A Restricted Complete Sharing Policy for a Stochastic Knapsack Problem in B-ISDN

Arieh Gavious and Zvi Rosberg

***Abstract*— Consider a *circuit switched* broadband ISDN network that support a variety of traffic classes (e.g., data, voice, video, facsimile), each of which has its own traffic requirement and reward function. We address the problem of dynamically allocating the capacity of each circuit among the traffic classes. As an optimal allocation policy is extremely hard to find, we apply a different methodology by which we bound from above the optimal expected reward, and propose a specific threshold policy—the restricted complete sharing (RCS)—that yields a reward sufficiently close to this bound. The initial parameters of the threshold policy are found with the aid of our bounding technique, and are improved by two iterative procedures. The quality of our policy is demonstrated by several numerical examples.**

I. Introduction and Model Formulation

We CONSIDER a *circuit switched* broadband ISDN network that supports a variety of traffic classes as data, voice, video, facsimile, etc. We assume that the circuits are given, and each one of them can be shared by several communication sessions of various classes. We focus on a generic circuit whose bandwidth is subdivided into N units, at which sessions of class i, $i = 1, \cdots, n$ arrive according to independent Poisson processes whose rates are given by λ_i, $i = 1, \cdots, n$. A class i session requires b_i units and releases them simultaneously after a random time which is exponentially distributed with mean μ_i^{-1}. We assume that all service requirements and arrival instances are independent. After a session is being accepted, it uses its required bandwidth without any interruption. In the case where more than b_i units are available, the system may allocate any arbitrary b_i units. An arriving session that finds less than b_i available bandwidth units, is blocked and is assumed lost. We may reject an arriving session even if there is enough available bandwidth. In this exponential system, the reward resulting from accepting a session can be described in two mathematically equivalent ways. One is an instantaneous reward, where a session of class i that has been accepted for service provides a reward of $r_i > 0$ after service completion. The other is a reward rate $r_i' > 0$ accrued during his service. Clearly, $r_i = r_i'/\mu_i$. We refer to this problem as a *stochastic knapsack* after Ross and Tsang who first introduced it in [11]. Note that we assume that the circuit routing is fixed and known.

Our goal is to find a dynamic bandwidth allocation among various traffic classes that maximizes the long-run average system reward. In the communication terminology it is also known as a flow control problem. As the optimization problem is intractable, we apply a different methodology by which we bound from above the optimal expected reward, and propose a specific threshold policy—*restricted complete sharing (RCS)*—that yields a reward sufficiently close to this bound.

This problem has been considered in [11] under a restricted set of allocation policies whose underlying Markov process is reversible, and the optimal policy has been found for the case of two traffic classes. In [12], the same authors have proposed a linear or a dynamic programming approach for an arbitrary number of classes where the action space is taken in a nonstandard manner. Monotonicity properties of the throughput and blocking probabilities in the case where the service and the arrival rates are state dependent, have been derived in [13].

Since the direct approach to the problem has not provided satisfactory results that can be implemented in practical problems, we adopt the approach from [8] and [9], that has been used for other control problems.

The optimal bandwidth allocation problem above can be formulated as a discrete-time Markov-decision process using the uniformization technique in [7].

Let $\boldsymbol{k}(\boldsymbol{t}) = (k_1(t), k_2(t), \cdots, k_n(t))$ be the state of the system at time t, $t = 0, 1, 2, \cdots$ where $k_i(t)$ is the number of sessions from class i that are present at time t. We have that $0 \le k_i(t) \le M_i$, $1 \le i \le n$ where $M_i = \left\lfloor \frac{N}{b_i} \right\rfloor$. (Here, $\lfloor x \rfloor$ is the largest integer that is smaller than or equal to x.)

At every arrival instant t, a decision whether or not to accept the arriving session is taken. This can be generally described by the vector $\boldsymbol{u}(t) = (u_1(t), u_2(t), \cdots, u_n(t))$ where $0 \le u_i(t) \le 1$. For every i, $u_i(t)$ is the probability that a session from class i that arrives at time t is being accepted. The decision action at time t, $\boldsymbol{u}(t)$, may depend on the current state and the history of states and decisions. When in state $\boldsymbol{k}$, an immediate expected reward of $r_i \cdot \mu_i \cdot k_i = r_i' \cdot k_i$ accrued to class i. The overall immediate expected reward is given by $\sum_{i=1}^n r_i \cdot \mu_i \cdot k_i = \sum_{i=1}^n r_i' \cdot k_i$. For every admissible policy π, let $R_i^T(\pi, \boldsymbol{k})$ be the total expected reward accrued to class i until time T, given that at time zero the system starts at state $\boldsymbol{k}$ and policy π is being used (see [10]). Also, let $\overline{R}_i(\pi, \boldsymbol{k}) = \lim\inf_{T\to\infty} \frac{1}{T} R_i^T(\pi, \boldsymbol{k})$. Observe that when the limit exists, $\overline{R}_i(\pi, \boldsymbol{k})$ is r_i times the throughput of sessions of class i. The Markov decision process that we consider is to find a policy that maximizes the weighted sum of the throughputs, $\overline{R}(\pi, \boldsymbol{k}) = \sum_{i=1}^n \overline{R}_i(\pi, \boldsymbol{k})$.

By the uniformization technique of exponential systems, [7], we may convert this problem into a discrete-time Markov-

Paper approved by M. Sidi, the Editor for Communication Networks of the IEEE Communications Society. Manuscript received March 7, 1991; revised December 5, 1991 and June 20, 1992.

A. Gavious is with the Faculty of Management, The Leon Recanati Graduate School of Administration, Tel-Aviv University, Tel-Aviv, Israel.

Z. Rosberg is with IBM Israel, Science and Technology, Technion City, Haifa 32000, Israel.

IEEE Log Number 9401818.

decision process. Since the state space and action space are finite, there exists an optimal stationary policy, [10]. Furthermore, since under every stationary policy the underlying Markov chain is irreducible and aperiodic, the limit above exists and is independent of the initial state $\boldsymbol{k}$. Hereinafter, we consider only stationary policies and omit the initial state $\boldsymbol{k}$ from the notations of $\overline{R}_i(\pi, \boldsymbol{k})$ and $\overline{R}(\pi, \boldsymbol{k})$.

Numerical procedures as value and policy iteration are time consuming and do not provide insight into the problem. Furthermore, they may yield policies which are too complex for practical use. Analytical solutions are mathematically intractable. Therefore, one may apply the technique which has been used in [8] and [9], by which an upper bound is first derived to $\overline{R}(\pi)$. Then, from the bounding technique, a heuristic set of policies is proposed, evaluated and compared to the upper bound.

Due to space constraints we will not peruse the full derivation here. We will present the upper bound, motivate the heuristic set of policies, evaluate them and present numerical comparison to the upper bound and to the performance of the complete sharing (CS) policy. The reader who is interested in the full derivation is referred to [3].

II. A The Upper Bound

It is shown in [3] that for every feasible policy π, $\overline{R}(\pi)$ is bounded by $\overline{R}^*$ defined below.

First re-order the classes in an nonincreasing order according to $\frac{r_i \mu_i}{b_i} = \frac{r'_i}{b_i}$. It is shown in [3] that the index $\frac{r_i \mu_i}{b_i}$ of class i reflects its benefit to the system. Classes with larger $\frac{r_i \mu_i}{b_i}$ are more rewarding to the system. This will be demonstrated in our numerical examples.

Let t^* be the nonnegative integer for which

$$\sum_{i=1}^{t^*} a_i b_i \leq \frac{\rho}{1+\rho} N \leq \sum_{i=1}^{t^*+1} a_i b_i. \tag{1}$$

Here, an empty summation is defined as zero. Now, $\overline{R}^*$ is evaluated by

$$\overline{R}^* = \sum_{i=1}^{n} \frac{r_i \mu_i}{b_i} \sum_{j=1}^{N} p_i^*(j) = \sum_{i=1}^{n} \frac{r'_i}{b_i} \sum_{j=1}^{N} p_i^*(j) \tag{2}$$

where

$$p_i^*(j) = \begin{cases} \frac{a_i \cdot b_i}{N}, \\ \quad 1 \leq i \leq t^* \\ \left(\frac{\rho}{1+\rho} \cdot N - \sum_{l=1}^{t^*} a_l \cdot b_l\right) \cdot \frac{1}{N}, \\ \quad i = t^* + 1 \\ 0, \\ \quad t^* + 1 < i \leq n \end{cases}$$

III. A Threshold Policy

Motivated by the importance order imposed by the indexes $\frac{r_i \mu_i}{b_i}$, $1 \leq i \leq n$, as reflected in the upper bound expression in (2), we define the following class of threshold policies. First, refine the order of the classes by rearranging the classes with the same index $\frac{r_i \mu_i}{b_i}$ according to their b_i's where those with smaller b_i's come first.

For every integer valued vector $\boldsymbol{l} = (l_1, \cdots, l_n)$, $0 \leq l_i \leq M_i$, let $\pi_{\boldsymbol{l}}$ be the threshold policy which accept an arriving session of class i if and only if there are at least b_i available servers, and the number of sessions in the system from that class, k_i, is less than l_i.

A subset of these policies are (t, l)—policies, which for class $i < t$ always accept if room is available, for class $i > t$ always reject, and for class t accept only if there are less than l class i customers, and room is available. That is, $l_i = M_i$ for $i < t$, $l_i = l$ for $i = t$ and $l_i = 0$ for $i > t$.

Note that the (n, M_n)—*policy* is the complete sharing (CS) policy from [11]. A special threshold policy which we propose to use is the $(t^* + 1, l^*$—*policy* where t^* is defined in (1) and l^* is the unique solution to the following inequalities.

Set $\rho_i = \frac{\lambda_i}{\mu_i}$, $i = 1 \cdots n$, and $\rho = \sum_{i=1}^{n} \rho_i$. Denote by a_i the expected number of customers in an $M/M/M_i/M_i$ queueing system under stationary conditions. From [6], $a_i = \left(\sum_{k=0}^{M_i} \frac{\rho_i^k}{k!}\right)^{-1} \cdot \sum_{k=0}^{M_i} \frac{k \cdot \rho_i^k}{k!}$. Now, l^* is the solution to the inequalities $\frac{\sum_{k=1}^{l} \frac{k}{k!} \rho^k}{1+\sum_{j=1}^{l} \frac{1}{j!} \rho^j} \leq p_{t^*+1}(1) < \frac{\sum_{k=1}^{l+1} \frac{k}{k!} \rho^k}{1+\sum_{j=1}^{l+1} \frac{1}{j!} \rho^j}$.

The values $(t^* + 1, l^*)$ are the parameters of a policy in a relaxed system which provides the upper bound $\overline{R}^*$ in [3]. The intuition behind it, is that in each case where the relaxation is not too hard, the bound will be close to the optimal value. Thus, the parameters of the policy in the relaxed system may provide a close to optimal solution in the nonrelaxed system. In other cases, it could serve as a good initial policy.

This policy may be referred as a restricted complete sharing (RCS). Indeed, classes $1, 2, \cdots, t^*$ behave as under CS, class $t^* + 1$ is restricted to at most $l^*_{t^*+1}$ sessions, and all others are not allowed to enter the system.

From [5], the Markov process generated by a threshold policy $\pi_{\boldsymbol{l}}$ is reversible, and has a product form stationary distribution. Hence, the expected reward under an arbitrary threshold policy $\pi_{\boldsymbol{l}}$ is

$$\overline{R}(\pi_{\boldsymbol{l}}) = \frac{\sum_{K \in S_L} \left(\sum_{i=1}^{n} k_i \cdot \mu_i \cdot r_i\right) \prod_{j=1}^{n} \frac{\rho_j^{k_j}}{k_j!}}{\sum_{K \in S_L} \prod_{j=1}^{n} \frac{\rho_j^{k_j}}{k_j!}}$$

The state space, $S_{\boldsymbol{L}}$ is given by $S_{\boldsymbol{L}} = \{K : 0 \leq k_i \leq l_i, \forall i, \sum_{i=1}^{n} k_i \cdot b_i \leq N\}$. The expected reward above can be evaluated by a convolution algorithm (see, e.g., [14]).

As the thresholds l_i^*'s are heuristic, we introduce two improvement algorithms within the set of threshold policies.

Algorithm A: Start with $\pi_{\boldsymbol{l}^*}$. At every iteration add one to the threshold of the lowest class i whose threshold is below M_i. Continue to iterate so long as the expected reward is improved.

The policy that results from Algorithm A has a similar structure to $\pi_{\boldsymbol{l}^*}$. However, it accepts more classes to the system. This is another RCS policy.

Algorithm B: Start with $\pi_{\boldsymbol{l}^*}$ that consists of all feasible threshold policies that differ from $\pi_{\boldsymbol{l}}$ by only one component which is at distance one from $\boldsymbol{l}$; and search for the best one in this neighborhood. Stop if no improvement is obtained.

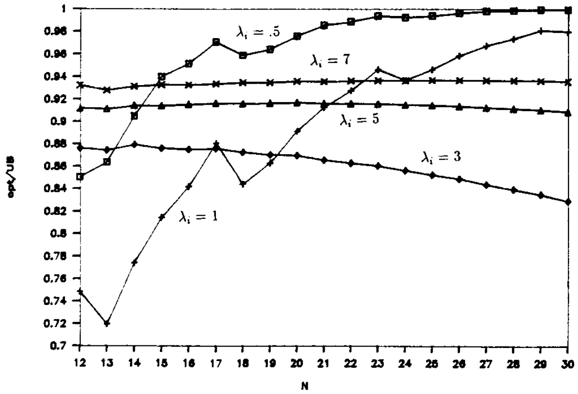

Fig. 1. Bound quality for $n = 3$, $\frac{r_i \cdot \mu_i}{b_i} = 1$ and various loads, as a function of N.

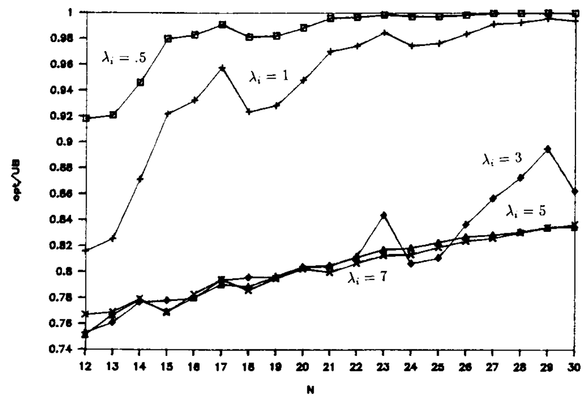

Fig. 2. Bound quality for $n = 3$, $\frac{r_1 \cdot \mu_1}{b_1} = 3$, $\frac{r_2 \cdot \mu_2}{b_2} = 2$, $\frac{r_3 \cdot \mu_3}{b_3} = 1$, and various loads, as a function of N (same arrival rates).

An alternative but extremely more time consuming approach is to search through all the threshold policies and select the best one. Another approach proposed by one of the referees is to exhaustively search within the set of all (t, l)—policies. Other search techniques could be also valuable.

Below, we present numerical results only of our algorithms.

IV. Numerical Examples

In this section we examine the quality of our bound, and evaluate the performance of the CS and the RCS policies with respect to the upper bound. We consider examples with different traffic classes for which the importance indexes $\frac{r_i \cdot \mu_i}{b_i}$, $1 \le i \le n$ are either homogeneous or heterogeneous. The quality of the bound is depicted by the ratio between the optimal reward and the upper bound, as a function of N—the number of bandwidth units. The quality of each policy is depicted by the ratio between its expected reward and the upper bound, as a function of N.

A. The Quality of the Upper Bound

In Figs. 1–2 we depict the ratio between the optimal expected reward and the upper bound as a function of N. The optimal reward is computed by the value iteration procedure [10] which stops when the error with respect to the optimal reward is less than 10^{-3}.

In Fig. 1 we consider three traffic classes ($n = 3$) with homogeneous importance indices. For example, $\frac{r_i \cdot \mu_i}{b_i} = 1$. The specific parameters that we use are $\boldsymbol{r} = (2, 3, 6)$, $\boldsymbol{b} = (2, 3, 6)$ $\mu = (1, 1, 1)$. We depict five cases where each one of them has $\lambda_1 = \lambda_2 = \lambda_3$. The cases are specified by $\lambda_i = (0.5, 1, 3, 5, 7)$, and the ratios are depicted for $N = 12, 13, \cdots, 30$. One may observe that for low loads ($\lambda_i = 0.5, 1$), the quality is basically increasing with N. For higher loads, the quality is almost insensitive to N. At any instant, the ratio is greater than 0.72, and for high loads ($\lambda_i = 5, 7$) the ratio is greater then 0.91 for every N. For lower loads, the ratio is above 0.9 only for large N's. One may also observe that in most cases, the bound is better for extreme loads (high or low) than for medium loads. Hence, one may conclude that in the homogeneous case and for high and low loads, the bound is quite close to the optimal reward, except for low loads and small N's.

In Fig. 2, we consider three traffic classes ($n = 3$) with heterogeneous importance indices. We present five cases with the following parameters held fixed. $\boldsymbol{r} = (2, 3, 6)$, $\boldsymbol{b} = (2, 3, 6)$ $\boldsymbol{\mu} = (3, 2, 1)$. We consider five different loads, where in each load all arrival rates are equal. The rates are given by $\boldsymbol{\lambda}_i = (0.5, 1, 3, 5, 7)$, and the ratios are depicted for $N = 12, 13, \cdots, 30$. Observe that the importance indexes assume the values $\frac{r_1 \cdot \mu_1}{b_1} = 3$, $\frac{r_2 \cdot \mu_2}{b_2} = 2$, $\frac{r_3 \cdot \mu_3}{b_3} = 1$.

A similar behavior to the homogeneous case in Fig. 1 is observed also for this heterogeneous case.

B. The Performance of the CS Policy

In Figs. 3 and 4 we depict the ratio between the expected reward under the CS policy and the upper bound as a function of N. In Fig. 3 we examine homogeneous importance indexes, and in Fig. 4 heterogeneous importance indexes. From many examples we considered we may draw two main conclusions. The first is that for equal importance indices, the CS performance is quite close to the optimal policy. This phenomenon has not been observed in [11]. The second one is that for heterogeneous importance indices, there is a large gap between the CS performance and the upper bound. This along with the improvement obtained by the RCS policy (as presented in the next subsection) implies that in the heterogeneous case the CS can significantly be improved.

In Fig. 3, we consider five cases that differ by the number of traffic classes, and depict the ratios as a function of N. The cases are specified by $n = 2, 3, 4, 5, 6$, and N varies from 12 to 60. All other parameters are given by $\boldsymbol{\lambda}_i = 5, i = 1, \cdots, 6$, $\boldsymbol{\mu}_i = 1, i = 1, \cdots, 6$, $\boldsymbol{b} = (1, 6, 2, 5, 3, 4)$ $\boldsymbol{r} = (1, 6, 2, 5, 3, 4)$. (The parameters for each case n are extracted from the first n elements in each vector.) Observe that $\frac{r_i \cdot \mu_i}{b_i} = 1$ for $i = 1, \cdots, 6$. One may observe that in the homogeneous case the CS is within 82% of the upper bound, and in many instances it is within 90% of the upper bound. However, it shows high sensitivity to small changes of N's.

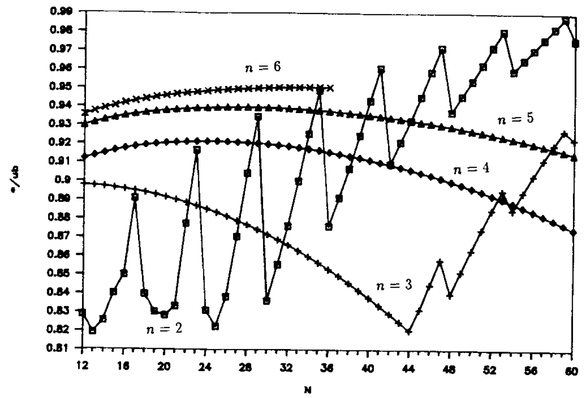

Fig. 3. CS quality for $n = 2, 3, 4, 5, 6$ as a function of N, when $\frac{r_i \cdot \mu_i}{b_i} = 1$.

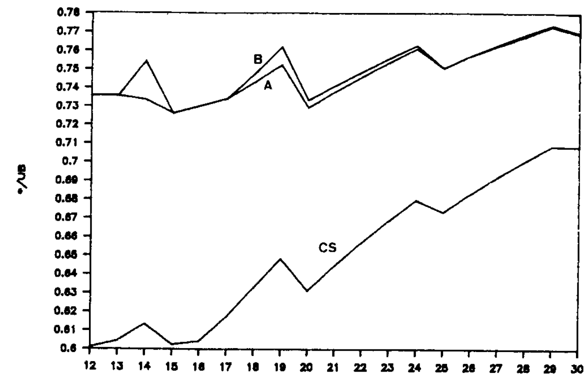

Fig. 5. RCS and CS qualities for $n = 4$ as a function of N.

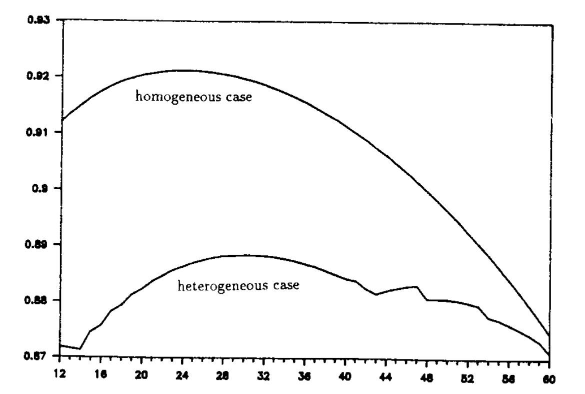

Fig. 4. CS quality for $n = 4$ as a function of N, for homogeneous and almost homogeneous indexes.

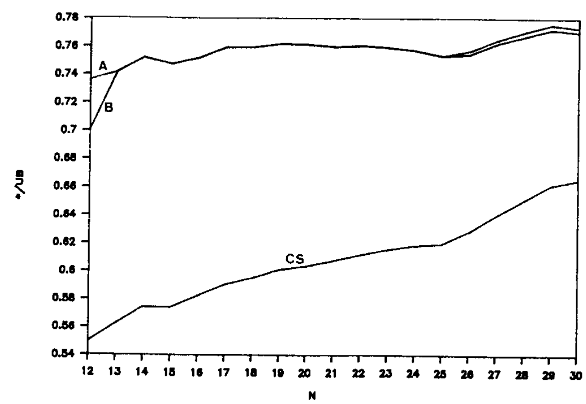

Fig. 6. RCS and CS qualities for $n = 5$ as a function of N.

In Fig. 4, we examine the sensitivity of the CS policy to small deviation from the homogeneous case. We consider two cases with four traffic classes. The arrival rates and the bandwidth requirements of all classes are the same and admit the equalities $\boldsymbol{\lambda}_i = 5,\ i = 1, \cdots, 4$ and $\boldsymbol{b} = (1, 6, 2, 5)$. In one case we consider $\boldsymbol{\mu}_i = 1,\ i = 1, \cdots, 4$ and $\boldsymbol{r} = (1, 6, 2, 5)$ which results in $\frac{r_i \cdot \mu_i}{b_i} = 1,\ i = 1, 2, 3, 4$. For the second case we consider $\mu = (1.1, 1, 1, 0.9)$ and $\boldsymbol{r} = (1, 5.9, 2.1, 5)$ which results in the following heterogeneous indices: $\frac{r_1 \cdot \mu_1}{b_1} = 1.1$, $\frac{r_2 \cdot \mu_2}{b_2} = 0.983$, $\frac{r_3 \cdot \mu_3}{b_3} = 1.05$ and $\frac{r_4 \cdot \mu_4}{b_4} = 0.9$.

One may observe that a slight deviation from homogeneous indices has a drastic affect on the CS behavior with respect to the bound. In this example, the ratio is reduced by 15% for small N's. The reduction becomes smaller as N increases. Although this behavior may be attributed to the upper bound, the improvement obtained by the RCS in the heterogeneous domain leads us to believe that it is the CS behavior.

Further observations on the CS policy are given in the next subsection.

C. The Quality of the RCS as Produced by Algorithms A and B

In this subsection we examine the quality of our proposed RCS policy and compare it to that of the CS policy. The final form of our RCS policy are derived by both of our improvement algorithms. One may observe below that the differences between them are minor. We consider only heterogeneous indices as the CS performs quite well for the homogeneous case.

In Figs. 5–7 we compare between the CS, RCS and the upper bound for $n = 4$, 5, and 6. In Fig. 8 we perform the same comparison for a case with highly different indexes. In all figures we depict three functions: The ratio between the expected reward under the CS policy and the upper bound as a function of N; the ratio between the expected reward under the RCS policy generated by Algorithm A and the upper bound as a function of N; and the ratio between the expected reward under the RCS policy generated by Algorithm B and the upper bound as a function of N.

In Fig. 5, we consider four traffic classes where $\lambda_i = 5,\ i = 1, \cdots, 4$, $\mu = (0.5, 1, 1.5, 2)$, $\boldsymbol{b} = (1, 6, 2, 5)$ and $\boldsymbol{r} = (2, 5, 2.5, 5)$. This results in $\frac{r_1 \cdot \mu_1}{b_1} = 1$, $\frac{r_2 \cdot \mu_2}{b_2} = 0.833$, $\frac{r_3 \cdot \mu_3}{b_3} = 1.875$ and $\frac{r_4 \cdot \mu_4}{b_4} = 2$.

One may observe that the quality of all policies are improved with N. Also, Algorithms A and B produce policies whose expected rewards are almost identical. The quality of the RCS policies vary from 0.73 to 0.77, while that of the

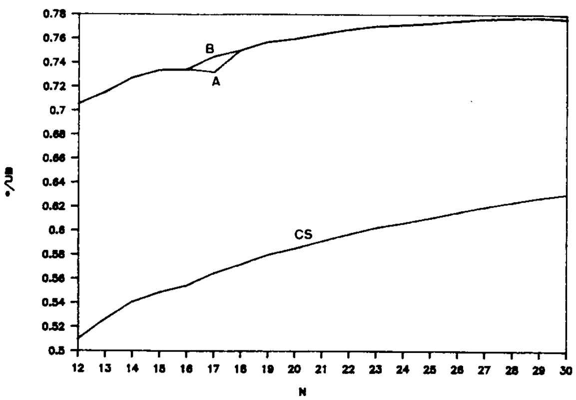

Fig. 7. RCS and CS qualities for $n = 6$ as a function of N.

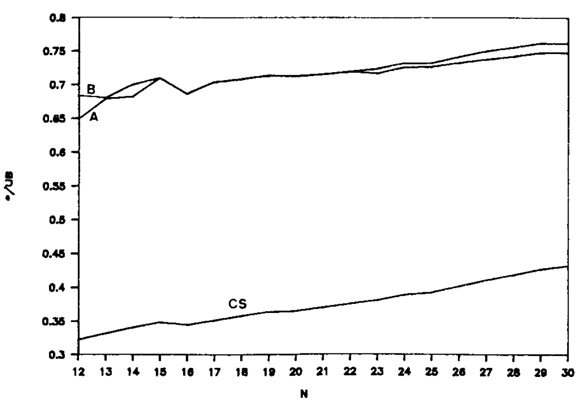

Fig. 8. RCS and CS qualities for $n = 6$ as a function of N.

CS policy varies from 0.6 to 0.71. Hence, we obtain an improvement of 8–21%.

Similar behavior is demonstrated in Fig. 6 for the case of $n = 5$ where $\boldsymbol{\lambda}_i = 5$, $i = 1, \cdots, 5$, $\boldsymbol{\mu} = (0.5, 1, 1.5, 2, 2.5)$, $\boldsymbol{b} = (1, 6, 2, 5, 3)$ $\boldsymbol{r} = (2, 5, 2.5, 5, 3.5)$. This results in $\frac{r_1 \cdot \mu_1}{b_1} = 1$, $\frac{r_2 \cdot \mu_2}{b_2} = .833$, $\frac{r_3 \cdot \mu_3}{b_3} = 1.875$, $\frac{r_4 \cdot \mu_4}{b_4} = 2$ and $\frac{r_5 \cdot \mu_5}{b_5} = 2.916$.

In this case the quality of the RCS policies vary from 0.70 to 0.77, while that of the CS policy varies from 0.55 to 0.66. Hence, we obtain an improvement of 16–27%. Notice that for $n = 5$ the graphs are more smooth than for $n = 4$.

In Fig. 7, we consider the case of $n = 6$ where $\boldsymbol{\lambda}_i = 5$, $i = 1, \cdots, 6$, $\boldsymbol{\mu} = (0.5, 1, 1.5, 2, 2.5, 3)$, $\boldsymbol{b} = (1, 6, 2, 5, 3, 4)$ $\boldsymbol{r} = (2, 5, 2.5, 5, 3.5, 3.5)$. This results in $\frac{r_1 \cdot \mu_1}{b_1} = 1$, $\frac{r_2 \cdot \mu_2}{b_2} = 0.833$, $\frac{r_3 \cdot \mu_3}{b_3} = 1.875$, $\frac{r_4 \cdot \mu_4}{b_4} = 2$, $\frac{r_5 \cdot \mu_5}{b_5} = 2.916$ and $\frac{r_6 \cdot \mu_6}{b_6} = 2.625$.

In this case the quality of the RCS policies vary from from 0.71 to 0.78, while that of the CS policy varies from 0.51 to 0.63. Hence, we obtain an improvement of 24–39%. Notice that the RCS policy becomes better with respect to the CS policy as the number of traffic classes increases.

Finally, in Fig. 8 we consider the case of $n = 6$, with the following parameters which produce highly heterogeneous indexes: $\boldsymbol{\lambda}_i = 5$, $i = 1, \cdots, 6$, $\boldsymbol{\mu} = (0.5, 1, 2, 2, 0.5, 3)$, $\boldsymbol{b} = (1, 6, 2, 5, 3, 4)$ $\boldsymbol{r} = (3, 4, 3, 7, 3, 8)$. This results in $\frac{r_1 \cdot \mu_1}{b_1} = 1.5$, $\frac{r_2 \cdot \mu_2}{b_2} = 0.667$, $\frac{r_3 \cdot \mu_3}{b_3} = 3$, $\frac{r_4 \cdot \mu_4}{b_4} = 2.8$, $\frac{r_5 \cdot \mu_5}{b_5} = 0.5$ and $\frac{r_6 \cdot \mu_6}{b_6} = 6$.

In this case the quality of the RCS policies vary from 0.65 to 0.75, while that of the CS policy varies from 0.33 to 0.43. Hence, we obtain an improvement of 74–97%.

REFERENCES

[1] E. V. Denardo, *Dynamic Programming.* Englewood Cliffs, NJ: Prentice Hall, 1982.

[2] A. Gavious, "A stochastic knapsack problem in an ISDN environment: A bound to the optimal reward and an efficient policy," M.S. thesis, Dep. Comput. Sci., Technion, Haifa 32000, Israel, Oct. 1990.

[3] A. Gavious and Z. Rosberg, "A restricted complete sharing policy for a stochastic knapsack problem in B-ISDN, IBM Israel Sci. Technol., Technion City, Haifa 32000, Tech. Rep. 88.305, Mar. 1991.

[4] A. Hordijk and C. M. Kallenberg, "Constrained undiscounted stochastic dynamic programming," *Math. Oper. Res.*, vol. 9, pp. 276–289, 1984.

[5] F. P. Kelly, *Reversibility and Stochastic Networks.* New York: Wiley, 1979.

[6] L. Kleinrock, *Queueing Systems, Volume 1: Theory.* New York: Wiley, 1975.

[7] S. A. Lippman, "Applying a new device in the optimization of exponential queueing systems," *Management Sci.*, vol. 23, pp. 687–710, 1975.

[8] Z. Rosberg and P. Kermani, "Customer routing to different servers with complete information," *Adv. Appl. Prob.*, vol. 21, pp. 861–882, 1989.

[9] ——, "Customer scheduling under queueing constraints," *IEEE Trans. Automat. Contr.*, Dec. 1991.

[10] S. Ross, *Introduction to Stochastic Dynamic Programming.* New York: Academic, 1983.

[11] K. W. Ross and D. Tsang, "The stochastic knapsack problem," *IEEE Trans. Commun.*, vol. 37, pp. 740–747, 1989.

[12] ——, "Optimal circuit access policies in an ISDN environment: A Markov decision approach," *IEEE Trans. Commun.*, vol. 37, pp. 934–939, 1989.

[13] K. W. Ross and D. D. Yao, "Monotonicity properties for the stochastic knapsack," *IEEE Trans. Inform. Theory*, vol. 36, pp. 1173–1179, 1990.

[14] D. Tsang and K. W. Ross, "Algorithms for calculating exact blocking probabilities in multirate networks," *IEEE Trans. Commun.*, vol. 38, pp. 1266–1271, 1990.

[15] D. D. Yao and Z. Schechner, "Decentralized control of service rates in a closed Jackson network," *IEEE Trans. Automat. Contr.*, vol. 34, pp. 236–240, 1989.