{sys}, Z^{nb} as defined in Equations (11), (17) and (19) as potential Lyapunov functions and show that they satisfy the conditions of Proposition 1. From Proposition 1, we conclude that the solutions computed from the algorithm of Fig. 4 are asymptotically stable. ■

5 An On-line Approach to System Optimal and Nash Bargaining Solutions

In this section we propose an on-line partially-decentralized mechanism to achieve the system optimal and the Nash bargaining solution performances. This mechanism has practical importance in scenarios where there is a considerable gap between the performances of decentralized routing schemes and centralized optimal solutions. Decentralized routing schemes are not efficient in these scenarios and centralized numerical methods are not applicable to on-line routing due to their computational load. The architecture we propose is capable of generating and maintaining a performance comparable to the one expected by the centralized numerical method with a reduced on-line computational load.

The proposed mechanism employs the LRI-based LSSR algorithm. In a stationary environment, when decentralized routing modules use their local observations of rejection/acceptance of their bandwidth requests to update their load sharing factors, we proved that the LRI-based LSSR converges to the user equilibrium solution where the admissibilities of the used RTs are equal and these values are greater than or equal that of the unused RTs. On the other hand, following the results of the previous section, at the system optimal and the Nash bargaining solutions, for each (o, d, s), the $s_k^{o,d,s}$ terms, as defined in Fig. 4, of those RTs with load sharing factor greater than zero, are equal and have values greater than or equal those of the unused RTs. Combining the properties of the LRI-based LSSR and those of the system optimal and the Nash bargaining solutions, if the probability of observing a reward by a learning module in the LRI-based LSSR is changed from $R_k^{o,d,s}$ to $s_k^{o,d,s}$ of the system optimization or the Nash bargaining problem, the LRI-based LSSR will asymptotically converge to the related optimal solution. The probability of observing a reward at a learning module can be changed from $R_k^{o,d,s}$ to $s_k^{o,d,s}$, using the following formulation:

$$\begin{aligned} s_k^{o,d,s} &= Pr(\overline{x}=1) \\ &= Pr(\overline{x}=1|x=0) \times Pr(x=0) + Pr(\overline{x}=1|x=1) \times Pr(x=1), \end{aligned} \tag{28}$$

with,

$$Pr(\overline{x}=1|x=0) = \begin{cases} 0 & R_k^{o,d,s} \geq s_k^{o,d,s} \\ \frac{s_k^{o,d,s} - R_k^{o,d,s}}{1 - R_k^{o,d,s}} & R_k^{o,d,s} < s_k^{o,d,s}, \end{cases}$$

$$Pr(\overline{x}=1|x=1) = \begin{cases} \frac{s_k^{o,d,s}}{R_k^{o,d,s}} & R_k^{o,d,s} \geq s_k^{o,d,s} \\ 1 & R_k^{o,d,s} < s_k^{o,d,s}, \end{cases}$$

where x is the reward received from the network with $x = 1$ if the bandwidth request is accepted and $x = 0$ if it is rejected and $\overline{x}$ is the reward that is used for updating the load sharing factors.

This leads to a partially-decentralized routing architecture using the LRI-based LSSR model where one centralized processor is interconnected to every learning module. In this architecture, on a regular basis, the centralized processor updates its knowledge about the measured traffic rates between different nodes, recalculates the reward modification probabilities and sends them to the learning modules. The learning modules use the rejected/accepted signals received from the network, adjust the observed rewards using the reward modification probabilities and update their load sharing factors accordingly. The calculation of the reward modification probabilities is done in parallel to the operation of the decentralized learning modules. The interval between two updates of reward modification probabilities is chosen considering the trade off between reducing the computational load by increasing the interval and the accuracy of the obtained results. A pictorial representation of this architecture is given in Fig. 5.

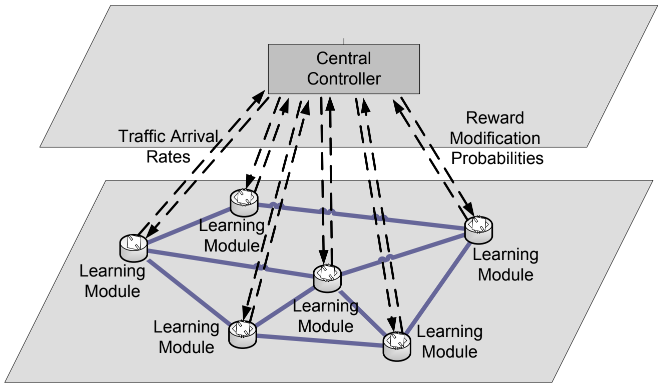

Fig. 5. An on-line partially-decentralized approach to the system local optimal and the Nash bargaining solutions using the LRI-based LSSR. Centralized methods of deriving the optimal routing policies are not applicable to on-line routing due to their computational load.

6 Simulation Results

The performance and the quality of the proposed routing algorithms have been investigated in three steps. In the first step, discrete event simulation has been employed to investigate the asymptotic and transient performances of the RL-based LSSR schemes. We show through discrete event simulation that the performances of the RL-based LSSR algorithms are comparable to other existing event dependent routing schemes in well engineered networks, i.e. networks where traffic flows and network capacities are "well matched". While for most of the event dependent and also state dependent routing algorithms, there is no theoretical performance prediction approach, the strength of the RL-based LSSR schemes is in their guaranteed asymptotic behavior as was explained in Section 3. For the performance comparison in transient phases, discrete event simulations have been carried out for scenarios where sudden changes such as link or node failure or abrupt changes in traffic pattern have happened in the network.

In the next step, the numerical algorithm of Fig. 4 has been used to investigate the performance gap between the user equilibrium and the system optimal and the Nash bargaining solutions in some practical scenarios. We study the efficiency of decentralized routing schemes in practical scenarios with well engineered networks; the worst case scenarios are not of interest in this section.

The last step illustrates the potential inefficiency in terms of resulting network admissibility of decentralized routing schemes when the network capacities and traffic demands are not well matched. These situations can happen in practice due to abnormal traffic demands or as a result of link or node failure. Simulation results of the partially-decentralized routing architecture show the merit in terms of improved admissibility performance of this architecture in these scenarios.

The following subsections summarize a description of three alternate path selection rules used for performance comparison and the three step investigation results.

6.1 Considered Path Selection Rules

Shortest Path First (SPF), Success-To-the-Top (STT) [11] and Repeated Load Sharing (RLS) [12] are the path selection rules used in this study for the performance comparison. SPF is not a QoS routing algorithm. It is chosen as a benchmark to verify how the RL-based LSSR algorithms can improve the QoS provisioning in comparison with the case where no QoS routing scheme is employed. STT is an event dependent routing algorithm successfully used in practice in AT&T networks [11] and RLS is an alternate recently proposed event dependent routing scheme [12]. One should note that the state dependent routing schemes that use the knowledge about the state of the network and/or the traffic demand such as the algorithms of [1, 3–9] are from a different setup and cannot be used as benchmark for the RL-based LSSR schemes. The path selection procedures of SPF, STT and RLS algorithms are reviewed in the following subsections.

Shortest Path First SPF is the off-line routing algorithm commonly used in IP networks. In SPF, the traffic of each (o, d, s) is routed along the shortest path between the two nodes. Different variants of the SPF algorithm use different distance functions such as the number of hops, the length of the links or the distance between two adjacent nodes being inversely proportional to the capacity of the link connecting these two nodes. In this study, we have used the SPF algorithm with the number of hops distance function.

Success-To-the-Top STT is a decentralized on-line routing method with a random updating scheme [11]. In this algorithm, the bandwidth request of an (o, d, s) is first sent through the primary path. When there is a direct link between two nodes, the primary path would be the direct path. If the request is blocked on this path, it is sent through the last successful secondary path. If the bandwidth request is blocked on both the primary and the last successful secondary, another alternate path is selected at random and the request is forwarded through this path. The algorithm allows a maximum of N crank backs. If the request is accepted on one of the alternate paths, that path is labeled as the last successful path to be used in routing the next bandwidth request of this (o, d, s).

To the best of our knowledge, there is no theoretical approach for evaluating the performance of STT.

Repeated Load Sharing In the routing algorithm proposed in [12], for each (o, d, s), there is a probability distribution associated with the set of alternate paths. Upon arrival of a new bandwidth request, one of the alternate paths is selected at random using this probability distribution and the bandwidth request is sent along this path. If there is not enough bandwidth on at least one of the links of the selected path, a notification message is sent back to the source node. The source node then excludes the paths already tried, renormalizes the probability of selecting remaining alternate paths such that they sum to '1' and selects another alternate path at random using this probability distribution. If the bandwidth request is accepted on one of the alternate paths, the probability of selecting this path is increased appropriately. In this study, we have used a modified version of this algorithm where the first path tried for routing the bandwidth request of each (o, d, s) is always the shortest path between the two nodes.

6.2 Discrete Event Simulation Results

Simulation studies have been carried out for two network topologies (14-node NSFnet [44] and 26-node AT&T [45]) and gravity model based traffic demands [46] where the traffic demand between each (o, d, s) is proportional to the product of two regions' population. These traffic matrices are presented in the Appendix. Forecasting errors are modeled with randomly generated variables with normal distribution and the standard deviation of 5% of the mean of the traffic demand. The traffic matrices used for admissibility performance comparison are the gravity model based traffic demands with a randomly generated forecasting error matrix. In order to have a fair comparison of different path selection procedures, the same set of alternate paths have been used for LSSR, STT and RLS with each RT in LSSR having four alternate paths and the maximum number of crank backs for STT and the maximum number of alternate paths tried in RLS being equal to four. For LSSR, only 6 RTs have been considered. As the fully isolated maximum allocation model (MAM) is used for bandwidth allocation to different classes, without loss of generality, only one class of traffic has been considered in the simulations and all bandwidth requests have been set to 1. In these simulations, for each (o, d, s), the parameters of the learning algorithms are tuned considering the number of RTs of that (o, d, s) and the RT admissibilities. For the LRI-based LSSR, the learning parameter is set to be $\gamma = .001$. For LRϵP-based LSSR, $\gamma = .001$ and $\beta = .0001$ and for the EXP3.P-based LSSR, the parameters are set at their optimal values that lead to optimal bound on the regret of the algorithm.

As a measure of network utility, an estimation of the total network admissibility is used. The admissibility is estimated using exponential smoothing. The following calculation is done recursively once rejected/accepted response to each bandwidth request is received:

$$R(n) = \rho R(n-1) + (1-\rho)x \ . \tag{29}$$

Here, $R(n)$ is the network admissibility after arrival of the n^{th} bandwidth request; x takes a value of '1' when the current bandwidth request is accepted and '0' otherwise. The parameter ρ is the smoothing parameter. The larger the ρ, the slower the R converges and the smoother the convergence curve appears. In the performed simulations, ρ has been set to 0.9999.

The mean and the standard deviation (STD) of the average admissibilities of SPF, STT, RLS and LSSR with different learning algorithms obtained from ten discrete event simulation runs each with a different randomly generated forecasting error matrix in the NSFnet and AT&T network topologies are presented in Table 1. The results of discrete event simulations are the average admissibilities after the transient phase. For each traffic matrix, the admissibility of the LSSR algorithm with load sharing factors at the user equilibrium computed from the numerical method of Section 4 is also listed in the table.

The results presented in Table 1 show that in all these cases, the event dependent routing algorithms have comparable admissibility performances and these performances are effectively better than the SPF algorithm. It is important to note that increasing the admissibility when employing the SPF algorithm translates to increasing the link capacities; SPF can achieve the admissibility performances of event dependent routing schemes only when the link capacities

Table 1. Admissibility performance comparison of different routing schemes in NSFnet and AT&T networks; 10 Runs, each with a different forecasting error matrix.

	NSFnet		AT&T	
	Mean	STD	Mean	STD
SPF	**.9537**	.0043	**.9712**	.0026
UE	.9888	.0020	.9830	.0027
LRI	.9891	.0024	.9832	.0025
LRϵP	.9890	.0020	.9830	.0027
EXP3.P	.9888	.0025	.9829	.0029
STT	.9906	.0025	.9835	.0025
RLS	.9912	.0022	.9840	.0025

are increased. Discrete event simulations with increased link capacities reveal that SPF achieves the admissibility performances of the event dependent routing schemes of Table 1 only when the link capacities are increased $12.78 \pm 256 \times 10^{-4}\%$ for NSFnet and $3.85 \pm 96 \times 10^{-4}\%$ for AT&T network topology. In practice, increasing the link capacities is a costly infrastructure modification operation. It is highly desirable to have routing algorithms that can improve the network admissibility without adding the infrastructure costs.

The next set of simulations illustrate the performance comparison of different routing algorithms in transient phase. For these simulations, the example 11-node network topology of Fig. 6 has been used. In this network, there are five alternate paths between $(1, 2)$ with the first path being the shortest path and four other paths having the same number of hops. Each route-tree of the RL-based LSSR has been built with three alternate paths and the maximum number of crank backs for STT and RLS has also been set to three. At time 300 sec, there has been an abrupt decrease on the admissibility of the link between $(1, 3)$ which could simulate abrupt increase on the traffic demand between $(1, 3)$ or a link or node failure along this path. In these simulation, for LSSR model, 36 RTs have been considered with the first 12 RTs having the shortest path as the first path and the next RTs having different combinations of other alternate paths. Considering the number of possible actions (number of RTs), the learning parameter in the LRI-based LSSR for these simulations is set to $\gamma = .0001$ and for LRϵP-based LSSR, $\gamma = .0005$ and $\beta = .00005$. For the EXP3.P-based LSSR, again, the γ and κ paramaters are set at their values that optimize the regret bound. Figure 7 is an example of simulation results of different algorithms.

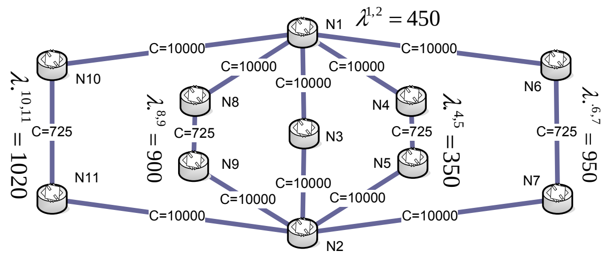

Fig. 6. Example 11-node network topology.

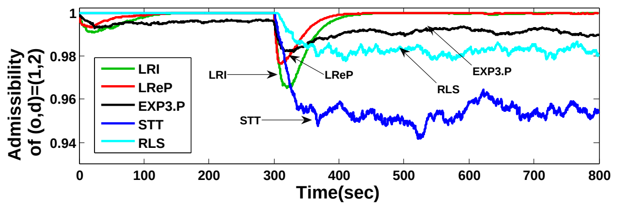

Fig. 7. Transient performance comparison in example 11-node network; at time 300 sec, the admissibility of the shortest path of $(1, 2)$ abruptly decreases.

After the startup, the longer transient time of the RL-based LSSR algorithms compared with that of STT and RLS is due to the large number of their route-trees. For the RL-based LSSR, the number of possible RTs when M alternate paths are used is $\mathcal{O}(M!)$ whereas for STT and RLS, at each trial, there are $\mathcal{O}(M)$ possible choices. The number of possible RTs can be an issue in employment of the RL-based LSSR schemes in mesh network topologies where all the alternate paths have the same number of hops and the number of alternate paths increases as the number of nodes increases. However, in general network topologies this issue is less important. This is due to the fact that in general network topologies, all the alternate paths do not have the same number of hops and the admissibility of the paths decreases as the number of hops along those paths increases. For that reason, in general network topologies, using only the first K-shortest paths can efficiently reduce the number of RTs and eliminate the number of RT complexity issue.

In the simulation results of Fig. 7, after the sudden change at time 300 sec on the admissibility of the shortest path of $(1, 2)$, this path cannot be used for routing the bandwidth requests of $(1, 2)$ anymore. STT has the lowest admissibility performance. This is due to the path selection rule of the STT algorithm where the shortest path is always tried first, then the last successful path is tried and if the bandwidth request is rejected on both these two paths, other alternate paths are tried randomly. Under normal traffic conditions, always choosing the shortest path first can improve admissibility performance as the traffic routed along the non-shortest paths uses more network resources. This is especially the case for fully connected networks where the traffic routed along a non-shortest path uses twice network resources than the traffic routed along the shortest path. However, when the shortest path is highly congested or when there is a failure along this path, as the maximum number of crank backs is fixed, always choosing the shortest path first can degrade the performance. In these simulations, RLS performs better than STT after the sudden change as a result the learning associated with its path selection policy. The RL-based LSSR schemes have the advantage over STT and RLS for the flexibility of learning the best sequence of alternate paths and adapting to the best sequence changes once they occur.

On the performance comparison of the LSSR with different learning schemes, the EXP3.P-based LSSR algorithm has a lower admissibility degradation peak and a shorter transient time when sudden changes happen in the network. However, after the transient phases, its average admissibility is smaller than the LRI-based LSSR and LRϵP-based LSSR. This observation can be explained considering the fact that the probability of selecting each RT is lower bounded in the EXP3.P algorithm. This lower bound helps in tracking the changes, while causing a smaller admissibility performance as a result of the lower bound on the amount of traffic sent to those RTs with high blocking rate. The shorter transient time of the LRϵP algorithm compared with the LRI scheme is due to the update procedure of this algorithm where the probability of the selected action is decreased if its associated reward is equal to zero.

6.3 Performance Investigation, Numerical Results

The comparison of the LSSR model with load sharing factors at user equilibrium, system optimal and Nash bargaining solutions are presented in this section. The performance is compared from two perspectives. First, the gap between the overall network admissibility obtained from decentralized routing schemes and that of the system optimal and the Nash bargaining solutions is investigated. Next, the fairness of the routing policies at the user equilibrium and system optimal solutions with respect to the Nash bargaining solution is examined. As the Nash bargaining solution maximizes the product of throughputs, the comparison of some function of product of throughputs can be used to investigate the fairness issue. In the following, the geometric mean of the admissibilities of all the users is employed to examine the fairness issue. The numerical results from the 14-node NSFnet and the 26-node AT&T network topologies with the same sets of RTs and the same ten different traffic matrices that were used in discrete event simulations are summarized in Table 2.

Table 2. Average network admissibility & geometric mean of (o, d, s) admissibilities for the LSSR model with load sharing factors at UE, SYS and NB solutions in the NSFnet and AT&T networks; average over 10 runs, each with a different forecasting error matrix

		Network Adm.		Adm. Geometric Mean	
		Mean	STD	Mean	STD
NSF	UE	.9888	.0020	.9898	.0022
	SYS	.9979	.0027	.9979	.0020
	NB	.9952	.0030	.9978	.0020
AT&T	UE	.9888	.0020	.9841	.0028
	SYS	.9892	.0022	.9855	.0028
	NB	.9868	.0021	.9870	.0029

The results of Table 2 show that there is a performance gap between the user equilibrium and system optimal solutions; however this gap is less than 1 $\pm 108 \times 10^{-4}\%$ and user equilibrium is relatively efficient for these well engineered networks. For these traffic matrices, the LSSR with load sharing factors at user equilibrium achieves the admissibility performance of the LSSR with system optimal load sharing factors when the link capacities are increased $8.28 \pm 801 \times 10^{-4}\%$ for NSFnet and $2.55 \pm 276 \times 10^{-4}\%$ for AT&T network topology. Moreover, these results reveal that the gap between the geometric mean of the admissibilities obtained from the system optimal and the Nash bargaining solutions are in the same range, suggesting that the system optimal solution is relatively fair in these cases. This in turn means that the proposed RL-based LSSR schemes provide decentralized, scalable and efficient approaches for routing the bandwidth guaranteed paths in well engineered networks.

The question of non-uniqueness of the local optimal solutions is investigated by solving the optimization problems with different randomly generated initial load sharing factors. In all the results obtained in this study, there is no numerically significant difference among the admissibilities obtained from different randomly generated starting points suggesting that the obtained results all belong to a flat admissibility region. Although in general there is no theoretical guarantee for the uniqueness of this region, in all the extensive numerical results we have performed, we have not observed any counter example.

6.4 Partially-Decentralized Architecture, Simulation Results

In this section we first give a simple example that illustrates the potential of performance degradation associated with decentralized routing when traffic matrix and link capacities are not well matched. We then demonstrate the merit of partially-decentralized routing architecture in terms of improved admissibility performance by converging to the system optimal solution for these scenarios.

Consider the small network of Fig. 8 with the link capacities and the traffic matrix as given in the figure. In this network, there are two alternate paths between nodes (1,2). Let us consider two RTs each with one alternate path for routing the traffic between (1,2). In the user equilibrium, all the traffic of (1,2) is routed over the first alternate path whereas in the system optimal solution, all the traffic of (1,2) is routed over the second path. A simple calculation using the ErlangB formula [47] shows that the system optimal solution improves the network admissibility by a factor of 1.03. In order to achieve the admissibility performance of the system optimal solution using the fully decentralized routing schemes, the link capacities need to be increased 5%. However, the partially decentralized routing architecture presented in Section 5 converges to this network admissibility performance without any need to increase the link capacities. The simulation results of fully decentralized LRI-based LSSR versus partially-decentralized routing architecture with system optimized reward modification probabilities are given in Fig. 9. In these simulations the learning parameter γ is set equal to .001.

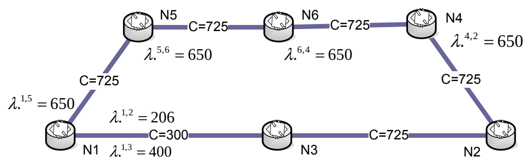

Fig. 8. Example 6-node network topology.

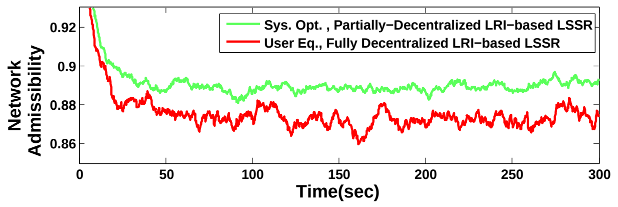

Fig. 9. Admissibility comparison of LSSR with user equilibrium and system optimal load sharing factors. System optimal performance is obtained from the partially-decentralized architecture.

This example is a network with abnormal link state/traffic demand simulating subnetwork abnormal conditions that can happen in practice due to link/node failures or abnormal traffic demands. In these scenarios, the proposed on-line partially-decentralized architecture converging to the system optimal solution can efficiently improve the network performance. Note that the on-line computational load overhead imposed to the decentralized modules in the proposed partially-decentralized routing architecture, compared with fully decentralized scheme, is related to the traffic measurements and the modification of the reward probabilities. The major computational load that is due to the update of the reward modification probabilities is performed in the central processor which is working in parallel with the decentralized modules. This reveals the applicability of the devised architecture for on-line routing of bandwidth guaranteed paths when the traffic demand and the network resources are not well matched and decentralized routing schemes are not efficient.

7 Summary and Conclusion

On-line routing of bandwidth guaranteed paths when there is no a-priori information about the traffic demand is a challenging problem with significant practical implications. In this paper, decentralized learning algorithms were proposed for bandwidth guaranteed path routing in MPLS networks. The proposed algorithms are rather simple and use only their locally observed information for updating their routing policy. The asymptotic behaviors of the proposed algorithms were investigated and the asymptotic convergence of one of these schemes to the user equilibrium under stationary assumption of the network state was proved. This predictable asymptotic behavior is especially important in the network design and analysis. The potential of using learning algorithms was demonstrated through discrete event simulation in 14-node NSFnet and 26-node AT&T network topologies. Simulation results suggest that the devised learning-based routing algorithms effectively outperform the SPF algorithm and have a comparable performance with alternate event dependent routing schemes, STT and RLS.

Our results suggest that the devised routing algorithms outperform the SPF algorithm in all the cases and are comparable with state-of-the-art event dependent routing schemes such as STT and RLS.

The performance of the proposed algorithms at the user equilibrium was compared with centralized solution benchmarks, the system optimal and the Nash bargaining solutions. Computationally efficient numerical methods for computing these solutions were provided. Numerical results for NSFnet and AT&T network topologies indicate that there is a performance gap between the user equilibrium and the centralized optimal solutions; however, this gap is typically quite small (less than $1 \pm 108 \times 10^{-4}\%$) for well engineered networks i.e. networks where traffic flows and network capacities are "well matched". These results indicate that the proposed decentralized learning techniques provide efficient, stable and scalable approaches for routing the bandwidth guaranteed paths in the well engineered networks.

As network capacities and traffic load may not always be well matched, due to abnormal traffic conditions or network failures, a partially-decentralized routing architecture was devised to obtain the performance of the system optimal and the Nash bargaining solutions. This mechanism has a performance comparable with the centralized routing methods and a reduced on-line computational load that makes it applicable to on-line routing. Example numerical results even for the well engineered NSFnet network topology, where the gap between the admissibility of the proposed routing schemes at the user equilibrium and the system optimal solution is $.9 \pm 108 \times 10^{-4}\%$, reveal that in order to achieve the admissibility performance of the system optimal solution using the fully decentralized routing schemes, the link capacities need to be increased $8.28 \pm 801 \times 10^{-4}\%$. For these scenarios, the partially-decentralized routing architecture achieves the system optimal performance without any need to increase the link capacities.

References

1. K. Kar, M. Kodialam, and T. V. Lakshman, "Minimum interference routing of bandwidth guaranteed tunnels with MPLS traffic engineering applications," *IEEE*

J. Sel. Areas Commun., vol. 18, no. 12, pp. 2566–2579, 2000.

2. D. O. Awduche, “MPLS and traffic engineering in IP networks,” *IEEE Commun. Mag.*, vol. 37, no. 12, pp. 42–47, 1999.
3. Z. Wang and J. Crowcroft, “Quality-of-Service routing for supporting multimedia applications,” *IEEE J. Sel. Areas Commun.*, vol. 14, no. 7, pp. 1228–1234, 1996.
4. R. Guerin, A. Orda, and D. Williams, “QoS routing mechanisms and OSPF extensions,” in *Proc. IEEE Conf. on Global Telecommunications*, Phoenix, AZ, USA, Nov. 1997.
5. W. Szeto, R. Boutaba, and Y. Iraqi, “Dynamic online routing algorithm for MPLS traffic engineering,” in *Proc. IFIP-TC6 Conf. on Networking*, Pisa, Italy, May 2002.
6. W. Bin, S. Xu, and C. L. P. Chen, “A new bandwidth guaranteed routing algorithm for MPLS traffic engineering,” in *Proc. IEEE Inter. Conference on Communications*, New York, NY, US, Apr./May 2002.
7. S. Suri, M. Waldvogel, D. Bauer, and P. R. Warkhede, “Profile-based routing and traffic engineering,” *Computer Communications*, vol. 26, no. 4, pp. 351–365, 2003.
8. F. Ricciato and U. Monaco, “Routing demands with time-varying bandwidth profiles on a MPLS network,” *Computer Networks*, vol. 47, no. 1, pp. 47–61, 2005.
9. N. Sengezer and E. Karasan, “An efficient virtual topology design and traffic engineering scheme for IP/WDM networks,” in *Proc. IFIP-TC6 Conf. on Optical Network Design and Modeling*, Athens, Greece, May 2007.
10. B. J. Oommen, S. Misra, and O. C. Granmo, “Routing bandwidth-guaranteed paths in MPLS traffic engineering: a multiple race track learning approach,” *IEEE Trans. Comput. (USA)*, vol. 56, no. 7, pp. 959–976, 2007.
11. G. R. Ash, *Traffic Engineering and QoS Optimization of Integrated Voice & Data Networks*. Morgan Kaufmann Publishers Inc., 2006.
12. N. Lam, Z. Dziong, and L. G. Mason, “Network capacity allocation in service overlay networks,” in *Proc. 20th International Teletraffic Congress*, Ottawa, Canada, June 2007.
13. R. S. Sutton and A. G. Barto, *Reinforcement Learning: an Introduction.* MIT Press, 1998.
14. G. Owen, *Game Theory.* Academic Press, Third Edition, 1995.
15. R. Mazumdar, L. G. Mason, and C. Douligeris, “Fairness in network optimal flow control: optimality of product forms,” *IEEE Trans. Commun.*, vol. 39, no. 5, pp. 775–782, 1991.
16. F. P. Kelly, A. K. Maulloo, and D. K. H. Tan, “Rate control for communication networks: shadow prices, proportional fairness and stability,” *Operational Research Society*, vol. 49, no. 3, pp. 237–252, 1998.
17. G. Shrimali, A. Akella, and A. Mutapcic, “Cooperative inter-domain traffic engineering using nash bargaining and decomposition,” in *Proc. IEEE Int. Conf. on Computer Communications*, Anchorage, AK, USA, May 2007.

18. H. Park and M. van der Schaar, "Bargaining strategies for networked multimedia resource management," *IEEE Trans. Signal Process.*, vol. 55, no. 7, Part 1, pp. 3496–3511, July 2007.
19. A. Girard and M. A. Bell, "Blocking evaluation for networks with residual capacity adaptive routing," *IEEE Trans. Commun.*, vol. 37, no. 12, pp. 1372–1380, 1989.
20. F. Heidari, S. Mannor, and L. G. Mason, "Reinforcement learning-based load shared sequential routing," in *Proc. IFIP-TC6 Conf. on Networking*, Atlanta, GA, USA, May 2007.
21. B. Jamoussi, L. Andersson, R. Callon, R. Dantu, L. Wu, P. Doolan, T. Worster, N. Feldman, A. Fredette, M. Girish, E. Gray, J. Heinanen, T. Kilty, and A. Malis, "Constraint-based LSP setup using LDP," *RFC 3212*, Jan. 2002.
22. D. Awduche, L. Berger, D. Gan, T. Li, V. Srinivasan, and G. Swallow, "RSVP-TE: Extension to RSVP for LSP tunnels," *RFC 3209*, Dec. 2001.
23. F. P. Kelly, "Loss networks," *Annals of Applied Probability*, vol. 1, pp. 319–378, 1991.
24. L. G. Mason, "On the stability of circuit-switched networks with non-hierarchical routing," in *Proc. IEEE Conf. on Decision and Control*, Athens, Greece, Dec. 1986.
25. F. L. Faucheur and W. Lai, "Maximum allocation bandwidth constraints model for Diff-Serv-aware MPLS traffic engineering," *IEFT Internet draft*, Mar. 2004.
26. K. S. Narendra and M. A. L. Thathachar, *Learning Automata: an Introduction.* Prentice Hall, 1989.
27. P. Auer, N. Cesa-Bianchi, Y. Freund, and R. E. Schapire, "The nonstochastic multi-armed bandit problem," *SIAM Journal on Computing*, vol. 32, no. 1, pp. 48–77, 2002.
28. K. Najim and A. Poznyak, *Learning Automata: Theory and Applications.* Springer, 1994.
29. N. Cesa-Bianchi and G. Lugosi, *Prediction, Learning, and Games.* Cambridge University Press, 2006.
30. N. Littlestone and M. K. Warmuth, "The weighted majority algorithm," *Information and Computation*, vol. 108, no. 2, pp. 212–261, 1994.
31. A. Kalai and S. Vempala, "Efficient algorithms for online decision problems," *J. of Computer and System Sciences*, vol. 71, no. 3, pp. 291–307, 2005.
32. A. György and G. Ottucsák, "Adaptive routing using expert advice," *Computer Journal*, vol. 49, no. 2, pp. 180–189, 2006.
33. A. Blum, E. Even-Dar, and K. Ligett, "Routing without regret, on convergence to Nash equilibria of regret-minimizing algorithms in routing games," in *Proc. ACM Symp. on Principles of Distributed Computing*, Denver, CO, USA, July 2006.
34. F. J. Vazquez-Abad and L. G. Mason, "Decentralized adaptive flow control of high-speed connectionless data networks," *Operations Research*, vol. 47, no. 6, pp. 928–942, 1999.

35. H. J. Kushner and F. J. Vazquez-Abad, “Stochastic approximation methods for systems over an infinite horizon,” *SIAM Journal on Control and Optimization*, vol. 34, no. 2, pp. 712–756, 1996.
36. F. J. Vazquez-Abad, C. G. Cassandras, and V. Julka, “Centralized and decentralized asynchronous optimization of stochastic discrete event systems,” *IEEE Trans. Autom. Control*, vol. 43, no. 5, pp. 631–655, 1998.
37. L. G. Mason, “An optimal learning algorithm employing cross-correlation,” in *Conference Record of Joint Automatic Control Conference*, Palo Alto, CA, US, Oct. 1972.
38. M. Alanyali, “Learning automata in games with memory with application to circuit-switched routing,” in *Proc. IEEE Conf. on Decision and Control*, Atlantis, Paradise Island, Bahamas, Dec. 2004.
39. C. Papadimitriou, “Algorithms, games, and the Internet,” in *Proc. ACM Symp. on Theory of Computing*, Heraklion, Crete, Greece, July 2001.
40. G. Brunet, F. Heidari, and L. G. Mason, “Load shared sequential routing in MPLS Networks: system and user optimal solutions,” in *Proc. Euro-NGI/FGI Conf. on Network Control and Optimization*, Avignon, France, June 2007.
41. M. J. Osborne and A. Rubinstein, *A Course in Game Theory.* MIT Press, 1994.
42. A. G. Greenberg and R. Srikant, “Computational techniques for accurate performance evaluation of multirate, multihop communication networks,” *IEEE/ACM Trans. Netw.*, vol. 5, no. 2, pp. 266–277, 1997.
43. N. G. Bean, R. J. Gibbens, and S. Zachary, “The performance of single resource loss systems in multiservice networks,” in *Proc. 14th International Teletraffic Congress*, Antibes Juan-les-Pins, France, June 1994.
44. “NSF diagrams.” [Online]. Available: http://www.merit.edu/networkresearch/projecthistory/nsfnet/nsfnet_maps.php
45. “An Atlas of cyberspaces: AT&T IP backbone network, 2Q2000.” [Online]. Available: http://www.cybergeography.org/atlas/more_isp_maps.html
46. J. P. Kowalski and B. Warfield, “Modeling traffic demand between nodes in a telecommunications network,” in *Proc. Australian Telecommunication Networks and Applications Conference*, Sydney, Australia, Dec. 1995.
47. J. R. Boucher, *Traffic System Design Handbook: Timesaving Telecommunication Traffic Tables and Programs.* Wiley, John & Sons, Incorporated, 1992.

Table 3. Capacity Matrix for NSFnet Network

0	78	130	90	0	0	0	0	0	0	0	0	0	0
76	0	146	0	0	0	0	172	0	0	0	0	0	0
129	149	0	0	0	216	0	0	0	0	0	0	0	0
93	0	0	0	99	0	0	0	117	0	0	0	0	0
0	0	0	102	0	126	70	0	0	0	0	0	0	0
0	0	218	0	119	0	0	0	0	0	0	0	107	98
0	0	0	0	70	0	0	70	0	0	0	0	0	0
0	167	0	0	0	0	73	0	0	0	0	209	0	0
0	0	0	116	0	0	0	0	0	253	109	0	0	0
0	0	0	0	0	0	0	0	253	0	0	247	113	0
0	0	0	0	0	0	0	0	109	0	0	74	70	0
0	0	0	0	0	0	0	218	0	242	72	0	0	112
0	0	0	0	0	103	0	0	0	119	70	0	0	0
0	0	0	0	0	97	0	0	0	0	0	112	0	0

Table 4. Traffic Matrix for NSFnet Network

0.000	16.090	72.643	1.574	7.658	22.593	1.778	14.147	9.283	9.055	3.581	7.112	1.562	3.975
16.090	0.000	11.785	0.927	4.254	10.960	1.066	8.293	5.816	5.613	2.006	4.432	0.990	1.981
72.643	11.785	0.000	4.470	14.732	48.425	3.022	27.177	15.709	15.836	7.327	12.776	2.895	6.883
1.574	0.927	4.470	0.000	2.679	4.559	0.317	3.390	2.019	1.803	1.181	1.498	0.330	0.711
7.658	4.254	14.732	2.679	0.000	18.237	2.451	15.595	7.645	6.261	2.578	5.892	1.244	2.692
22.593	10.960	48.425	4.559	18.237	0.000	8.801	69.113	34.290	30.340	14.858	27.393	6.604	21.869
1.790	1.066	3.022	0.317	2.451	8.801	0.000	18.288	4.572	2.971	1.282	3.340	0.647	1.676
14.160	8.293	27.177	3.390	15.595	69.113	18.288	0.000	56.108	32.384	14.401	42.671	6.858	22.021
9.283	5.816	15.709	2.019	7.645	34.290	4.572	56.108	0.000	36.436	16.611	64.008	9.258	12.280
9.055	5.613	15.836	1.803	6.261	30.340	2.984	32.384	36.436	0.000	68.008	43.954	12.954	10.401
3.581	2.006	7.327	1.181	2.578	14.858	1.282	14.401	16.611	68.008	0.000	22.402	2.540	4.991
7.112	4.432	12.776	1.498	5.892	27.393	3.340	42.671	64.008	43.967	22.402	0.000	15.367	10.769
1.562	0.990	2.895	0.330	1.244	6.604	0.647	11.226	9.258	12.954	2.540	15.367	0.000	3.022
3.975	1.981	6.883	0.711	2.692	21.869	1.676	22.021	12.115	10.401	4.991	10.769	3.022	0.000

Table 5. Traffic Matrix for AT&T Network, Columns[1-19]

0	40	47	0	0	0	45	0	0	0	0	0	0	0	0	0	0	0	0
40	0	47	0	0	0	50	0	0	0	0	0	0	0	0	0	0	0	0
47	47	0	198	0	0	1593	180	0	0	0	0	0	0	0	0	0	0	0
0	0	186	0	75	0	313	51	0	0	0	0	0	0	0	0	0	0	0
0	0	0	83	0	45	94	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	43	0	295	0	0	0	0	0	0	0	0	0	0	0	0
44	51	1609	292	101	317	0	0	991	0	0	0	0	0	0	0	1104	153	154
0	0	195	40	0	0	0	0	94	136	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	979	83	0	0	0	1027	0	0	0	0	132	0	0
0	0	0	0	0	0	0	140	0	0	38	67	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	40	0	66	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	1010	71	63	0	224	599	0	361	0	0	0
0	0	0	0	0	0	0	0	0	0	0	223	0	47	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	600	42	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	157	297	0	0
0	0	0	0	0	0	0	0	0	0	0	359	0	0	142	0	672	0	0
0	0	0	0	0	0	1114	0	116	0	0	0	0	0	308	665	0	72	0
0	0	0	0	0	0	163	0	0	0	0	0	0	0	0	0	57	0	0
0	0	0	0	0	0	155	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	139	0	0	0	0	0	0	0	0	0	70
0	0	0	0	0	0	103	52	60	0	0	46	0	0	0	87	0	0	0
0	0	0	0	0	0	0	0	42	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	126	0	0	0	0	175	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	167	0	0
0	0	0	0	0	0	0	0	0	0	0	396	0	0	61	0	0	0	0

Table 6. Traffic Matrix for AT&T Network, Columns[20-26]

0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	105	0	0	0	0	0
0	42	0	0	0	0	0
130	60	39	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	44	0	122	0	0	394
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	63
0	92	0	0	0	0	0
0	0	0	170	0	168	0
0	0	0	0	0	0	0
70	0	0	0	0	0	0
0	49	0	0	0	0	0
41	0	40	0	0	0	0
0	70	0	0	0	0	0
0	0	0	0	82	0	0
0	0	0	87	0	40	0
0	0	0	0	40	0	0
0	0	0	0	0	0	0

Table 7. Traffic Matrix for AT&T Network, Columns[1-13]

0.000	1.115	9.363	2.212	1.066	1.115	3.333	1.103	1.148	1.112	1.138	2.261	1.083
1.107	0.000	9.925	2.500	1.137	2.119	4.529	1.125	1.115	1.044	1.058	2.069	1.012
9.504	10.134	0.000	136.958	29.729	81.424	251.831	51.103	30.575	36.552	16.688	101.505	61.489
2.173	2.421	134.162	0.000	6.951	16.625	54.438	10.027	6.570	8.238	3.263	21.183	11.728
1.095	1.096	27.140	6.726	0.000	4.325	11.470	2.428	2.231	2.149	1.053	4.737	3.279
1.043	2.489	90.049	16.912	4.505	0.000	29.489	5.848	4.656	4.407	2.238	12.505	7.416
3.454	4.346	247.538	47.163	10.101	30.634	0.000	19.016	12.587	14.871	6.636	39.704	22.235
1.037	1.109	45.378	10.016	2.041	6.934	19.354	0.000	2.050	3.447	2.075	7.392	4.311
1.060	1.150	30.970	6.402	2.232	4.295	12.559	2.272	0.000	2.215	1.062	5.413	3.310
1.035	1.090	40.181	7.707	2.184	4.415	13.075	3.204	2.175	0.000	1.084	5.334	3.009
1.158	1.071	16.543	3.369	1.037	2.243	7.107	2.297	1.131	1.009	0.000	3.507	2.385
2.125	2.357	106.455	22.144	4.335	13.936	35.401	7.766	5.417	5.617	3.094	0.000	8.944
1.005	1.052	58.221	11.424	3.203	7.938	23.220	4.586	3.487	3.178	2.134	7.873	0.000
2.289	3.433	174.198	28.637	7.354	20.599	59.221	13.836	7.630	10.179	4.294	25.923	15.443
2.358	2.143	112.554	21.198	5.568	12.294	40.363	8.208	5.096	6.143	3.462	16.956	8.877
4.547	5.704	328.567	65.289	13.234	38.241	124.533	24.122	15.297	18.005	8.378	44.633	26.477
1.107	1.146	71.917	13.194	3.351	8.945	26.335	5.862	3.372	4.538	2.357	10.886	6.658
1.052	1.054	51.885	10.324	1.977	6.270	18.734	4.450	2.262	3.362	2.300	7.536	4.516
1.058	1.129	34.324	6.855	2.258	4.464	12.588	3.064	2.123	2.088	1.117	5.635	3.208
1.167	1.156	35.692	7.030	2.237	4.643	13.126	3.275	2.291	2.161	1.103	5.260	3.349
1.163	1.042	37.653	8.174	2.283	5.567	13.725	3.216	1.953	2.248	1.017	6.626	3.097
1.066	1.119	2.096	1.152	1.061	1.109	1.008	1.004	1.160	1.012	1.076	1.071	1.089
1.144	1.097	49.467	9.080	2.274	6.734	17.857	4.308	2.199	3.184	2.197	7.155	4.333
1.111	1.032	16.792	3.136	1.041	2.333	6.794	2.132	1.011	1.040	1.112	3.236	2.305
1.079	1.084	34.676	7.963	2.298	4.718	13.565	3.418	2.064	2.310	1.044	5.374	3.211
2.108	2.182	121.334	21.974	4.916	13.567	42.009	8.949	5.557	7.170	3.190	17.343	9.453

Table 8. Traffic Matrix for AT&T Network, Columns[14-26]

2.101	2.344	4.748	1.124	1.009	1.056	1.110	1.117	1.109	1.071	1.111	1.058	2.336
3.345	2.273	5.230	1.131	1.041	1.055	1.180	1.147	1.123	1.139	1.099	1.067	2.342
166.259	106.317	342.008	72.268	49.963	34.087	35.845	39.854	2.384	52.193	16.516	37.774	114.978
35.645	20.875	63.196	13.024	9.257	6.360	6.453	7.706	1.209	10.147	3.207	7.726	23.350
7.921	5.250	13.880	3.231	2.335	2.036	2.194	2.191	1.138	2.504	1.203	1.998	5.673
21.235	12.511	42.716	8.709	7.093	4.591	4.417	5.226	1.024	6.547	2.175	4.802	13.907
62.225	40.339	115.731	25.079	18.920	13.024	14.492	13.765	1.119	19.101	6.133	13.631	43.392
13.408	8.047	25.685	5.559	4.091	3.041	3.287	3.294	1.068	3.967	2.166	3.045	8.956
8.217	5.851	15.432	3.577	2.272	2.156	2.303	2.308	1.042	2.139	1.068	2.158	5.643
8.792	6.557	15.761	4.426	2.878	2.184	2.191	2.134	1.138	2.988	1.093	2.222	6.396
4.684	3.401	8.835	2.271	2.141	0.971	1.055	1.181	1.126	2.188	1.078	1.057	3.522
25.900	18.114	51.681	12.144	8.932	5.126	5.585	6.331	1.089	7.193	3.160	5.470	18.457
14.301	11.021	28.943	6.488	4.496	3.000	3.473	3.406	1.079	4.252	2.008	3.355	8.722
0.000	25.906	77.219	18.242	12.441	9.287	8.067	9.251	1.051	11.554	4.301	9.254	27.037
27.168	0.000	49.948	11.161	8.041	5.460	5.200	6.313	0.998	7.640	3.044	5.113	18.395
73.818	50.196	0.000	30.545	22.772	17.418	16.341	7.517	1.010	24.194	7.663	16.693	48.477
16.276	10.613	29.878	0.000	5.784	3.392	4.501	4.448	1.035	5.926	2.299	4.258	13.048
13.578	7.916	22.066	4.845	0.000	3.378	3.147	3.608	1.077	4.494	2.460	3.328	8.902
8.399	5.486	16.985	3.242	3.521	0.000	2.128	2.212	1.109	3.207	1.128	2.347	6.813
8.082	5.819	18.020	4.265	3.466	2.231	0.000	2.154	1.112	3.329	1.122	2.412	6.227
9.722	6.817	19.960	4.310	3.335	2.053	2.053	0.000	1.031	3.172	1.157	2.281	6.665
1.078	1.014	1.072	1.022	1.083	1.060	1.157	1.034	0.000	1.048	1.123	1.144	1.193
11.397	6.533	20.584	5.562	4.456	3.190	3.442	3.284	1.128	0.000	2.342	3.276	9.119
4.155	3.389	7.791	2.222	2.179	1.110	1.064	1.124	1.180	1.990	0.000	1.094	3.430
8.927	5.222	16.952	4.241	3.286	2.002	2.272	2.167	1.071	3.289	1.107	0.000	6.799
25.868	19.549	49.104	12.558	9.107	6.142	7.027	6.544	1.099	9.095	3.191	6.762	0.000