Gain Scheduled Routing in Multi-Service Networks

Jakob Carlström and Ernst Nordström

Department of Computer Systems, Information Technology, Uppsala University,
Box 325, SE - 751 05 Uppsala, Sweden
Email: {jakobc, ernstn}@DoCS.UU.SE

Abstract

This paper presents gain scheduled routing, a new scheme for reward-maximization in multi-service loss networks. Per-link control policies and their expected future rewards are precomputed using Markov decision theory and stored in lookup tables. Selection of appropriate table entries is based on measured or predicted call arrival rates. An approximation is introduced, which makes it possible to use identical lookup tables on links having the same capacities, although the links carry calls with different link reward parameters. Simulations with Poisson arrival processes show that gain scheduled routing offers significant improvement of the average reward rate, compared to least loaded path routing. Promising results are also obtained in simulations with self-similar call traffic.

1 Introduction

In multi-service connection-oriented communications networks, such as Asynchronous Transfer Mode (ATM) or Synchronous Transfer Mode (STM) networks, resource control is of crucial importance for the network operator as well as the network users. The objective of resource control is to maintain the service quality for network calls while maximizing the average reward rate (revenue) obtained by the network operator.

For call traffic, service quality (grade of service) is measured in terms of call blocking probabilities, and the key resource to control is bandwidth. If a uniform call charging policy is used, the average reward rate is proportional to the long-term bandwidth utilization. Network routing and call admission control (CAC) are two important resource control problems in broadband networks.

Reinforcement learning offers a framework for optimal CAC and routing. Two types of reinforcement learning algorithms have been proposed for these problems: temporal-difference (TD) methods, which incrementally update the control policy from samples of the controlled system [SB98], and dynamic programming methods [Ber95], which explicitly model the decision task as a Markov decision process.

Marbach et al. successfully applied TD methods to a CAC and routing problem where the size of the state space makes dynamic programming infeasible [MMT00]. Tong and Brown [TB00] demonstrated that TD methods are able to incorporate quality-of-service constraints in the objective of CAC and routing. Two examples of research on dynamic programming for CAC and routing are Dziong's and Masons scheme [DM94], and the efficient approximative solution by Krishnan and Huebner-Szabo de Bucs [KH97].

A standard assumption when applying dynamic programming to CAC and routing is that calls arrive according to (constant-rate) Poisson processes. The memory-less property of the Poisson process makes the models of the dynamics relatively simple. However call arrival processes in many types of networks can be statistically self-similar [PF94, FGWK98]. This makes the application of Markov decision theory more complicated, compared to the case of Poisson traffic.

In a previous paper [Car00a], we presented gain scheduling for CAC in the single-link case. Control policies and their associated relative value functions (which estimate the future reward of control decisions) were precomputed and stored in a lookup table. These control policies were employed in admission control of self-similar call arrival processes, basing the selection of appropriate policies on predicted future call arrival rates.

In this paper, gain scheduling is applied to network routing and CAC. By decomposition of the problem, the control policies are not required to be adapted on-line. We evaluate the performance of gain scheduled routing in simulations, using both Poisson and self-similar call arrival processes.

The organization is as follows: Section 2 formulates the CAC and routing problem in terms of objective and constraints. Section 3 provides the parts of the mathematical modeling which are needed for the remaining sections. Further details are given in the appendix. Section 4 presents the gain scheduling scheme. Section 5 discusses how to modify the scheme for bursty call arrival processes. Section 6 gives the numerical results, quantifying the approximation errors and the performance of the proposed scheme. Section 7 summarizes the properties of gain scheduled routing. Finally, section 8 concludes the paper.

2 Problem Formulation

A multi-service network can be described as a set of nodes and a set of links connecting the nodes. The multi-service network studied in this paper is offered calls indexed by their class $j \in J$, $J = \{1, ..., K\}$. The j-th class is characterized by:

- Origin-destination (OD) node pair
- Bandwidth requirement, b_j
- Set of alternative routes, W_j
- Reward parameter, r_j — expected reward for accepting a class-j call
- Call arrival process
- Distribution of the holding time

As the basis for modeling, we assume that the call arrival process is Poissonian with rate λ_j, and that call holding times are exponentially distributed with mean $1/\mu_j$. We further assume that the reward parameters r_j are given by:

$$r_j = r_j' v_j \tag{1}$$

where r_j' is a normalized reward parameter and v_j is the average number of data units in class-j call:

$$v_j = b_j/\mu_j \tag{2}$$

The task is to find an optimal CAC and routing policy π^*, which maximizes the mean reward from the network, defined as:

$$\overline{R}(\pi) = \sum_{j \in J} r_j \overline{\lambda}_j = \sum_{j \in J} r_j' \, v_j \, \overline{\lambda}_j \tag{3}$$

where $\overline{\lambda}_j$ denotes the average rate of accepted class-j calls. To achieve this, CAC may reject call requests, even if the available capacity in the network is sufficient to accept the call.

For a solution to be feasible in terms of implementation and operation, it should should fulfil the following criteria:

- The computational complexity should be sufficiently low to give a short call set-up delay (in the millisecond range).
- The memory requirements needed should be sufficiently low.

3 Reward-Maximization Framework

We propose the application of reinforcement learning to the problem of finding optimal CAC and routing rules or policies. The behavior of the network can be described as a semi-Markov decision process (SMDP). Transitions between states in the SMDP Markov chain, which evolves in continuous time, are controlled by actions (control decisions). State-dependent reward is delivered to the system every time a state change occurs. In the following, we describe the CAC and routing framework for multi-service loss networks, introduced in [DM94].

The exact SMDP solution of the CAC and routing task has a large network state and policy space, even for moderate-size networks, which prevents its use in real multi-service networks. The network is therefore decomposed into a set of links, assumed to have independent traffic and reward processes. Instead of solving one large network SMDP, a set of small link SMDPs is solved. The results of the link analysis are then combined to produce a CAC and routing decision on the network level.

In the decomposed SMDP solution, the state of a link is given by the vector $x=(x_1, ..., x_K)$, where x_j denotes the number of active link calls from class j. The state space is given by the set of state vectors which fulfil the link capacity constraint:

$$\sum_{j \in J} x_j b_j \leq C_l \tag{4}$$

where C_l denotes the capacity of link l.

The action in each state is represented by a vector $a=(a_1, ..., a_K)$ which controls the admission of presumptive class-j call requests. The per-class decisions a_j can take on two values: 0 (reject) and 1 (accept). The set of permissible actions at a state only contains reject decisions for class-j arrivals that would violate the link capacity constraint.

The *relative value function* is an important concept in SMDP theory. The relative value function maps each state in the state space to an estimate of the difference in future reward earnings when starting in the given state, compared to a reference state. The relative value function is a function of the state x and the CAC and routing policy π, $v_l=v_l(x, \pi)$, computed for link capacity C_l and per-class bandwidth requirements $b=(b_1, ..., b_K)$, link call arrival rates $\lambda_l=(\lambda_{l1}, ..., \lambda_{lK})$ call departure rates $\mu=(\mu_1, ..., \mu_K)$ and reward parameters $r=(r_1, ..., r_K)$. The relative value function can be obtained by solving a sparse linear equation system, as described in the appendix.

The link call arrival rate is a state-dependent function of the CAC and routing policy π:

$$\lambda_{lj}(x,\pi) = \lambda_{jk}(\pi)\, f_{lj}(x,\pi) \prod_{c \in k \setminus \{l\}} (1 - B_{cj}(\pi)) \tag{5}$$

where k is uniquely specified by l and j, $B_{cj}(\pi)$ denotes the probability that link c does not have sufficient capacity to accept a class-j call, $\lambda_{jk}(\pi)$ denotes the arrival rate of class-j calls to route k, and $f_{lj}(x,\pi)$ denotes a filtering probability that the route net-gain is positive. The filtering probability can be computed from link state distributions. However, our experiments show that $f_{lj}(x,\pi)$ can be approximated with one without reducing the average reward rate. In the following, we assume that the link call arrival rates are not state-dependent; that is, given by $\lambda_{lj}(\pi)$. A load sharing model is used to compute $\lambda_{jk}(\pi)$:

$$\lambda_{jk}(\pi) = \lambda_j \frac{\bar{\lambda}_{jk}(\pi)}{\sum_{h \in W_j} \bar{\lambda}_{jh}(\pi)} \tag{6}$$

where $\bar{\lambda}_{jk}(\pi)$ denotes the rate of accepted class-j calls to route k, and W_j denotes the set of alternative routes between the origin-destination node pair.

The link net-gain due to admission of a class-j calls is the difference in relative values before and after the admission:

$$g_{lj}(x,\pi) = v_l(x + \delta_j, \pi) - v_l(x,\pi) \tag{7}$$

where δ_j denotes a vector of zeros except for a one in position j. The link net gain expresses the increase in long-term reward due to accepting a class-j call at state. The route net-gain is obtained as a sum of link net-gains:

$$g_j(y,\pi) = \sum_{l \in k} g_{lj}(x,\pi) \tag{8}$$

where $y=\{x\}$ denotes the network state in the decomposed network model, and k denotes the route index prescribed by the policy π, and $l \in k$ means that l is one of the links in route k.

The CAC and routing rule is to accept the call on the route with maximum route net-gain, and reject the call if the route net-gain is negative.

In order to reduce the computational complexity on a link, call classes j having the same bandwidth requirements and holding times are aggre-

gated into the same category i. The set of categories on link l is denoted by I_l. In many cases, this significantly reduces the dimension of the state vector. In the case of class aggregation, the link call arrival rate for category i is a sum of per-class link call arrival rates:

$$\lambda_{li}(\pi) = \sum_{j \in i} \lambda_{lj} \tag{9}$$

Further, the route net-gain becomes:

$$g_j(y, \pi) = r_j - \sum_{l \in k} p_{li}(x, \pi),\ j \in i \tag{10}$$

where $j \in i$ means that class j is aggregated into category i, and the *link shadow price*, $p_{li}(x,\pi)$, is defined as:

$$p_{li}(x, \pi) = r_{li}(\pi) - g_{li}(x, \pi) \tag{11}$$

which can be interpreted as the expected cost for accepting an i-th category call in state x. The link reward parameter $r_{li}(\pi)$ for the i-th category is based on aggregation of the reward process:

$$r_{li}(\pi) = \frac{\sum_{j \in i} r_{lj}(\pi) \bar{\lambda}_{lj}(\pi)}{\sum_{j \in i} \bar{\lambda}_{lj}(\pi)} \tag{12}$$

where $\bar{\lambda}_{lj}$ denotes the average acceptance rate of class-j calls on link l. The per-class link reward parameters $r_{lj}(\pi)$ fulfils the obvious constraint:

$$r_j = \sum_{l \in k} r_{lj}(\pi) \tag{13}$$

Concerning the actual evaluation of the link call parameters different models are possible. In this paper we use a simple model:

$$r_{lj}(\pi) = r_j / L, \tag{14}$$

where L denotes the number of links the call's route.

The policy iteration algorithm for determining the optimal CAC and routing policy is:

1. Initialize the relative values $v_l(x,\pi)$ in a way that makes all link-net gains in states with permissible admission positive.
2. Measure the per-route call acceptance rates $\bar{\lambda}_{kj}$ and per-link call blocking probabilities $B_{lj}(\pi)$ while employing the maximum net-gain routing rule.

3. a. Identify the link SMDPs by computing the aggregate link reward parameters $r_{li}(\pi)$ and link call arrival rates $\lambda_{li}(\pi)$.
   b. Determine the relative values $v_l(x,\pi)$ for the current routing policy π.
   c. Improve the link CAC policies π_l based on the new relative values.
4. Repeat from 2 until the average reward per time unit converges.

The computational complexity of the policy iteration is dominated by the determination of the relative values. The procedure has complexity $O(N^p)$ where N is the number of states in the state space and p is a characteristic of the method used for value determination (typically $p \leq 3$).

In the next section, we present a CAC and routing scheme based on gain scheduling, which is an alternative to the above policy iteration algorithm.

4 Gain Scheduling

Gain scheduling is a technique from control theory, where the parameters of a controller are changed as a function of varying operating conditions [ÅW95].

The approach taken here is to store relative value functions in a lookup table, where per-category offered call arrival rates $\lambda_{li}(\pi)$ serve as indices. For each link, such a table is precomputed for points in a grid covering the region of possible arrival rate vectors, λ_l. In operation, table indices pointing out a relative value function are computed by dividing the arrival rates by the distance between grid points. Thus, the need for repeated determination of relative values and policies in the original scheme [DM94] is eliminated.

In the network, exponentially weighted running averages of per-link blocking probabilities $B_{li}(\pi)$ and per-route call acceptance rates $\overline{\lambda}_{kj}(\pi)$ are maintained. Indexing call acceptance and call departure events on a link l by m, their occurrence time is denoted by $t(m)$. The estimate of the blocking probability, here expressed as the *time congestion*, after event number m+1 is

$$B_{li}(\pi, m+1) = (1-\beta_1)B_{li}(\pi, m) + \beta_1 \theta(x(m), i)[t(m+1) - t(m)], \tag{15}$$

where $0 < \beta_1 < 1$ is a forgetting factor, $x(m)$ is the link state *after* the event at time $t(m)$, and $\theta(x, i)$ indicates whether x is a blocking state for category i or not:

$$\theta(x,i) = \begin{cases} 0 & \text{if } \sum_{s \in I} x_s b_s + b_i \leq C_l \\ 1 & \text{otherwise} \end{cases}. \tag{16}$$

Using index n for the n-th class-j call set-up on route k, occurring at time $t(n)$, the estimate of the reciprocal call acceptance rate is

$$\bar{\lambda}_{kj}^{-1}(\pi, n+1) = (1-\beta_2)\bar{\lambda}_{kj}^{-1}(\pi, n) + \beta_2[t(n+1) - t(n)], \tag{17}$$

where $0 < \beta_2 < 1$ is a forgetting factor. From this estimate, the offered call arrival rate are obtained by a load sharing approximation, thinning the original arrival process:

$$\lambda_{kj}(\pi, n) = \lambda_j(n) \frac{\bar{\lambda}_{kj}(\pi, n)}{\sum_{h \in W_j} \bar{\lambda}_{hj}(\pi, n)} \tag{18}$$

From equations (5) and (9), the per-link, per-category offered rates $\lambda_{li}(\pi)$ are obtained, where $\lambda_j(n)$ is the class-j arrival rate at time $t(n)$ (discussed below).

It is possible to use a set of relative value functions computed for one link l_1 at another link l_2 if $C_{l_1} = C_{l_2}$ and $I_{l_1} = I_{l_2}$, under the following assumption:

$$r_{l_1 i}(\pi) = r_{l_2 i}(\pi) \; \forall i \in I_l. \tag{19}$$

However, as the load sharing coefficients depend on the number of links traversed by category-i calls (equation 14), assumption (19) is unrealistic. Combining (12) and (14), and denoting the number of links in route k by L_k, the reward parameters on link l can be written as follows (denoting the normalized reward parameter for category-i calls by $r_{li}'(\pi)$):

$$r_{li}(\pi) = \frac{\sum_{j \in i} \bar{\lambda}_{kj}(\pi) r_j'}{\sum_{j \in i} \bar{\lambda}_{kj}(\pi) L_k} \nu_i = r_{li}'(\pi) \nu_i, \tag{20}$$

where ν_i is the mean number of data units of a category-i call, and k is uniquely determined by $l \in k$ and $k \in W_j$. We define the aggregate normalized reward parameter on link l, $\bar{r}_l'(\pi)$:

$$\bar{r}_l'(\pi) = \frac{\sum_{i \in I} \bar{\lambda}_{li} r_{li}'(\pi) \nu_i}{\sum_{i \in I} \bar{\lambda}_{li} \nu_i}. \tag{21}$$

In our proposed gain scheduling scheme, a table of relative value functions is computed for unity normalized reward parameters ($r_{li}'(\pi) \equiv 1 \; \forall i \in I_l$). We denote by $w_l(x, \pi)$ such a relative value function, computed for some arrival rate vector, λ_l. When this value function is looked up by the gain scheduling scheme, the actual values are approximated by using $\bar{r}_l'(\pi)$ as a multiplicative adjustment:

$$v_l(x, \pi) \approx \bar{r}_l'(\pi) \, w_l(x, \pi). \tag{22}$$

As can be seen from Bellman's optimality equation (A12), this is equivalent to approximating $r_{li}'(\pi)$ by $\bar{r}_l'(\pi)$ for all $i \in I_l$.

The validity of this approximation depends on the policy and the class properties. We can, however make some general observations:

For a fixed policy π and the definition of the aggregate normalized reward parameter (21) it follows that

$$\bar{R}_l(\pi) = \sum_{i \in I_l} r_{li}'(\pi) \nu_i \bar{\lambda}_{li}(\pi) = \bar{r}_l'(\pi) \sum_{i \in I_l} \nu_i \bar{\lambda}_{li}(\pi) \; . \tag{23}$$

Hence, the proposed approximation yields the correct long-term reward in this case. However, note that as the policy is given by the relative value functions, some routing decisions may be different with gain scheduling, compared to [DM94].

Further, we note that the approximation error of the relative values (22) vanishes in two cases:

1. The offered rate is non-zero for one category i, and zero for all other categories.
2. The normalized reward parameters of all categories are equal.

The equality is trivial in the the first case, and the second case can be shown to hold by inserting (21) in Bellman's optimality equation (A12). For other cases, the approximation error and its effects on routing performance are numerically quantified in section 6.

5 Modification to Bursty Arrival Processes

In the load sharing approximation (18), the per-class offered rates $\lambda_j(n)$ are assumed to be constants, λ_j, on a time scale of minutes to hours. As

the traffic characteristics in networks normally vary with the time of day and other factors, a measurement system should be installed in the network, updating λ_j at a regular basis. The gain scheduling scheme automatically adjusts the routing policy with respect to the arrival rates.

In networks where the call arrival process is not Poisson, but has temporal correlations causing bursty traffic, the Markov property of the state descriptor used above no longer holds. One approach to incorporating the memory of the arrival process into the model is to augment the state descriptor with statistical measures from the arrival process.

In [CN99], this was done for self-similar call arrival processes. Per-class exponentially weighted running averages of the call inter-arrival time, computed on multiple time scales, were included in the state descriptor used for link admission control. As this results in a continuous-valued state descriptor, where the state-transition probabilities are not readily available, simulation-based temporal-difference learning [SB98] was employed instead of dynamic programming, such as example policy iteration.

Augmenting the state descriptor used by temporal-difference learning increases the long-term reward, but the convergence of the method is slow. In [Car00a], we therefore proposed decomposition of the link admission control problem by gain scheduling. In this scheme, nonlinear neural networks predict near-future per-class call arrival rates. These predictions are used to select value functions in gain scheduling, thus approximating the arrival processes by modulated Poisson processes.

In section 6.2, we evaluate the performance of gain scheduled (GS) routing of Poisson call traffic as well as bursty call traffic. For bursty traffic, exemplified by self-similar arrival processes, we modify the scheme by adding neural-network based time-series predictors, replacing $\lambda_j(n)$ by predictions $\hat{\lambda}_j(n)$.

6 Numerical Results

The performance of GS routing was evaluated in simulations. Each OD pair was offered traffic from two classes: one wide-band class, with bandwidth requirement $b_{WB} = 6$ units per second, and one narrow-band class, with bandwidth requirement $b_{NB} = 1$ unit per second. The mean holding time was set to 1 for all classes, $1/\mu_{NB} = 1/\mu_{WB} = 1$, and a uniform call charging policy ($r_j' = 1$ for all $j \in \mathcal{J}$) was assumed.

6.1 Approximation of Relative Values

To quantify the approximation error induced by the aggregation of normalized reward parameters on the links, the difference between the relative values in the original routing scheme [DM94] and the gain scheduling scheme was computed, varying the normalized reward parameters on a link. Under the assumption of uniform call charging, the difference in the per-category reward parameters on a link results from calls belonging to different classes being routed along routes of different lengths.

Figure 1 plots the weighted mean relative root mean square error of the approximation, computed as

$$\epsilon_l = \left[\sum_{x \in X} p_l(x) \left[\frac{v_l^a(x) - v_l(x)}{v_l(x)} \right]^2 \right]^{1/2}, \tag{24}$$

where $p_l(x)$ is the limit probability of the state of link l being x, computed as described in [Car00a], and $v_l^a(x)$ is the approximated relative value function resulting from aggregation of normalized reward parameters. Narrow-band classes are aggregated into category 1, and wide-band classes into category 2. The approximation error is shown for varying ratio between per-category arrival rates, expressed as the angle between per-category loads:

$$\phi = \arctan \frac{\lambda_{l2} \nu_2}{\lambda_{l1} \nu_1}, \tag{25}$$

and for the ratio between the per-category normalized reward parameters ranging from 1/3 to 3. A ratio of 1/3 corresponds to all narrow-band calls being routed along routes consisting of three links, and all wide-band calls getting a single-link route, for example. A ratio of 3 corresponds to the reverse situation.

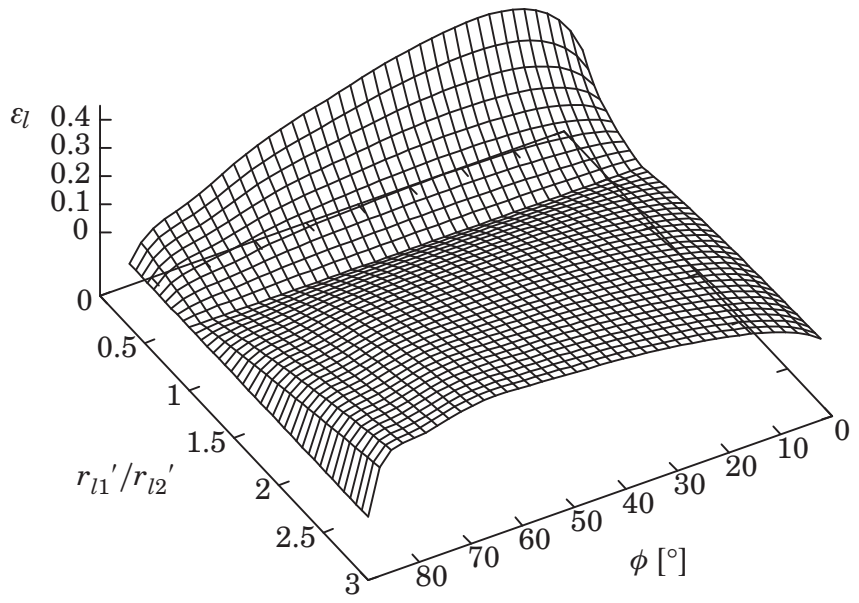

Figure 1: Relative approximation error induced by aggregation of the normalized reward parameters for varying ratio between per-category normalized reward parameters and varying arrival rate ratios (expressed as the angle between per-category loads).

The resulting approximation error ranges from zero for the cases discussed in the previous section (note the "fold" for $r_{l1}'/r_{l2}' = 1$) up to 45%.

6.2 Routing

We simulated routing using two models of multi-service networks: an 8-node network and a 4-node network, shown in Figure 2.

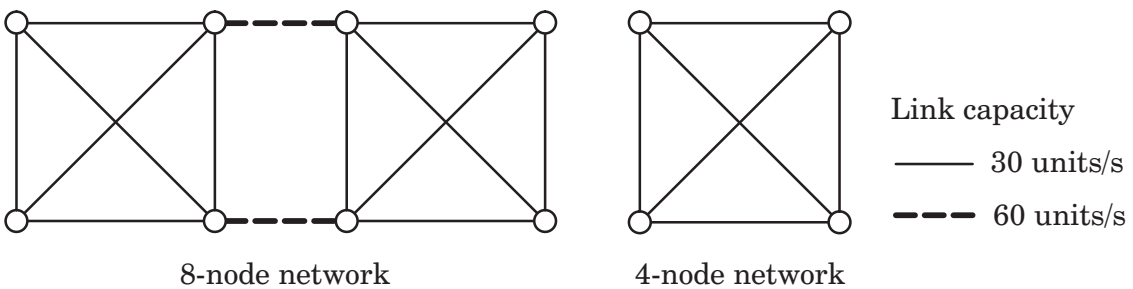

Figure 2: The network models used in simulations

Six routing scenarios were simulated, differing in terms of the network used, and in terms of load distribution and call arrival process. The studied cases were:

1. Symmetric load and asymmetric load.
2. Poisson arrival process and bursty (self-similar) arrival process.

Symmetric load means that all narrow-band classes have the same call arrival rates, and all wide-band classes have the same arrival rates. Under asymmetric load this is not the case. Table 1 summarizes the scenarios.

Mnemonic name	**Arrival process**	**Load distribution**	**Network**	**Traffic**
PS4	Poisson	Symmetric	4-node	185
PA4	Poisson	Asymmetric	4-node	160
PS8	Poisson	Symmetric	8-node	345
BS4	Bursty	Symmetric	4-node	168
BA4	Bursty	Asymmetric	4-node	134
BS8	Bursty	Symmetric	8-node	333

Table 1: The simulation scenarios

Bursty call traffic was generated using a FARIMA model, generating statistically self-similar call arrival processes. The Hurst parameter, which expresses the degree of self-similarity of a stochastic process, was set to 0.85. For a full description of the bursty traffic model, see [Car00a]. Radial-basis function neural networks [Hay99] were used to predict future per-class call arrival rates, getting two exponentially weighted running averages per class as inputs. The running averages were computed using two different forgetting factors, optimized by linear search with low prediction error as performance criterion. For the details of this time-series prediction, see [Car00a].

Traffic is expressed as the product of per-class traffic in Erlang multiplied by its bandwidth requirement b_j. The traffic was tuned to yield approximately 5% reward loss in the simulations with GS routing. The reward loss is defined as $1 - \overline{R}(\pi)/R$, where R is the total reward rate offered to the network.

Two tables of relative value functions were precomputed, one for links with capacity 30 units/s and one for links with capacity 60 units/s. Relative value functions were computed for the points of a uniform grid dividing the arrival rate of each category (wide-band and narrow-band) into 30 intervals. This resolution of the grid was assumed to be sufficiently fine, making no attempt to minimize the number of grid points.

For each scenario, simulations were run for varying ratios $\lambda_{NB}/\lambda_{WB}$ between the per-class arrival rates for each OD pair. The diagrams in Figure 3 and Figure 4 compare the reward loss for GS routing to that of *least loaded path* (LLP) routing, a competitive state-dependent routing method [Dzi97]. In LLP routing, the single-link route is chosen, if such a route exists and the call is permitted by the capacity constraint. Otherwise, the alternative route having the largest unused capacity on the bottleneck link is selected.

The maximum path length allowed was two links in the 4-node network, and three links in the 8-node network.

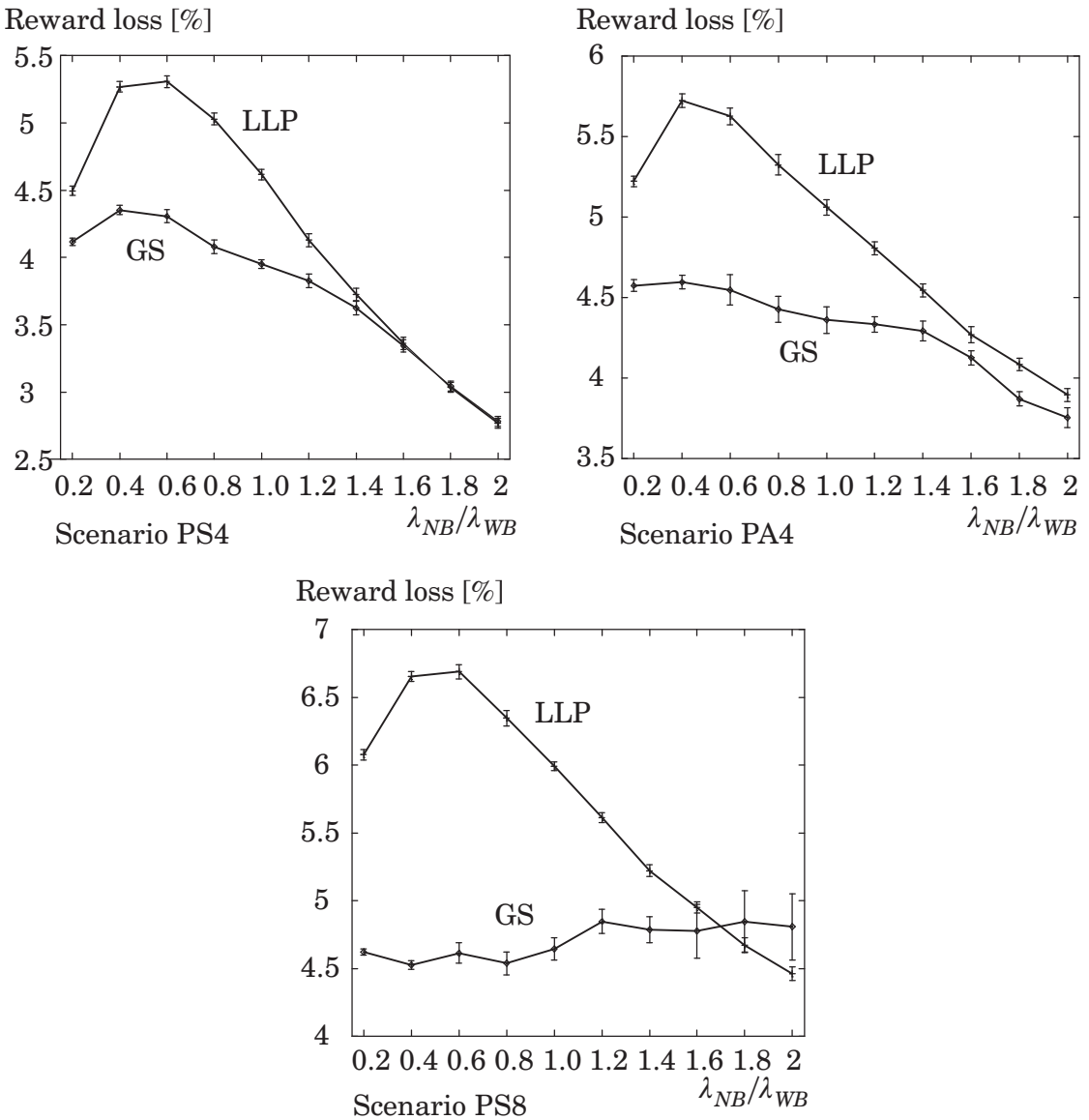

Figure 3: Reward loss in simulations of scenarios with Poisson arrival processes.

In the simulations, the forgetting factors used to in the updating of blocking probabilities and carried arrival rates were $\beta_1 = \beta_2 = 0.998$. Each curve is based on 10 simulation runs, and the error bars are 95% confidence intervals.

For bursty call traffic, the variations in the traffic are not evened out by long simulations run. This is an inherent property of self-similar arrival processes. To exclude the effects of traffic variance from the error bars, the same set of self-similar traffic traces were used in the simulations of LLP and GS routing. The diagrams to the right in Figure 4 show the average pairwise difference in reward loss, computed for runs with the same traffic traces.

6.3 Result Analysis

With Poisson traffic, GS routing outperforms LLP routing. The difference in reward loss is highest for low arrival rate ratios $\lambda_{NB}/\lambda_{WB}$, where the reward loss of LLP routing is up to 45% higher than that of GS routing. With bursty traffic, LLP routing yields up to 39% higher reward loss than LLP for low arrival rate ratios. However, for high arrival rate ratios, the reward loss of GS routing is up to 41% higher than that of LLP routing for the simulated arrival rate ratios.

We currently have no conclusive explanation to this performance degradation of GS routing. The approximation of bursty call traffic by modulated Poisson processes is one possible reason, although no similar effects were observed in simulations of admission control in the single-link case [Car00a]. Further, the load-sharing model, where an arrival process is assumed to be thinned by stationary route selection probabilities, may cause approximation errors. With bursty call traffic, the route selection probabilities may differ between periods of low short-term call arrival rates and periods with high arrival rates.

We also make an observation regarding the nominal network traffic yielding reward loss of about 5% for the two different call arrival processes, in the case of GS routing. On average, this nominal traffic is 8% lower for self-similar call arrival processes than for Poisson arrival processes.

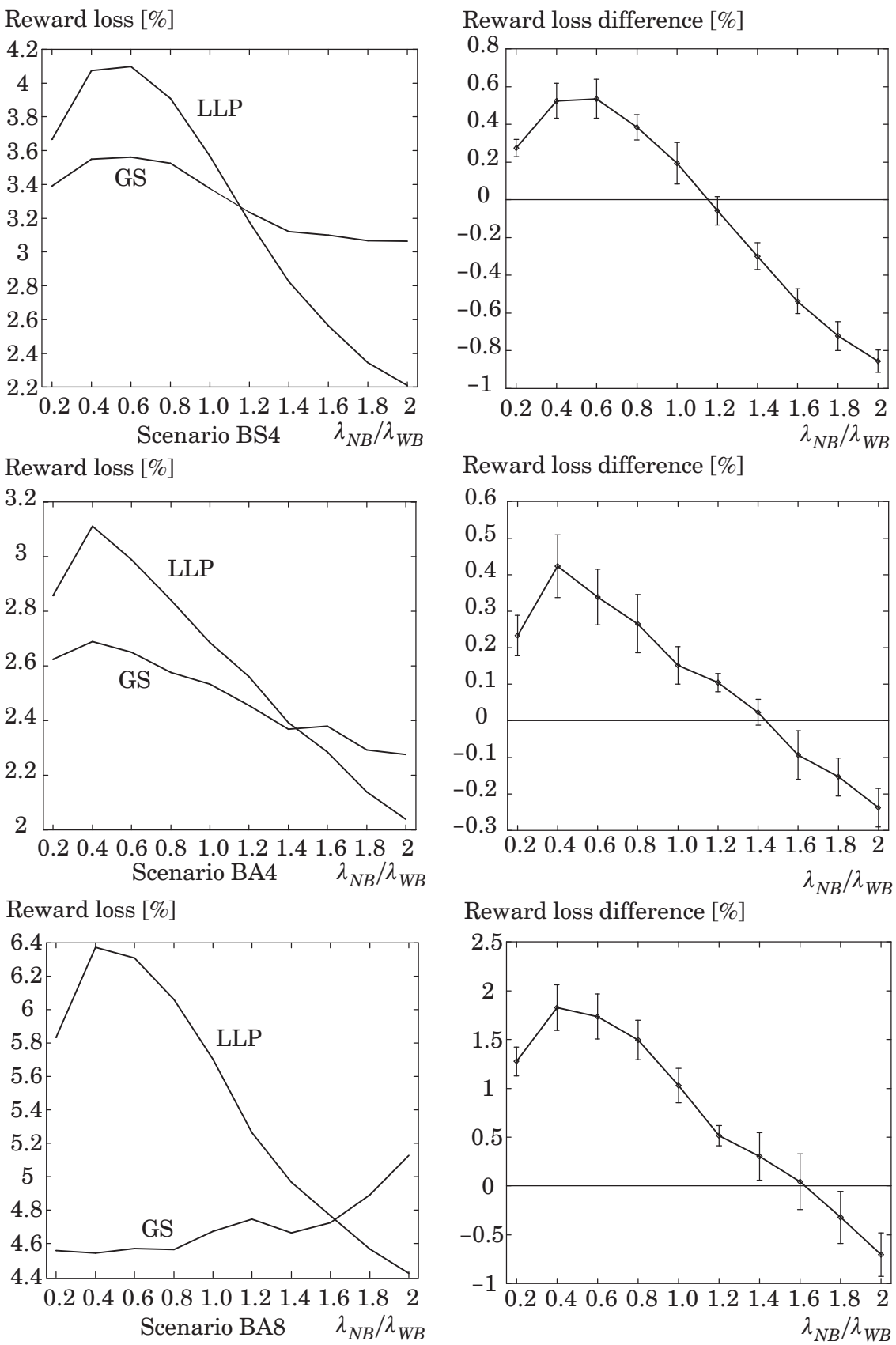

Figure 4: Reward loss in simulation of scenarios with bursty call arrival processes (left), and the difference between the reward loss for LLP and the reward loss for GS routing (right).

7 Discussion

Compared to Dziong's and Mason's SMDP scheme for routing, gain scheduled routing has the following advantages:

1. All the necessary SMDP computations, which determine the relative value functions for table grid points, are done off-line, prior to the operation phase. In operation, real-time routing decisions are based on simple table lookup.
2. Fast convergence of the routing policy, as there is no need for iterated measurement phases.
3. Gain scheduled routing can exploit the temporal correlations of non-Poisson traffic, provided the sources are equipped with suitable time-series predictors.

Gain scheduled routing also raises a few concerns:

1. The number of entries in the relative value function table is a potential bottleneck of the routing scheme. However, this problem can be addressed by error-driven design of the grid and interpolation of relative values, as described in [Car00a].
2. The need for low-complexity computation of the relative values at the table grid points puts a limit on the number of states in the link state space. However, in case of large state space, state aggregation method [KH97] offers a computationally feasible solution alternative.
3. In case the average number of hops per category is different on a link an approximation error in the estimation of relative values is introduced.

In the simulations, GS routing converged smoothly in the case of Poisson arrival processes. With self-similar traffic, fluctuations due to long-term burstiness in the traffic were observed. However, the convergence has not been proven, neither for GS routing nor for the policy iteration scheme proposed by Dziong and Mason [DM94].

8 Conclusion

We have presented gain scheduled routing, a new scheme for reward-maximizing CAC and routing in multi-service loss networks. In this

scheme, per-link relative value functions are precomputed and stored in lookup tables. Selection of relative value functions is based on arrival rates of call categories, defined by common bandwidth requirement and expected holding time. The approximation of relative values is adjusted for varying route lengths by aggregating reward parameters. In the paper, the approximation error induced by this aggregation was analyzed, analytically and numerically.

In simulations with narrow-band and wide-band Poisson traffic, least loaded path routing was shown to yield up to 45% higher reward loss than gain scheduled routing for low ratios between the arrival rate of narrow-band calls and the arrival rate of wide-band calls. For higher ratios, the difference diminishes.

Further, gain scheduled routing was applied to routing of bursty (self-similar) call arrival processes, approximated by modulated Poisson processes. Neural networks were used to predict the near-future call arrival rates. In simulation comparisons to least loaded path routing, gain scheduling achieved lower reward loss for low arrival rate ratios, but higher reward loss for high arrival rate ratios.

A benefit of gain scheduled routing is that the same table of relative value functions can be used for arbitrary call arrival processes. Only the arrival rate predictors need to be modified, although our neural-network based predictors appear to cope well with a range of arrival processes.

Future improvement of gain scheduled routing includes state-dependent modeling of route selection probabilities in the case of bursty call traffic. In case of Poisson traffic, the scheme should be compared to the scheme in [DM94].

To conclude, gain scheduled routing combines the advantages of SMDP routing with low computational cost in operation. It is also a promising approach to routing of bursty call traffic.

References

[Ber95] D.P. Bertsekas, Dynamic Programming and Optimal Control, Athena Scientific, Belmont, MA., USA, 1995.

[Car00a] J. Carlström, “Predictive Gain Scheduling for Control of Self-Similar Call Traffic in Multi-Service Networks”, Technical Report 2000–008, Department of Information Technology, Uppsala University, 2000.

[Car00b] J. Carlström, “Efficient Approximation of Values in Gain Scheduled Routing”, Technical Report 2000–010, Department of Information Technology, Uppsala University, 2000.

[CN99] J. Carlström and E. Nordström, ”Reinforcement Learning for Control of Self-Similar Call Traffic in Broadband Networks”, in Teletraffic Engineering in a Competitive World, Proceedings of The 16th International Teletraffic Congress (ITC16), P. Key and D. Smith (eds.), Elsevier, pp. 571-580, 1999.

[DM94] Z. Dziong and L. Mason, “Call Admission and Routing in Multi-service Loss Networks”, *IEEE Transactions on Communications*, vol. 42, no. 2/3/4, pp. 2001-2022, Feb./March/April 1994.

[Dzi97] Z. Dziong, ATM Network Resource Management, McGraw-Hill,1997

[PF94] V. Paxson and S. Floyd, ”Wide-Area Traffic: The Failure of Poisson Modeling”, in Proc. of SIGCOMM 94 –8/94, London, UK, 1994, pp. 257-268.

[FGWK98] A. Feldmann, A. C. Gilbert, W. Willinger and T. G. Kurtz, ”The Changing Nature of Network Traffic: Scaling Phenomena”, Computer Communication Review, vol. 28, no. 2, pp. 5-29, April 1998.

[FGWK98] A. Feldmann, A. C. Gilbert, W. Willinger and T. G. Kurtz, ”The Changing Nature of Network Traffic: Scaling Phenomena”, Computer Communication Review, vol. 28, no. 2, pp. 5-29, April 1998.

[Hay99] S. Haykin, *Neural Networks — A Comprehensive Foundation*, 2nd ed., Prentice-Hall, 1999.

[KH97] K.R. Krishnan and F. Huebner-Szabo de Bucs, “Admission Control and State-dependent Routing for Multirate Circuit-Switched Traffic”, *Teletraffic Contributions for the Information Age*, ITC-15, V. Ramaswami et al. (eds.), pp. 1043-1054, Elsevier Science, 1997.

[MMT00] P. Marbach, O. Mihatsch, J.N. Tsitsiklis, “Call Admission Control and Routing in Integrated Service Networks Using Neuro-Dynamic Programming”, *Journal on Selected Areas in Communciations*, vol. 18, no. 2, February 2000.

[SB98] R.S. Sutton and A.G. Barto, Reinforcement Learning: An Introduction, MIT Press, Cambridge, MA, 1998.

[TB00] H. Tong and T.X. Brown, “Adaptive Call Admission Control Under Quality of Service Constraints: A Reinforcement Learning Solution”, *Journal on Selected Areas in Communciations*, vol. 18, no. 2, February 2000.

[ÅW95] K.J. Åström and B. Wittenmark, *Adaptive Control*, 2nd ed., Addison-Wesley, 1995.

Appendix: Computation of Relative Values

The state of each link can be described by a vector $x = (x_1, \ldots, x_K)$, where x_j denotes the number of class-j calls accepted on link l. To simplify the notation, we assume that alternative routes for class j have no common links — this is not a limitation of the approach. The state space is given by:

$$X = \left\{ x : x_j \geq 0 \; \forall j \in J, \sum_{j \in J} x_j b_j \leq C_l \right\} \tag{A1}$$

where C_l denotes the capacity of link l.
The optimal link CAC policy can be computed by the policy iteration algorithm, which consists of two main steps:

- relative value determination
- policy improvement

The first step of policy iteration determines the *relative value* $v_l(x, \pi)$ for all states $x \in X$ and routing policy π by solving a sparse linear equation system:

$$\begin{cases} v_l(x,\pi) = R_l(x,a) - \overline{R}_l(\pi)\tau(x,a) + \sum_{y \in X} p_{xy}(a) v_l(y,\pi), & x \in X \\ v_l(x_r,\pi) = 0, & x_r \in X \end{cases} \tag{A2}$$

where the following quantities need to be specified for link l:

- $a = \pi_l(x)$: the control action in state x, given by the link CAC policy
- $\tau(x,a)$: the expected sojourn time in state x,
- $R_l(x,a)$: the expected reward delivered when leaving state x,
- $p_{xy}(a)$: the transition probability from state x to state y, given that action a is taken in state x,
- x_r: reference state (for example the state corresponding to an empty link).

in order to compute the unknowns:

- $v_l(x,\pi)$: the relative value in state x under routing policy π,
- $\overline{R}_l(\pi)$: the average reward rate under routing policy π.

The *control action* for each link is represented by a vector $a = (a_{1, \ldots,} a_K)$ corresponding to admission decisions for presumptive call requests. The action space A is a given by:

$$A = \left\{ a = \{a_j\} : \; a_j \in \{0, 1\}, \, j \in J \right\} \tag{A3}$$

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

$$A(x) = \left\{ a \in A : a_j = 0 \text{ if } x + \delta_j \notin X, \, j \in J \right\} \tag{A4}$$

where δ_j denotes a vector of zeros except for a one at position j.

The *expected sojourn time* $\tau(x,a)$ in state x is given by:

$$\tau(x, a) = \left\{ \sum_{j \in J} x_j \mu_j + a_j \lambda_{lj}(x, \pi) \right\}^{-1} \tag{A5}$$

The *expected reward* delivered when leaving state x is given by

$$R_l(x, a) = q_l(x) \tau(x, a) \tag{A6}$$

where the reward accumulation rate $q_l(x)$ is:

$$q_l(x) = \sum_{j \in J} r_{lj}(\pi) x_j \mu_j \tag{A7}$$

The *state transition probabilities* from state x to states y are:

$$p_{xy}(a) = \begin{cases} \lambda_{lj}(x, \pi) a_j \tau(x, a), & y = x + \delta_j \in X, \, j \in J \\ x_j \mu_j \tau(x, a), & y = x - \delta_j \in X, \, j \in J \\ 0, & \text{otherwise} \end{cases} \tag{A8}$$

where $\lambda_{lj}(x,\pi)$ denotes the arrival rate of class-j calls to link l in state x under routing policy π. As shown in section 3, the per-link call arrival rates $\lambda_{lj}(x,\pi)$ can be approximated with constant rate values $\lambda_{lj}(\pi)$.

The second step of policy iteration consists of finding the action that maximizes the relative value in each state $x \in X$:

$$a = \underset{u \in A(x)}{\operatorname{argmax}} \left\{ R_l(x, u) - \overline{R}_l(\pi) \tau(x, a) + \sum_{y \in X} p_{xy}(u) v_l(y, \pi) \right\}, \; x \in X \tag{A9}$$

using the relative values $v(x,\pi)$ for the current routing policy π. The set of actions constitute an improved link CAC policy π_l' to be used again in the first step. The policy iteration algorithm terminates when the improved link CAC policy is the same as the current link CAC policy. In the case of finite action and state spaces, the algorithm converges to an optimal policy under the assumption that each stationary policy is *unichain* [Ber95, proposition 3.3]. After some simplification, the second step can be transformed into:

$$a = \underset{u \in A(x)}{\operatorname{argmax}} \left\{ u_j [v_l(x + \delta_j, \pi) - v(x, \pi)] \right\}, \; x \in X \tag{A10}$$

That is, the link CAC policy π_l' for class-j calls maximizes the state-dependent *link net-gain* $g_{lj}(x,\pi)$:

$$g_{lj}(x, \pi) = v_l(z + \delta_j, \pi) - v_l(x, \pi), \tag{A11}$$

Finally, we note that the *Bellman's optimality equation* is valid for the optimal link CAC policy:

$$\begin{cases} v_l^*(x, \pi^*) = \max\limits_{u \in A(x)} \left\{ R(x, u) + \sum\limits_{y \in X} p_{xy}(u) v_l^*(y, \pi^*) \right\} - \overline{R}^*, \; x \in X \setminus \{x_r\} \\ v_l^*(x_r, \pi^*) = 0 \end{cases} \tag{A12}$$

where $v_l^*(x,\pi^*)$ denotes the relative value for the optimal routing policy π^*.