A Jointly Optimum Scheduling and Memory Management for Matching Based Service

Saswati Sarkar*

Abstract

We consider resource allocation in a system which must process and subsequently deliver jobs from N input terminals to N output terminals as per matching constraints. Jobs are stored in a common memory before processing and transfer. The resource allocation objective is to minimize the job loss due to memory overflow. We present optimal scheduling and memory management policies for attaining this objectives for $N = 2$ and symmetric traffic. We identify certain characteristics of the optimal strategy for $N = 2$ and asymmetric traffic, and present a near optimal heuristic for this case. We obtain a heavy traffic lower bound for the optimal discounted cost function for the general case of arbitrary N and arbitrary traffic. We use this lower bound to propose a heuristic strategy whose performance is close to the optimal strategy in this case. The policies proposed in this paper substantially outperform the existing strategy for the system, which was designed and proved to be optimal under the assumption of infinite storage.

1 Introduction

We consider a jointly optimum scheduling and memory management problem for delivery of jobs from N input terminals to N output terminals. The service constraint is that a job requires a pre-determined input-output pair for service and an input or an output can be used for serving at most one job at a time. Multiple input-output pairs may be utilized at the same time as long as they do not share a common node. These constraints are commonly referred to as "matching constraints." Jobs are stored in a common memory before service. The resource management objective is to design the scheduling and the memory management so as to minimize job loss due to memory overflow. This resource allocation problem arises in several communication and computer systems. An example application is a shared memory processor architecture with finite storage. The architecture consists of N processors which need to process and subsequently transfer jobs to N output terminals. Each processor can perform only one type of task. Thus each job must be served by a pre-assigned processor depending on its nature and subsequently transferred to a pre-assigned output terminal. All jobs are stored in a common memory. The transfer of jobs from input to output must follow matching constraints, i.e., one input can transfer only one job at a time, and one output can receive only one job at time. Multiple jobs can be transferred as long as they do not share a common input or output. This transfer constraint arises when the input and output terminals are connected by a crossbar type switching fabric. Refer to Figure 1 for an illustration.

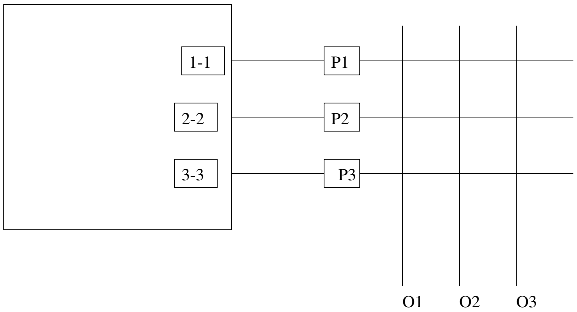

Figure 1: The figure shows a system with 3 input processor (P1, P2, P3) and 3 outputs (O1, O2, O3) connected by a crossbar. A packet $i - j$ is waiting for processing at input i and subsequent transfer to input j. Here, each input queue has only one job waiting for transfer to the output, $x_{11} = x_{22} = x_{33} = 1, x_{ij} = 0$, if $i \neq j$. The input-output pairs $1 - 1, 2 - 2, 3 - 3$ constitute a matching. Similarly, $1 - 2, 2 - 1, 3 - 3$ is another example matching. Either matching can be scheduled, but the first completes 3 jobs while the second completes only one job as the $1 - 2$ and $2 - 1$ queues are empty ($x_{12} = x_{21} = 0$).

The resource management problem has two components here: (a)job scheduling and (b)memory management. The first decides which jobs can be transferred to the output without violating the matching constraints. Memory management determines the job acceptance policy, namely which incoming job will be accepted, and which job should be dropped in case of memory overflow (the choice is between incoming jobs and existing jobs). These decisions will depend on buffer occupancy and possibly arrival and service statistics. More importantly, the job scheduling and memory management decisions must be taken in conjunction, and will depend on each other. For example, only one job can be transferred to the output side if all outstanding tasks are waiting at the same input or are destined to the same output. However, many more jobs can be transferred if the outstanding tasks do not have common inputs and outputs. Refer to figure 2 for an illustration. Memory man-

*S. Sarkar is with the Electrical and Systems Engineering Department in the University of Pennsylvania. Email: swati@ee.upenn.edu

agement can be utilized efficiently to ensure that fewer outstanding jobs share input and output terminals. We discuss this in details later.

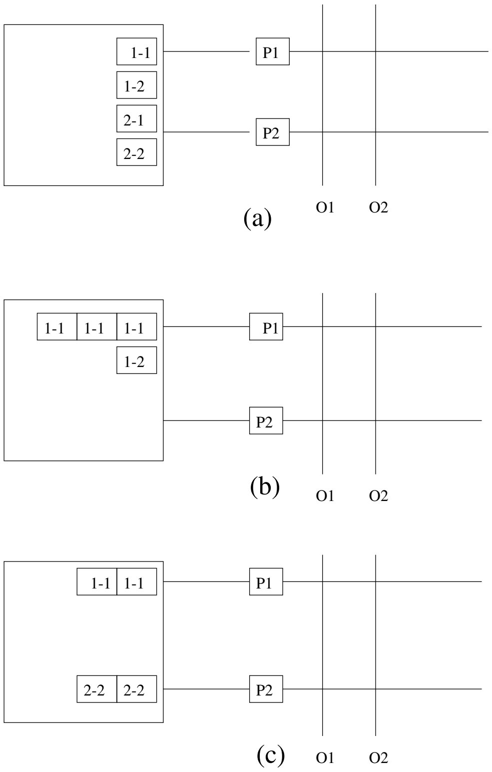

Figure 2: The figure shows a system with $N = 2$ with three different buffer occupancies. The buffer occupancies are $\vec{x} = (1, 1, 1, 1)$ (Figure (a)), $\vec{x} = (3, 1, 0, 0)$ (Figure (b)) and $\vec{x} = (2, 0, 0, 2)$ (Figure (c)).

We first briefly review the existing literature in optimum scheduling and memory management. Tassiulas *et al.*[11] presented a maximum difference backlog based scheduling which attains the maximum possible throughput in any network of constrained queues which includes the resource allocation problem we consider here. A similar result was obtained by Mckeown *et al.* in [6]. However, the important point to note is that *both these generic results hold for infinite storage assumption only*. The authors do not consider any memory constraints there, and as such have not addressed memory management at all, while focusing on scheduling only. From the results in [6, 11], job scheduling as per a maximum weighted matching attains the maximum possible throughput in shared memory processors if the storage is infinite. We refer to this scheduling as MM scheduling in the rest of this paper. Interestingly, we observe that MM scheduling is no longer the optimum scheduling for the finite buffer case which arises in practice. Thus memory constraints introduce the memory management problem and also necessitates a change in the scheduling policy.

Cassandras *et al.*[10, 9, 12] have considered several routing and scheduling problems in context of N queues sharing a single server. These queues have separate storage. So memory is not a shared resource. The scheduling problem is a N to 1 scheduling while we need a N to N scheduling. Memory management has been addressed for other types of architectures which do not consider any scheduling. For example, Guerin *et al.*[2] and Kamoun *et al.*[3] present optimal memory management schemes in shared memory switches, where multiple jobs can be simultaneously transferred to an output from different inputs. The outputs can simultaneously serve jobs independent of each other. Thus, scheduling is not an issue in these switches and memory is the only shared resource. Understandably, the authors focus on memory management only. Summarizing, the problem of deciding a jointly optimal scheduling and memory management strategy does not arise, and the memory management strategies need not be designed to cater to scheduling dependencies in these cases.

Now we outline the contribution of this paper. We present a markov decision problem based mathematical formulation of the resource allocation problem in section 2. We present a closed form computationally simple optimal scheduling and memory management strategy for the special case of $N = 2$ (2 input processors and 2 output terminals) under the assumption of equal arrival rates for all input-output streams in section 3.1. The optimal scheduling transfers as many jobs as possible at all times. The optimal memory management uses an acceptance policy which balances the number of outstanding jobs of different input-output streams. This in turn allows the scheduling to transfer several jobs to the output and reduces the buffer occupancy and the memory overflow. We obtain several features of the optimal strategy for $N = 2$ and arbitrary arrival rates in section 3.2, and using these properties we design a near optimal heuristic strategy, "pair scheduling minimum difference" (PSMD) in this case. The numerical results using markov decision process based computation will also demonstrate that the MM scheduling which is known to be optimal in absence of memory constraint[6, 11] is strictly suboptimal in this case. In fact, PSMD reduces the job loss by 30% as compared to MM in many cases. We present a heavy traffic lower bound for the optimal discounted cost function for the general case of architectures with N input processors and N output terminals in section 4. This lower bound and certain properties of the optimal strategy derived from the general markov decision process framework help us design a heuristic, "minimum congestion policy" (MC) in this case. We present numerical performance evaluation to demonstrate that MC performs substantially better than MM scheduling of [6, 11], and in many cases reduces the job loss by 30% or more.

2 Network Model

In an abstract setting, we consider the problem of jointly optimizing the scheduling and memory management problems where the scheduling must satisfy matching constraints. There are N input terminals and N output terminals. A job must receive service from a pre-determined input output pair i, j. The service can be scheduled under the following matching constraints. An input-output pair can be involved in only one service at a time. Thus the service must constitute a matching* at any time t. Jobs are stored in a common memory of size B until they are transferred to the intended outputs.

We present our assumptions about the arrival and the service processes. Arrival processes are independent Poisson. The arrival rate of jobs for processor i and output j is λ_{ij}, $i = 1, \dots, N$, $j = 1, \dots, N$. Queue length of jobs waiting for input i and output j at time t is $x_{ij}(t)$. Clearly $\sum_{i=1}^{N} \sum_{j=1}^{N} x_{ij}(t) \leq B$. Services have exponential duration with μ_{ij} being the service rate for the jobs for the ith input and jth output. For simplicity we will assume equal service rates, $\mu_{ij} = \mu$ for all i, j. The assumption can be justified from the observation that system performance is a function of the arrival rates, service rates and the buffer size, and unequal service rates have the same effect on the performance as unequal arrival rates. We will study the effect of unequal arrival rates. In general, the performance depends on the ratio between the arrival and service rates rather than their absolute values, and as such without loss of generality we assume that $\mu + \sum_{i=1}^{N} \sum_{j=1}^{N} \lambda_{ij} = 1$, and $\mu > 0$. A preemptive service model is normally assumed for scheduling problems, e.g., [10]. More specifically, the standard assumption is that after every event (service completion and new arrival), all ongoing processing and transfer are interrupted and a new scheduling decision is taken. We make the same assumption here. In addition, for analytical convenience we assume that all jobs which start service at the same time end service at the same time as well. Thus a service completion does not pre-empt any ongoing service as all ongoing services finish at the same time. However, arrivals still preempt the service. This additional assumption simplifies the mathematical framework without affecting the results substantially.

The scheduling problem is to decide which jobs are served at any time t. There can be several possible matchings and the scheduling problem is to choose the appropriate matching. Refer to Figure 1 for examples of valid scheduling. Every time a job transfer is completed from input i to output j, the queue length $x_{ij}(t)$ decreases by one.

The memory management problem is to decide how the input buffer is shared by jobs intended for different input-output pairs. When a new job arrives at an input, one of the following actions can be taken: (a)the job is accepted (b)the job is rejected and (c) the job is accepted while some other job waiting at the input is dropped. The last action is commonly referred to as "push-out." If a job arrives when the buffer is full, then one of the last two actions must be taken. The memory management problem is to choose the appropriate course of action when a new job arrives. Every time a new job is accepted in input i for output j, the queue length $x_{ij}(t)$ increases by 1. If a job waiting at input i intended for output j is dropped, the queue length $x_{ij}(t)$ decreases by one.

The objective is to choose the scheduling and the memory management scheme so as to minimize the average job loss. A job loss happens every time an incoming job is rejected or an existing job is dropped. Throughput of the switch is the average number of jobs transferred from the inputs to the outputs. Throughput maximization and loss minimization are equivalent objectives, as the sum of throughput and loss rates is equal to the sum of the arrival rates. Thus a strategy which minimizes loss maximizes throughput and vice versa.

The optimal scheduling and memory management policy is the one which minimizes the average job loss. We present a generic markov decision process(MDP)[7, 8] based technique for computing the optimal loss rates and the optimal scheduling and memory management decisions. We define some notations next.

Let $S(\vec{x})$ be the set of possible next states when the current state is $\vec{x}$ and the next event is completion of job transmission. Let $\mathcal{M} = \{\vec{M}_1, \vec{M}_2, \dots \vec{M}_P\}$ be the collection of all matchings. Now, $S(\vec{x}) = \{\vec{y} : \vec{y} = \vec{x} - \vec{M}, \vec{M} \in \mathcal{M}, y_{ij} \geq 0 \forall i, j\}$. Let $A(\vec{x}, i, j)$ be the set of possible next states when the current state is $\vec{x}$ and the next event is arrival of a job at the ith input for the jth output. The new job can be accepted, rejected or accepted while dropping some other job at the same input. Let $\vec{e}_{ij}$ be a vector with a 1 in the $i-j$th position and a 0 elsewhere. Thus $\vec{y} \in A(\vec{x}, i, j)$ if either (a)$\vec{y} = \vec{x}$ or (b)$\vec{y} = \vec{x} + \vec{e}_{ij}$ or (c)$\vec{y} = \vec{x} + \vec{e}_{ij} - \vec{e}_{kl}$. If input buffer is full, (i.e., $\sum_{k=1}^{N} \sum_{l=1}^{N} x_{kl} = B$), case (b) is ruled out as a new job can be accepted only by pushing out an old job.

The job loss experienced when the current state is $\vec{x}$, the ith input has an arrival for output j and the next state is $\vec{y}$ is denoted by the function $c(\vec{x}, \vec{y}, i, j)$, where

$$c(\vec{x}, \vec{y}, i, j) = 1 + \sum_{i=1}^{N} \sum_{j=1}^{N} (x_{ij} - y_{ij}) \tag{1}$$

We assume that every job loss is associated with a cost of one unit. Let the optimal discounted cost associated with a state $\vec{x}$ be $J_{\beta,\mathrm{OPT}}(\vec{x})$, where β is the discount factor. This cost is a measure of the minimum future loss associated with a state. Then $J_{\beta,\mathrm{OPT}}(\vec{x})$ is the unique solution of the following equations[7]

$$J_{\beta,\mathrm{OPT}}(\vec{x}) = T J_{\beta,\mathrm{OPT}}(\vec{x}) \tag{2}$$

where T is a linear operator on any function $f(\vec{x})$ defined

*In graph-theoretic notions, a matching is a collection of edges which do not share a node. In the switching context a matching is a collection of input output pairs i, j which do not have a common input or output.

as follows:

$$
\begin{aligned}
Tf(\vec{x}) &= \sum_{i=1}^{N}\sum_{j=1}^{N} \min_{\vec{y}\in A(\vec{x},i,j)} \lambda_{ij}\left(c(\vec{x},\vec{y},i,j)\right. \\
&\quad \left.+\beta f(\vec{y})\right) + \beta\mu \min_{\vec{y}\in S(\vec{x})} f(\vec{y})) \qquad (3)
\end{aligned}
$$

The optimal average job loss is given by $\lim_{\beta\to 1}(1-\beta)J_{\beta,\text{OPT}}(\vec{0})$[7, 8].

Consider a scheduling strategy $s_\beta(\vec{x})$ and a memory management strategy for arrival in the ith input, jth output $a_\beta(\vec{x},i,j)$ which choose the next states which have the minimum possible discounted cost:

$$
\begin{aligned}
s_\beta(\vec{x}) &= \arg\min_{\vec{y}\in S(\vec{x})} J_{\beta,\text{OPT}}(\vec{y})) \qquad (4) \\
a_\beta(\vec{x},i,j) &= \arg\min_{\vec{y}\in A(\vec{x},i,j)} \left(c(\vec{x},\vec{y},i,j)\right. \\
&\quad \left.+\beta J_{\beta,\text{OPT}}(\vec{y})\right) \qquad (5)
\end{aligned}
$$

The strategy which minimizes average job loss is a limiting case ($\beta \to 1$) of equations (4) and (5)([8] Chapter V). **The optimal scheduling, $s_{\textbf{OPT}}(\vec{x})$ and memory management policy for arrival in the $i-j$th queue, $a_{\textbf{OPT}}(\vec{x},i,j)$ is given by**

$$
\begin{aligned}
s_{\text{OPT}}(\vec{x}) &= \lim_{\beta_n\to 1} s_{\beta_n}(\vec{x}) \qquad (6) \\
a_{\text{OPT}}(\vec{x},i,j) &= \lim_{\beta_n\to 1} a_{\beta_n}(\vec{x},i,j) \qquad (7)
\end{aligned}
$$

for a certain infinite sequence $\beta_1,\beta_2\ldots$ such that $\beta_i < 1$ for all i and $\lim_{n\to\infty}\beta_n = 1$.

Now we present value iteration based standard techniques for computing the solutions to the linear equations (2)[7] which will yield the optimal discounted loss rates associated with the states and thereafter the optimal strategy via equations (4) to (7). **The sequence of iterations**

$$
J_{\beta,k+1}(\vec{x}) = TJ_{\beta,k}(\vec{x}), k = 0,1,2,\ldots \qquad (8)
$$

converge to the optimal discounted costs $J_{\beta,\textbf{OPT}}(\vec{x})$ for all states $\vec{x}$ irrespective of the initial choice of the $J_{\beta,0}(\vec{x})$ function. Thus $\lim_{k\to\infty} J_{\beta,k}(\vec{x}) = J_{\beta,\textbf{OPT}}(\vec{x})$ for all $\vec{x}$[7].

Value iteration based computations become intensive with increase in input buffer size. This happens because the computation needs several iterations and each iteration has a complexity $\Omega\left((N^2+1)^B\right)$.† For example, for $N = 2, B = 50$ each iteration involves computations with 5^{50} variables. The bottomline is that the generic computation technique does not scale with the buffer size. However, we will use the general results obtained from this MDP framework to compute closed form optimal strategies for the special case of $N = 2$ where all arrival streams have equal arrival rates ($\lambda_{ij} = \lambda$ for all i,j). Subsequently we will identify certain key characteristics of the optimal strategy for $N = 2$ and arbitrary arrival rates, and use these properties to design a near optimal heuristic using these properties. Finally, we will prove a heavy traffic lower bound for the optimal discounted cost-to-go functions for arbitrary arrival rates and arbitrary N. The lower bound will help us design a heuristic which performs attains low job loss in this case.

†A computation is said to have $\Omega(n)$ complexity for input size n if it requires at least cn steps for some constant c and all large n.

Finally, we show how this resource allocation problem applies to the shared memory processor architecture. Consider a system with N processors at the input side and N terminals at the output side. Each processor can perform a specific task. A job needs service from a pre-determined processor, and after service completion the job must be switched to a pre-decided output. The jobs are stored in a shared memory before processing and transfer to the outputs. Since each memory access is a time consuming operation, a processor must transfer a job to the output immediately after processing, i.e., the processor can not store it in the memory once again and transfer it at a later time. This is further motivated by the fact that often times depending on the connection speed between the input and the output, the transfer time is much larger than the processing time. Thus a separate memory access is not worth for processing alone. Thus we consider the processing and transfer as one combined operation, or as "service" in the terminology of the generic resource allocation setting. While input i is transferring a job to output j, input i can not transfer any other job and output j can not receive any other job. This happens because of the physical nature of the connections between inputs and outputs (e.g., crossbar type). Since transfer and processing are considered as one operation here, the jobs receiving service at any time t must involve distinct input-output pairs, i.e., service must constitute a matching. In practice the service process will be non-preemptive, as a preemptive service calls for additional memory access each time the service is pre-empted. However, in general it has been observed that a preemptive assumption does not alter the nature of the optimal strategies, and in addition this assumption simplifies the mathematical framework and the analysis substantially.

3 A loss optimal scheduling and memory management for $N = 2$

In this section we will consider the scheduling and memory management strategies for the special case of $N = 2$. We will first present the intuition behind the optimal resource allocation problem in this case. Subsequently we will present the closed form optimal strategy for symmetric traffic in subsection 3.1. Extension to the case of asymmetric traffic is considered in subsection 3.2. There we obtain certain key properties of the optimal strategy for the asymmetric traffic case, and then using these properties we design a heuristic. In subsection 3.3 we will show that this heuristic attains near optimal job loss using numerical computations.

We first describe the optimal resource allocation problem in this case. Refer to Figure 2 for illustration. There are four different queues of jobs, x_{11}, x_{12}, x_{21} and x_{22} sharing the common memory of size B. The queue x_{ij} stores jobs which must be served by input i and thereafter transferred to output j. The jobs in the pairs x_{11}, x_{22} and x_{12}, x_{21} can be served simultaneously as the queues in these pairs do not share an input or output terminal. Also, any single queue can receive service at a time, but no other combination is allowed. Job loss can be minimized if the buffer occupancy is reduced. Intuitively this happens if a pair of queues is served together rather than singletons because the former removes two jobs from the buffer while the latter removes only one job from the buffer. However, if the buffer occupancy is such that one queue is empty in each pair then only singletons can be scheduled. For example, if $x_{11} = x_{21} = 0$ then either x_{22} or x_{12} can be served (Figure 2(b)). For further illustration consider the buffer occupancies in Figures 2(a) ($\vec{x} = (1,1,1,1)$) and 2(b)($\vec{x} = (3,1,0,0)$). Observe that the total number of jobs waiting for service is the same for both configurations(4 jobs). Only one job can be served in Figure 2(b) (from x_{11} or x_{12}) as all jobs must be served by the same input (input 1). However, two jobs can be served in Figure 2(a) (can schedule either the pair x_{11}, x_{22} or the pair x_{12}, x_{21} as all the queues are nonempty).

Thus a larger number of jobs can be served and hence job loss can be reduced when the load is balanced across the queues. However, load should be balanced across "appropriate" queues. For example, consider the buffer occupancies in Figures 2(b) ($\vec{x} = (3,1,0,0)$) and 2(c)($\vec{x} = (2,0,0,2)$). Only two queues have jobs in both cases and the total number of jobs waiting for transmission (4 jobs) are the same in both. However, only one job can be served in Figure 2(b) as discussed before, whereas two jobs can be served in Figure 2(c) (from queues x_{11} and x_{22}) since there are jobs waiting for different inputs and different outputs. The configuration in Figure 2(c) allows a service of a larger number of jobs because it balances the load between appropriate queues. This can be quantified by considering the difference between the queue lengths of the queues which can be scheduled together, i.e., $|x_{11} - x_{22}|$ and $|x_{12} - x_{21}|$. Note that these differences are $0, 0$ for Figures 2(a) and 2(c) and $3, 1$ for Figure 2(b). In general, a larger number of jobs can be served if these differences are smaller.

Appropriate push-out policy can be used to reduce these differences. When a new job arrives and the buffer is full, then the job can be rejected or accepted by pushing out an existing job. The appropriate decision can be taken with a view to reducing this difference. Assume that $B = 4$ in Figure 2. Let a job arrive for input 2, for output 1. In Figure 2(b) the differences between the queue lengths of the queues in the pairs remain $3, 1$ if the new job is rejected, while the differences become $2, 0$ if the new job is accepted while pushing out an existing job waiting for input 1, output 1 (in the x_{11} queue). After this replacement, both queues x_{12} and x_{21} can be scheduled serving two jobs simultaneously. In Figure 2(a) the new job should be rejected as its acceptance using push-out will adversely affect the balance and increase the differences (job rejection leaves the differences at $0, 0$ and push out increases the differences to $1, 1$). Summarizing, *a pair of queues should be scheduled whenever possible, and job acceptance/rejection/push-out should be used judiciously to balance the load across the appropriate queues which facilitates a simultaneous service of two jobs whenever possible.* These observations are the key behind designing optimal strategies for symmetric traffic and near-optimal heuristics for asymmetric traffic.

3.1 Equal arrival rates

In this subsection we will consider the symmetric traffic case. We assume that all arrival rates are equal, i.e., $\lambda_{ij} = \lambda$, for all i, j. We present the scheduling and memory management strategies which minimize the average job loss in this case. We will refer to the optimal policy as "PSMD" (**pair scheduling minimum difference acceptance**).

PSMD Scheduling (pair scheduling): The optimal scheduling is to serve a pair of queues whenever possible. Let the current state be $\vec{x}$ (let $\vec{x} \neq \vec{0}$). If all the queues are nonempty ($x_{ij} > 0$ for all i, j), schedule either the $1-1, 2-2$ pair (x_{11}, x_{22}) or the $1-2, 2-1$ pair (x_{12}, x_{21}). The choice between the pairs can be arbitrary and do not affect performance. Now consider the case when some queues are empty. If $\min(x_{11}, x_{22}) > \min(x_{12}, x_{21}) = 0$ schedule the $1-1, 2-2$ pair. If $\min(x_{12}, x_{21}) > \min(x_{11}, x_{22}) = 0$ schedule the $1-2, 2-1$ pair. If $\min(x_{12}, x_{21}) = \min(x_{11}, x_{22}) = 0$, only one queue can be scheduled, and the longest queue is selected. If all the queues are empty ($\vec{x} = \vec{0}$), no queue is scheduled for service.

PSMD Memory Management (minimum difference memory management): The memory management strategy decides whether to accept an incoming job, and if the decision is to accept the job then whether to push out an existing job. The optimal decision is to accept an incoming job without any push out as long as there is available storage. If the buffer is full when a new job arrives, then the job is either rejected or accepted while dropping an existing job from a different queue. The choice between the two is made with the objective of reducing the difference between the queue lengths of the pairs. We introduce some notations for describing this part of the memory management more formally. A "co-operating set" is a pair of queues which can be served together. Co-operating set 1 consists of the pair $1-1$ and $2-2$ and co-operating set 2 consists of the pair $1-2$ and $2-1$. We quantify the differences between the queues in the same co-operating set: $\text{diff}_1(\vec{x}) = |x_{11} - x_{22}|$, $\text{diff}_2(\vec{x}) = |x_{12} - x_{21}|$. Let $\vec{e}_{ij}$ be a vector with a 1 in the $i-j$th position and a 0 elsewhere. Let a job arrive for input i, output j (in queue $i-j$) and let the storage be full ($\sum_{i=1}^{N} \sum_{j=1}^{N} x_{ij} = B$). Then the system considers the state

$\vec{y} = \vec{x} + \vec{e}_{ij}$. If $\text{diff}_1(\vec{y}) = \text{diff}_2(\vec{y}) = 0$ new arrival is rejected. Otherwise, a job is dropped so as to reduce the larger of the two differences under buffer occupancy $\vec{y}$. If the two differences are equal, a job is dropped so as to reduce any one of the differences. If queue $i - j$ is selected for dropping a job, then the new arrival is rejected. Intuitively, the process is as though the new job is accepted and then a job is dropped so as to reduce the larger "diff" function in the resulting state. An illustrative example follows.

Example 3.1: We illustrate PSMD using the configurations in Figure 2. Let $B = 4$. In Figure 2(a) ($\vec{x} = (1, 1, 1, 1)$), PSMD schedules either the pair $1 - 1, 2 - 2$ or the pair $1 - 2, 2 - 1$ choosing between them arbitrarily. Any incoming job is rejected. In Figure 2(b) ($\vec{x} = (3, 1, 0, 0)$), only one queue can be scheduled. PSMD schedules the longer queue, queue $1 - 1$. An incoming job for input 1, output 1 is rejected. Any other incoming job is accepted while dropping a job from the $1 - 1$ queue. In Figure 2(c) ($\vec{x} = (2, 0, 0, 2)$), PSMD schedules the pair $1 - 1, 2 - 2$. Any incoming job is rejected. Now let $B = 5$. The scheduling remains the same as before in each case. The memory management decision is to accept any incoming job in all three cases.

The following theorem outlines the optimality of PSMD.

Theorem 1 *PSMD minimizes the average job loss for systems with* 2 *inputs and* 2 *output terminals* ($N = 2$) *with equal arrival rates in all streams* ($\lambda_{ij} = \lambda$ *for all* i, j).

The proof for Theorem 1 uses results from the general MDP based framework presented in section 2 and has been presented in appendix A.

3.2 Unequal arrival rates

In this section we will consider the general case with unequal arrival rates for different arrival streams, i.e., we no longer assume that $\lambda_{ij} = \lambda$. First, we identify certain properties of the optimal scheduling and memory management strategy, and subsequently present a heuristic using these properties.

Optimal Scheduling: The optimal scheduling schedules a pair of queues whenever possible.
Optimal Memory Management: The optimal decision is to accept an incoming job without any push out as long as the input buffer has space.

We prove these properties in appendix B using results from the general MDP based framework presented in section 2. These properties specify the strategy in certain cases. For instance, in Figure 2(c) the optimal scheduling is to serve the pair $1 - 1, 2 - 2$. Also, if $B = 5$ then the optimal memory management strategy accepts all incoming jobs for all three configurations in Figure 2. However, these properties do not completely specify the optimal strategy. For example, it is not known how to choose between pairs of queues when either pair can be scheduled (e.g., Figure 2(a)), or how to choose a queue when the system state is such that only one queue can be scheduled, i.e., when $\min(x_{12}, x_{21}) = \min(x_{11}, x_{22}) = 0$(e.g., Figure 2(b)). The optimal job acceptance/rejection/push-out decision when the input buffer is full is also not known in the asymmetric case (e.g., all three cases in Figure 2 with $B = 4$).

We propose PSMD (Section 3.1) as a heuristic here. We justify the choice as follows. PSMD satisfies the properties obtained for the optimal strategy in this general case. Besides, the scheduling and the memory management decisions of PSMD strive to balance the traffic across different queues which allow the service of larger number of jobs and thus reduce the buffer occupancy and the job loss. Numerical computations presented in the next subsection will demonstrate that PSMD attains near minimum job loss in this case.

3.3 Performance Evaluation using Numerical Computation

First we evaluate the performance of PSMD for unequal arrival rates, using the job loss experienced by the optimal strategy as a benchmark. We compute the average job losses of PSMD and the optimal strategy using MDP based techniques. We have observed that PSMD attains near minimum average job loss. We present some numerical results next for two different traffic patterns.

When all arrival rates are equal, then there is no difference between the arrival rates of queues in the co-operating sets. We discussed that the difference in queue lengths between queues in these sets is an important metric. Thus an intuitive "measure" of non-uniformity of the arrival rates is the difference between arrival rates of queues in these sets. Hence, in nonuniform traffic model, we study the cases when the differences between arrival rates of queues in these pairs are nonzero.

First, we consider the case where the arrival rates are unequal for one pair, and equal for the queues in the other pair. Specifically, $\lambda_{11} = 7\lambda_{22}$, $\lambda_{12} = \lambda_{21}$ and $\lambda_{11} = 3.5\lambda_{12}$. We denote this pattern as "NU1." We compare the performances of PSMD with the optimal for different values of the departure rates (μ) (Figure 3), and for each μ we choose the arrival rates such that the above ratios are satisfied and $\mu + \sum_{i=1}^{2}\sum_{j=1}^{2}\lambda_{ij} = 1$ ($\lambda_{22} = \frac{1-\mu}{15}$ ensure this). Figure 3 show that the performances of the two are almost identical in most cases with the difference between the loss probabilities less than 1% (3%) in most (all) cases. The maximum difference is 2.8% and this happens for a high value of service rate when the blocking probabilities are less than 3% for both PSMD and the optimal. Next, we consider the case where the arrival rates are unequal for both pairs, while the total arrival rates are equal for both inputs, i.e, $\lambda_{11} + \lambda_{12} = \lambda_{21} + \lambda_{22}$ (this did not hold in the previous case). Here, $\lambda_{11} = 4\lambda_{22}$, $\lambda_{21} = 4\lambda_{12}$ and $\lambda_{12} = \lambda_{22}$. We denote this pattern as "NU2." Again, we vary μ and for each

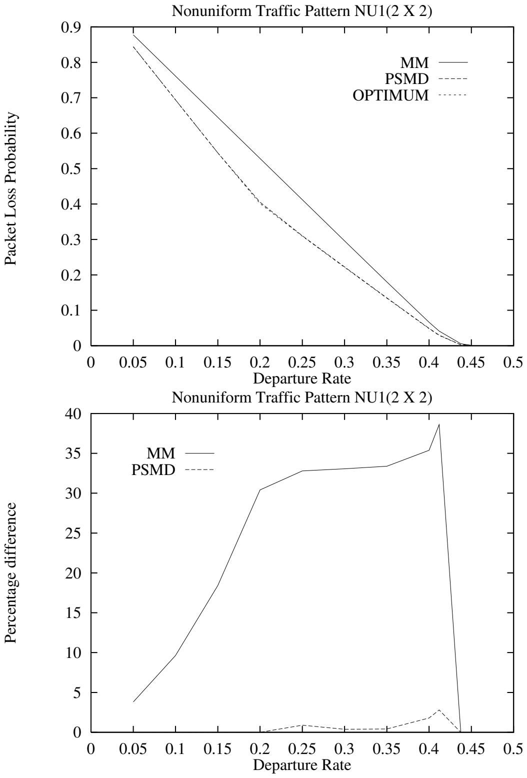

Figure 3: The figures compare the performances of PSMD and MM with the optimal strategy for different departure rates (μ) for $N = 2, B = 50$ and the following arrival rate patterns: $\lambda_{11} = 3.5\lambda_{12} = 3.5\lambda_{21} = 7\lambda_{22}, \lambda_{22} = \frac{1-\mu}{15}$. Figure (a) plots the absolute values of the job loss probabilities ($l_{\text{MM}}, l_{\text{PSMD}}, l_{\text{OPT}}$). The PSMD and the OPTIMAL curves can not be distinguished. Figure (b) plots the percentage relative difference between the loss probabilities for the optimal and PSMD (curve PSMD) $(100 * (l_{\text{PSMD}}/l_{\text{OPT}} - 1))$, and that between PSMD and MM (curve MM) $(100 * (l_{\text{MM}}/l_{\text{PSMD}} - 1))$.

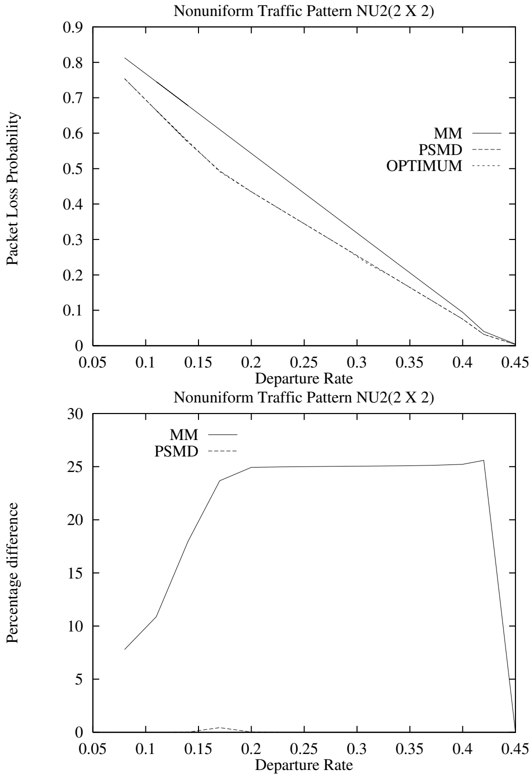

Figure 4: The figures compare the performances of PSMD and MM with the optimal strategy for different departure rates (μ) for $N = 2, B = 50$ and the following arrival rate patterns: $\lambda_{11} = \lambda_{21} = 4\lambda_{12} = 4\lambda_{22}, \lambda_{22} = \frac{1-\mu}{10}$. Figure (a) plots the absolute values of the job loss probabilities($l_{\text{MM}}, l_{\text{PSMD}}, l_{\text{OPT}}$). The PSMD and the OPTIMAL curves can not be distinguished. Figure (b) plots the percentage relative difference between the loss probabilities for the optimal and PSMD (curve PSMD) $(100 * (l_{\text{PSMD}}/l_{\text{OPT}} - 1))$, and that between PSMD and MM (curve MM) $(100 * (l_{\text{MM}}/l_{\text{PSMD}} - 1))$.

μ we choose the arrival rates such that the above ratios are satisfied and $\mu + \sum_{i=1}^{2} \sum_{j=1}^{2} \lambda_{ij} = 1$ ($\lambda_{22} = \frac{1-\mu}{10}$ ensure this). Figure 4 show that the performance of PSMD is near optimal all through. The trends remain similar for other patterns of arrival rates which we do not exhibit on account of space constraints.

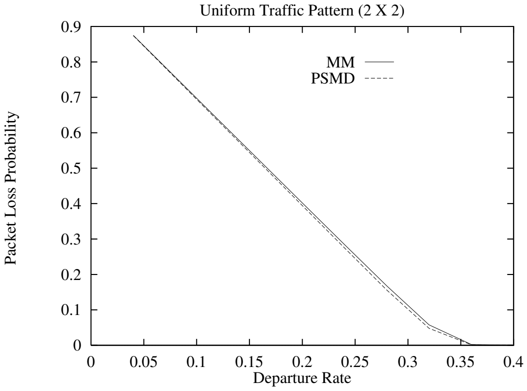

Figure 5: The figure plots the job loss probabilities of MM and PSMD for different departure rates (μ), and equal arrival rates $\lambda_{ij} = \frac{1-\mu}{4}, \forall i, j$ for $N = 2, B = 50$. Note that PSMD is optimal in this case (Theorem 1).

Now we consider the MM policy which schedules jobs as per a maximum weighted matching. This policy has been proposed for the infinite buffer case and attains the maximum possible throughput there[6, 11]. Memory management is not an issue under infinite buffer assumption, and as such MM does not address it. We consider the obvious extension, whereby MM always accepts a job whenever there is space in the buffer and always rejects a job if the input buffer is full. Push out is not used. Average job loss experienced by this policy can be computed like that for PSMD, using MDP based techniques. Numerical computation shows that MM is strictly suboptimal and has considerably higher job loss than PSMD for unequal arrival rates. For traffic patterns NU1 and NU2 (Figures 3 and 4 respectively), PSMD decreases the job loss by more than 30% and 25% for NU1 and NU2 respectively. Note that the difference brought about by intelligent resource allocation is pronounced for medium departure rates. For large departure rates, all policies attain low job loss and all policies suffer from heavy job loss in the other extreme. Thus MM and PSMD perform similarly in these two extremes, while PSMD substantially outperforms MM in the middle. We observed similar trends for other unequal arrival rate patterns as well. When arrival rates are equal, MM is consistently inferior to PSMD for all departure rates(Figure 5). However, the difference in performance is small in this case, and thus the curves look identical in the figure.

The relative performance of MM w.r.t. PSMD is much worse for nonuniform traffic (Figures 3 and 4) than for uniform traffic (Figure 5). This happens as the queues tend to be heavily dis-balanced when arrival rates are unequal, while the discrepancy is less for equal arrival rates. This load mismatch worsens the performance for MM. Thus MM experiences heavier job loss for non-uniform traffic than for uniform traffic. The memory management in PSMD restores a balance in the queue lengths by replacing jobs of more heavily loaded queues with those of the lightly loaded queues whenever possible, and this retains the performance for nonuniform traffic at the same level as the uniform traffic case. As an example, at service rate 0.2, sum of the arrival rates 0.8, MM has 52.84% and 40.19% job loss for equal arrival rates and arrival pattern NU1 respectively, while the numbers are 40.52% and 39.44% for PSMD. This shows that PSMD is robust to different traffic patterns (note that neither PSMD nor MM uses statistics information in the decision process).

4 Resource allocation for arbitrary N

We consider the scheduling and memory management problems for arbitrary N. The objective is to minimize the job loss. No closed form solution is known for this case. We present a heavy traffic lower bound for the optimal discounted cost function $J_\beta(\vec{x})$, and subsequently propose a computationally simple heuristic which minimizes this lower bound. Numerical computations show that this heuristic attains low job loss.

The objective of the scheduling policy is to transfer as many jobs as possible so as to minimize the job loss, and memory management decides which jobs to accept, reject and push-out so as to allow the transfer of several jobs in every scheduling. Consider the example shown in Figure 1. All three jobs can be transferred at the same time since they are waiting at different inputs and intended for different outputs. However, if all three were waiting at the same input, only one can be transferred. Thus like in the special case for $N = 2$ we need to balance the load across appropriate queues. The memory management attains this in PSMD by accepting/rejecting/pushing out jobs so as to reduce the difference between queue lengths of the queues which can be scheduled together($1-1, 2-2$ and $1-2, 2-1$). For $N > 2$ more than two queues can be scheduled together. For example, $1-1, 2-2, 3-3$ can be scheduled together for $N = 3$ (Figure 1). Thus the notion of reducing these differences can not be extended directly for $N > 2$.

We consider a new measure of congestion, the maximum of the total number of jobs waiting at an input and waiting for an output. We denote this measure as $\mathrm{cong}(\vec{x})$, where $\mathrm{cong}(\vec{x}) = \max(\max_i \sum_{j=1}^{N} x_{ij}, \max_j \sum_{i=1}^{N} x_{ij})$, and $\vec{x}$ is the current state. Note that $\mathrm{cong}(\vec{x}) = 1$ in Figure 1, $\mathrm{cong}(\vec{x}) = 3$ if all three jobs were waiting at the same input, or waiting for the same output in Figure 1. Three jobs can be transferred to the outputs simultaneously in the first case, while only one job can be transferred in the latter cases. Thus intuitively larger number of jobs can be transferred when $\mathrm{cong}(\vec{x})$ is smaller. We propose a strategy which minimizes this congestion measure. First we justify why this is a good measure to minimize. If the current state is $\vec{x}$, every job has duration T units and there is no future arrival, then the minimum time taken to transfer all waiting jobs is $T\,\mathrm{cong}(\vec{x})$[5, 13]. This indicates that higher the value of $\mathrm{cong}(\vec{x})$, more congested is the switch in some sense. Next, we present lemma 1 which shows the optimal discounted cost function $J_\beta(\vec{x})$ is lower bounded by $\mathrm{cong}(\vec{x})$ under heavy traffic.

Lemma 1 *Let the total arrival rate at(intended for) each input(output) is greater than the service rate, i.e.,* $\min(\min_i \sum_{j=1}^{N} \lambda_{ij}, \min_j \sum_{i=1}^{N} \lambda_{ij}) > \mu$. *Then for all* $\beta \in [\beta_0 1)$, $J_{\beta,OPT}(\vec{x}) \geq \mathrm{cong}(\vec{x})$ *for all states* $\vec{x}$, *where* $\beta_0 = \frac{B}{B+\min(\min_i \sum_{j=1}^{N} \lambda_{ij}, \min_j \sum_{i=1}^{N} \lambda_{ij}) - \mu}$.

We prove Lemma 1 in appendix C. Lemma 1 motivates the design of a heuristic approach which minimizes this congestion metric in every step. The scheduling is to serve the matching of largest size which minimizes $\mathrm{cong}(\vec{y})$ of the next state $\vec{y}$. Note that there can be several maximum matchings or matchings of the largest size. We choose a specific maximum matching which minimizes the congestion measure of the next state. Under such a matching, the congestion measure reduces by one whenever the currently scheduled jobs complete transmission. Fast algorithms can be found for computing such matchings using the theory of edge coloring of bipartite graphs [4]. The memory management policy accepts jobs without pushing out any existing job as long as the input buffer has space (in fact this is the optimal decision as stated in concluding parts of appendix C). We consider the acceptance decision when an incoming job finds the input buffer full. Let the current state be $\vec{x}$. Suppose the job is for input i and output j. The policy considers the state $\vec{y} = \vec{x} + \vec{e}_{ij}$. Note that this would have been the next state had the new job been accepted. A queue is selected for job drop so as to reduce $\mathrm{cong}(\vec{y})$ by one, i.e., a job is dropped from the most congested terminal for state $\vec{y}$. Overall, the policy minimizes the congestion at both inputs and outputs, and thus we denote it as the "minimum congestion" policy (MC). We present an example to illustrate this policy.

Example 4.1: Let $N = 3$. Let the current state be $\vec{x}$ with $x_{11} = 2, x_{21} = x_{12} = x_{23} = 1$. All other queues are empty. Note that at most 2 jobs can be transferred, and the matchings which transfer two jobs are $1-1, 2-3$ or $1-2, 2-1$ or $1-2, 2-3$. The next states have congestion measures $2, 2, 3$ respectively for these choices. Thus either the first or the second matching can be selected. If $B > 5$ then any new arrival is accepted without any push out. Suppose $B = 5$. A new arrival for input 1 or output 1 is rejected. However, a new arrival for input 3, output 3 is accepted while pushing out a job from queue $1-1$.

MC applies for the $N = 2$ case as well, and takes the same decisions as PSMD there. Weller *et al.*[13] proposed a similar scheduling strategy, and analyzed its throughput in presence of infinite buffers and pseudo-deterministic traffic arrival (a "$\alpha - S$" traffic model)[13]. However, memory constraint was not considered there, and as such the memory management component has not been proposed.

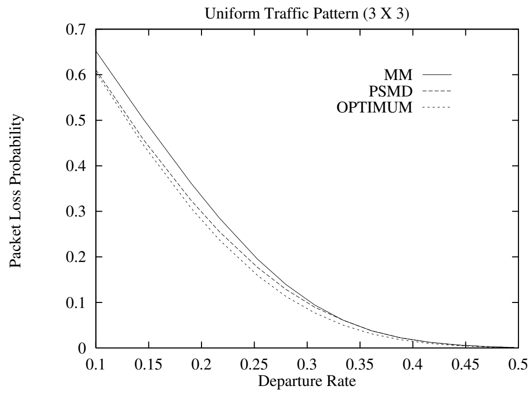

Figure 6: The figure plots the loss probabilities of MC, MM and the optimal strategy for different departure rates (μ), and equal arrival rates $\lambda_{ij} = \frac{1-\mu}{9}, \forall i, j$ for $N = 3, B = 10$.

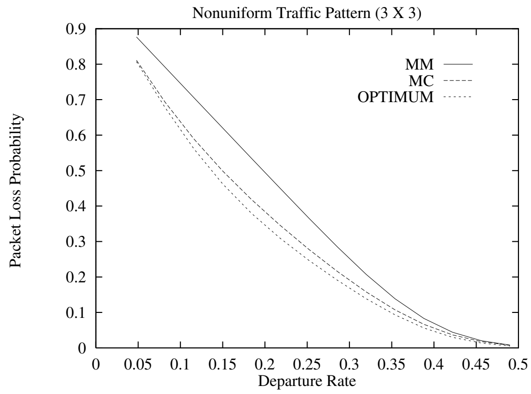

Figure 7: The figure compares the performances of MC and MM with the optimal strategy for different departure rates (μ) for $N = 3, B = 10$ and the following arrival rate patterns: $\lambda_{11} = 9\lambda, \lambda_{ij} = \lambda, (i,j) \neq (1,1), \lambda = \frac{1-\mu}{17}$. The MC and the OPTIMAL curves are close. MM has substantially higher loss probability than both MC and the optimal strategies.

Average job loss experienced by this policy can be computed using MDP based techniques. We present the numerical performance evaluations for $N = 3$. We first compare the performance of MC with the optimal. However, computation of the minimum possible job loss experienced using MDP involves several iterations, each involving 10^B for buffer size B (refer to concluding discussions of Section 2). We could thus compare the performances for small values of B ($B = 10$) only. First consider the case of uniform traffic $\lambda_{ij} = \lambda, \forall i, j, \lambda = \frac{1-\mu}{9}$ for scaling purposes. Figure 6 shows that the job loss of MC is close to that of the optimal for all departure rates, μ. Now consider a nonuniform traffic pattern where the arrival rate in one queue is significantly higher than others. Specifically, $\lambda_{11} = 9\lambda$, $\lambda_{ij} = \lambda$ for $(i, j) \neq (1, 1)$ and $\lambda = (1 - \mu)/17$ (the last relation scales the sum of the arrival and departure rates to 1). Figure 7 shows that the job loss of MC is close to that of the optimal for all departure rates.

A key observation is that MC performs better than the existing heuristic MM for both uniform and non-uniform traffic (Figures 6 and 7). The performance difference is higher for the nonuniform traffic case. MC decreases the job loss rate by more than 30% over MM (Figure 7) in the nonuniform traffic case. Once again, we observe similar differences in performances between MC and MM for other nonuniform traffic patterns we considered. We omit the figures for space constraint. Like in the $N = 2$ case, the load balancing brought about by the memory management of MC safeguards against the detrimental effect of the difference in arrival rates, while MM has no such protection, and this explains the difference in performance for nonuniform traffic patterns. We conclude that MC is more robust than MM.

5 Conclusion and Discussion

We summarize our contribution in this paper. We investigated a jointly optimal scheduling and memory management problem in a system with N inputs and N outputs where the service scheduling is governed by matching constraints. We presented a closed form optimal scheduling and memory management strategy (PMMD) for $N = 2$ and equal arrival rates. Subsequently, we obtain certain properties of the optimal strategy for more general traffic patterns for $N = 2$, and propose PMMD as a heuristic in this case. Numerical computations show that PMMD attains near optimal job loss and substantially outperforms the maximum weighted matching based scheduling (MM) which maximizes throughput in the same system under infinite storage assumption. We obtain a heavy traffic lower bound for the optimal discounted loss function for the general case of arbitrary N and arbitrary arrival rates. We design a heuristic MC which minimizes this lower bound. Numerical computations show that MC attains significantly lower job loss than MM. Future research will be directed towards designing the optimal strategy for a multi-stage network. Investigation for non-preemptive service models forms an interesting topic of future research.

References

[1] D. Bertsekas. *Dynamic Programming and Optimal Control*. Athina Scientific, 1995.

[2] I. Cidon, L. Georgiadis, R. Guerin, and A. Khamisy. Optimal buffer sharing. *IEEE Journal on Selected Areas in Communications*, 13(7), 1995.

[3] F.Kamoun and L.Kleinrock. Analysis of shared storage in a computer network node environment under general traffic conditions. *IEEE Transactions on Communications*, 28(7), July 1980.

[4] H. Gabow and O. Kariv. Algorithms for edge coloring bipartite graphs and multigraphs. *SIAM Journal of Computing*, 11(1), February 1992.

[5] M. Hall Jr. *Combinatorial Theory*. John Wiley and Sons, 1998.

[6] N. Mckeown, A. Mekkittikul, V. Anantharam, and J. Walrand. Achieving 100% throughput in an input queued switch. *IEEE Transactions on Communications*, 47(8), 1999.

[7] M. Putterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley and Sons, 1998.

[8] S. Ross. *Introduction to Stochastic Dynamic Programming*. Academic Press, 1998.

[9] P. Sparaggis, C. Cassandras, and D. Towsley. On the duality between routing and scheduling systems with finite buffer space. *Proc. 31st IEEE Conf. Decision and Control, pp. 2364-2365*, December 1992.

[10] J. Suk and C. Cassandras. Optimal scheduling of two competing queues with blocking. *IEEE Trans. on Automatic Control pp. 1086-1091*, 36(9), 1991.

[11] L. Tassiulas and A. Ephremidis. Stability properties of constrained queueing systems and scheduling for maximum throughput in multihop radio networks. *IEEE Transactions on Automatic Control*, 37(12), 1992.

[12] D. Towsley, P. Sparaggis, and C. Cassandras. Optimal routing and buffer allocation for a class of finite capacity queueing systems. *IEEE Trans. on Automatic Control pp. 1446-1451*, 37(9), 1992.

[13] T. Weller and B. Hajek. Scheduling nonuniform traffic in a packet switching system with small propagation delay. *Proceedings of INFOCOM'1994*, 1994.

A Proof of Theorem 1

We will prove certain properties of the optimal discounted cost function $J_{\beta,\mathrm{OPT}}(\vec{x})$ for each $\beta < 1$, in a sequence of Lemmas (Lemmas 2 to 6). These properties will lead to Theorem 1. The standard technique in proving a specific property is to show that the property holds for $J_{\beta,k}(\vec{x})$ in iteration (8) for all $\beta < 1$ and k, using induction. More specifically, the induction argument is to show that the property holds for the function $f(\vec{x}) = 0$, $\forall \vec{x}$ and subsequently assume the property for $J_{\beta,k}(\vec{x})$ and show it for $TJ_{\beta,k}(\vec{x})$. The result holds for $J_{\beta,k}(\vec{x})$ for all k by induction, if the initial values of these iterates are chosen as zero. It follows that $J_{\beta,\mathrm{OPT}}(\vec{x})$ satisfies the same property. For simplicity we will drop the index β and k henceforth.

We first obtain the following relations using (3). Note that $r^+ = \max(r,0)$ and $\vec{x}^+ = (x_{11}^+, x_{12}^+, x_{21}^+, x_{22}^+)$.

$$
\begin{aligned}
TJ(\vec{x}) &= \mu\phi(\vec{x}) + \sum_{i=1}^{2}\sum_{j=1}^{2}\lambda_{ij}\theta_{ij}(\vec{x}) && (9)\\
\phi(\vec{x}) &= \beta \min\Bigg(\min_{\substack{ij\\ x_{ij}>0}} J(\vec{x}-\vec{e}_{ij}),\\
&\quad J\left((\vec{x}-\vec{e}_{11}-\vec{e}_{22})^+\right),\\
&\quad J\left((\vec{x}-\vec{e}_{12}-\vec{e}_{21})^+\right)\Bigg) && (10)\\
\theta_{ij}(\vec{x}) &= \min_{\vec{d}:0\le d_{ij}\le x_{ij}}\left(\sum_{i=1}^{2}\sum_{j=1}^{2} d_{ij} + \beta J(\vec{x}-\vec{d})\right) && (11)
\end{aligned}
$$

Lemma 2 $J(\vec{x}) \le J(\vec{x}+e_{ij}) \le J(\vec{x})+1, \forall\, i,j.$

Since Lemma 2 holds for all $\beta < 1$, from (7), irrespective of the arrival and service rates, the optimal job acceptance policy is to accept all new arrivals without any push outs as long as the input buffer has space.

We now prove Lemma 2. The intuition behind the result is that the optimal cost increases with increase in queue lengths of the states. However, addition of one job can increase the cost by at most one since the original state can be reached by dropping the additional job and incurring a penalty of one.

Proof: The result holds trivially if $J(\vec{x}) = 0$ for all $\vec{x}$. Now we show the induction case. We assume that the inequalities hold for $J(\vec{x})$ for all $(\vec{x})$ and then show the result for $TJ(\vec{x})$ for all $(\vec{x})$. It follows from this assumption and the fact that $\beta < 1$, that

$$
\begin{aligned}
\phi(\vec{x}) &= \beta \min\left(J(\vec{x}-\vec{e}_{11}-\vec{e}_{22})^+,\right.\\
&\qquad\left. J(\vec{x}-\vec{e}_{12}-\vec{e}_{21})^+\right)\\
\theta_{11}(\vec{x}) &= \beta J(\vec{x}+\vec{e}_{11}) \text{ if } \sum_{i=1}^{2}\sum_{j=1}^{2} x_{ij} < B\\
\theta_{11}(\vec{x}) &= 1 + \beta \min_{\substack{l,m:\\ (\vec{x}+\vec{e}_{11})_{lm}>0}} J(\vec{x}+\vec{e}_{11}-\vec{e}_{lm})\\
&\qquad \text{if } \sum_{i=1}^{2}\sum_{j=1}^{2} x_{ij} = B
\end{aligned}
$$

The functions $\theta_{ij}(\vec{x})$ for $(i,j) \neq (1,1)$ can be constructed similarly.

Now we compare corresponding terms in $TJ(\vec{x})$ and $TJ(x_{11}+1, x_{12}, x_{21}, x_{22})$. Note that $\phi(x_{11}+1, x_{12}, x_{21}, x_{22}) = \beta \min\left(J\left(x_{11}, x_{12}, x_{21}, (x_{22}-1)^+\right),\right.$ $\left. J\left(x_{11}+1, (x_{12}-1)^+, (x_{21}-1)^+, x_{22}\right)\right)$. Thus, by hypothesis,

$$
\phi(\vec{x}) \le \phi(x_{11}+1, x_{12}, x_{21}, x_{22}) \le \phi(\vec{x}) + 1 \qquad (12)
$$

Since $\sum_{i=1}^{2}\sum_{j=1}^{2} x_{ij} + 1 \le B$, $\theta_{kl}(\vec{x}) = \beta J(x_{11}, \ldots, x_{kl}+1, \ldots, x_{22})$, for each k,l. Now, we consider two cases separately, $\sum_{i=1}^{2}\sum_{j=1}^{2} x_{ij} + 1 < B$, and $\sum_{i=1}^{2}\sum_{j=1}^{2} x_{ij} + 1 = B$. In the former case, $\theta_{11}(x_{11}+1, x_{12}, x_{21}, x_{22}) = \beta J(x_{11}+2, x_{12}, x_{21}, x_{22})$, and for $(k,l) \neq (1,1)$, $\theta_{kl}(x_{11}+1, \ldots, x_{kl}+1, \ldots, x_{22}) = \beta J(x_{11}+1, \ldots, x_{kl}+1, \ldots, x_{22})$. Thus, by hypothesis, $\theta_{kl}(\vec{x}) \le \theta_{kl}(x_{11}+1, x_{12}, x_{21}, x_{22}) \le \theta_{kl}(\vec{x}) + 1$.

Now consider the case $\sum_{i=1}^{2}\sum_{j=1}^{2} x_{ij} + 1 = B$. First consider $\theta_{11}(\vec{x})$ and $\theta_{11}(x_{11}+1, x_{12}, x_{21}, x_{22})$. Now $\theta_{11}(x_{11}+1, x_{12}, x_{21}, x_{22}) = 1 + \beta \min\left(J(x_{11}+1, x_{12},\right.$ $x_{21}, x_{22}), \min_{i,j:x_{ij}>0,(i,j)\neq(1,1)} J(x_{11}+2, \ldots x_{ij}-1,$ $\left.\ldots x_{22})\right)$. Thus, $\theta_{11}(\vec{x}) \le 1 + \theta_{11}(x_{11}+1, x_{12}, x_{21}, x_{22})$. Now,

$$
\begin{aligned}
\theta_{11}(\vec{x}) &= \beta J(x_{11}+1, x_{12}, x_{21}, x_{22})\\
&\le \beta J(x_{11}+1, \ldots x_{ij}-1, \ldots, x_{22}) + 1\\
&\qquad \text{if } x_{ij} > 0, \text{ (by hypothesis)}\\
&\le \beta J(x_{11}+2, \ldots x_{ij}-1, \ldots, x_{22}) + 1\\
&\qquad \text{(by hypothesis)}
\end{aligned}
$$

It follows that $\theta_{11}(\vec{x}) \le \theta_{11}(x_{11}+1, x_{12}, x_{21}, x_{22})$.

Next we compare $\theta_k l(\vec{x})$ and $\theta_{11}(x_{11}+1, x_{12}, x_{21}, x_{22})$ for other k,l and when $\sum_{i=1}^{2}\sum_{j=1}^{2} x_{ij} + 1 = B$. For brevity we consider the case for $(k,l) = (1,2)$. The argument is similar for other cases. Recall that $\theta_{12}(\vec{x}) = \beta J(x_{11}, x_{12}+1, x_{21}, x_{22})$, and $\theta_{12}(x_{11}+1, x_{12}, x_{21}, x_{22}) = 1 + \beta \min\left(J(x_{11}+1, x_{12}, x_{21}, x_{22}),\right.$ $J(x_{11}, x_{12}+1, x_{21}, x_{22}), \min_{i,x_{2i}>0} J(x_{11}+1, x_{12}+1, \ldots$ $\left. x_{2i}-1, \ldots)\right)$. Thus, $\theta_{12}(\vec{x}) \le 1 + \theta_{12}(x_{11}+1, x_{12}, x_{21}, x_{22})$. Again,

$$
\begin{aligned}
\theta_{12}(\vec{x}) &= \beta J(x_{11}, x_{12}+1, x_{21}, x_{22}) && (13)\\
&\le \beta J(\vec{x}) + 1 \text{ (by hypothesis)}\\
&\le \beta J(x_{11}+1, x_{12}, x_{21}, x_{22}) + 1 && (14)\\
&\qquad \text{(by hypothesis)}\\
\theta_{12}(\vec{x}) &\le 1 + \beta J(x_{11}, x_{12}+1, \ldots x_{2i}-1, \ldots)\\
&\qquad \text{if } x_{2i} > 0 \text{ (by (13) and hypothesis)}\\
&\le 1 + \beta J(x_{11}+1, x_{12}+1, \ldots x_{2i}-1, \ldots) && (15)\\
&\qquad \text{(by hypothesis)}
\end{aligned}
$$

From (14) and (15), $\theta_{12}(\vec{x}) \le \theta_{12}(x_{11}+1, x_{12}, x_{21}, x_{22})$. We have thus shown that

$$
\theta_{ij}(\vec{x}) \le \theta_{ij}(x_{11}+1, x_{12}, x_{21}, x_{22}) \le \theta_j(\vec{x}) + 1 \forall j, \qquad (16)
$$

The result follows from (12) and (16) and the fact that $\mu + \sum_i \lambda_i = 1$.

From Lemma 2, equations (10) and (11)and the fact that $\beta < 1$, we have,

$$
\begin{aligned}
\phi(\vec{x}) &= \beta \min\left(J(\vec{x}-\vec{e}_{11}-\vec{e}_{22})^+,\right.\\
&\qquad\left. J(\vec{x}-\vec{e}_{12}-\vec{e}_{21})^+\right) && (17)
\end{aligned}
$$

$$\theta_{ij}(\vec{x}) = \beta J(\vec{x}+\vec{e}_{ij}) \text{ if } \sum_{i=1}^{2}\sum_{j=1}^{2} x_{ij} < B \quad (18)$$

$$\theta_{ij}(\vec{x}) = 1+\beta \min_{k,l:(\vec{x}+\vec{e}_{ij})_{kl}>0} J(\vec{x}-\vec{e}_{kl}+\vec{e}_{ij}) \text{ if } \sum_{i=1}^{2}\sum_{j=1}^{2} x_{ij} = B, \quad (19)$$

Lemma 3

$$\begin{aligned}
J(\vec{x}-\vec{e}_{12}-\vec{e}_{21}) &\le J(\vec{x}-e_{11}) \textit{ if } x_{22}=0, \\
& x_{ij}>0 \textit{ for } (i,j)\neq(2,2) \quad (20)\\
J(\vec{x}-\vec{e}_{12}-\vec{e}_{21}) &\le J(\vec{x}-e_{22}) \textit{ if } x_{11}=0, \\
& x_{ij}>0 \textit{ for } (i,j)\neq(1,1) \quad (21)\\
J(\vec{x}-\vec{e}_{11}-\vec{e}_{22}) &\le J(\vec{x}-e_{21}) \textit{ if } x_{12}=0, \\
& x_{ij}>0 \textit{ for } (i,j)\neq(1,2) \quad (22)\\
J(\vec{x}-\vec{e}_{11}-\vec{e}_{22}) &\le J(\vec{x}-e_{21}) \textit{ if } x_{21}=0, \\
& x_{ij}>0 \textit{ for } (i,j)\neq(2,1) \quad (23)
\end{aligned}$$

From Lemmas 2, 3 and relation (17),

$$\begin{aligned}
\phi(\vec{x}) &= \beta J(\vec{x}-\vec{e}_{11}-\vec{e}_{22}) \\
& \quad \text{if } \min(x_{11},x_{22}) > \min(x_{12},x_{21}) = 0 \quad (24)\\
\phi(\vec{x}) &= \beta J\left(\vec{x}-\vec{e}_{12}-\vec{e}_{21}\right) \\
& \quad \text{if } \min(x_{12},x_{21}) > \min(x_{11},x_{22}) = 0 \quad (25)\\
\phi(\vec{x}) &= \beta \min(J(\vec{x}-\vec{e}_{11}-\vec{e}_{22}), \\
& \quad J(\vec{x}-\vec{e}_{12}-\vec{e}_{21})), \text{ if } x_{ij}>0 \forall i,j \quad (26)\\
\phi(r\vec{e}_{ij}) &= \beta J\left((r-1)\vec{e}_{ij}\right), \text{ for all } r\ge 1 \quad (27)
\end{aligned}$$

Since Lemmas 2 and 3 hold for all $\beta < 1$, from (6), the optimal scheduling is to schedule a pair of queues whenever possible irrespective of the arrival and service rates. The results in Lemma 3 is intuitive as scheduling a pair of queues as opposed to serving only one queue reduces the buffer occupancy and hence the job loss. We present the formal proof now.

Proof: We will prove only (20). The rest will follow similarly. The result holds trivially if $J(\vec{x}) = 0$ for all $\vec{x}$. Now we show the induction case. We assume that (20) holds for $J(\vec{x})$ for all $(\vec{x})$ and then show the result for $TJ(\vec{x})$ for all $\vec{x}$.

Consider $\vec{x}$ such that $x_{22} = 0$ and $\min(x_{11}, x_{12}, x_{21}) > 0$. From (17) $\phi(\vec{x}-\vec{e}_{12}-\vec{e}_{21}) \le \beta J\left(\vec{x}-\vec{e}_{11}-\vec{e}_{12}-\vec{e}_{21}\right)$. Now, $\phi(\vec{x}-e_{11}) = \beta\min\left(J\left(\vec{x}-\vec{e}_{11}-\vec{e}_{12}-\vec{e}_{21}\right),\right.$ $\left.J\left((x_{11}-2)^{+}, x_{12}, x_{21}, 0\right)\right)$. If $x_{11}-1 > 0$, then $J\left(\vec{x}-\vec{e}_{11}-\vec{e}_{12}-\vec{e}_{21}\right) \le J\left((x_{11}-2)^{+}, x_{12}, x_{21}, 0\right)$ by hypothesis. If $x_{11}-1=0$, then the inequality again holds from Lemma 2. Thus $J\left(x_{11}-1, x_{12}-1, x_{21}-1, 0\right) \le J\left((x_{11}-2)^{+}, x_{12}, x_{21}, 0\right)$. Thus

$$\phi(x_{11}, x_{12}-1, x_{21}-1, 0) \le \phi(x_{11}-1, x_{12}, x_{21}, 0) \quad (28)$$

If $x_{11}+x_{12}+x_{21} \le B$, then $\theta_{ij}(x_{11}, x_{12}-1, x_{21}-1, 0) \le \theta_{ij}(x_{11}-1, x_{12}, x_{21}, 0)$ from (18) and induction hypothesis. Let $x_{11}+x_{12}+x_{21} = B+1$ ($x_{11}+x_{12}+x_{21} \le B+1$ since $(x_{11}-1, x_{12}, x_{21}, 0)$ is a valid state).

$$\theta_{ij}(x_{11}, x_{12}-1, x_{21}-1, 0) = \beta J(\vec{x}-\vec{e}_{12}-\vec{e}_{21}+\vec{e}_{ij}) \quad (29)$$

From (29), Lemma 2 and the fact that $\beta < 1$

$$\begin{aligned}
\theta_{ij}(\vec{x}-\vec{e}_{12}-\vec{e}_{21}) &< 1+\beta J(\vec{x}-\vec{e}_{12}-\vec{e}_{21}) \quad (30)\\
\theta_{ij}(\vec{x}-\vec{e}_{12}-\vec{e}_{21}) &< 1+\beta J(\vec{y}_{ij}-\vec{e}_{lm}) \text{ where} \\
& \quad \vec{y}_{ij} = \vec{x}-\vec{e}_{12}-\vec{e}_{21}+\vec{e}_{ij}, \\
& \quad (l,m)\neq(i,j), (\vec{y}_{ij})_{lm} > 0 \quad (31)
\end{aligned}$$

$$\begin{aligned}
\theta_{ij}(\vec{x}-\vec{e}_{11}) &= 1+\beta \min_{l,m:(\vec{y}_{ij})_{lm}>0} J(\vec{y}_{ij}-e_{lm}) \\
& \quad \text{where } \vec{y}_{ij} = \vec{x}-\vec{e}_{11}+\vec{e}_{ij}, \quad (32)
\end{aligned}$$

The last equality follows from (19). From (29) to (32) using the fact that $x_{11} \ge 1$,

$$\theta_{ij}(\vec{x}-\vec{e}_{12}-\vec{e}_{21}) \le \theta_{ij}(\vec{x}-\vec{e}_{11}-1) \quad (33)$$

The lemma follows from (28) and (33).

Recall that $\text{diff}_1(\vec{x}) = |x_{11}-x_{22}|$, $\text{diff}_2(\vec{x}) = |x_{12}-x_{21}|$. We introduce some notations which we will use often. Let $\text{queue}(\vec{x}) = \sum_{i=1}^{N}\sum_{j=1}^{N} x_{ij}$ be the total number of jobs in the buffer under state $\vec{x}$, i.e., $\text{queue}(\vec{x}) = \sum_{i=1}^{2}\sum_{j=1}^{2} x_{ij}$. Let $\text{maxsum}(\vec{x}) = \max(x_{11}, x_{22}) + \max(x_{12}, x_{21})$. Henceforth, we will derive properties of the optimal policy assuming complete symmetry: $\lambda_{ij} = \lambda$.

Lemma 4 *The following results hold:*

1. $J(\vec{x}) = J(x_{22}, x_{12}, x_{21}, x_{11}) = J(x_{11}, x_{21}, x_{12}, x_{22})$.
2. *For any two feasible states $\vec{x}$ and $\vec{y}$ such that* $\text{diff}_i(\vec{x}) \le \text{diff}_i(\vec{y})$, $i=1,2$, $x_{11}+x_{22} = y_{11}+y_{22}$, *and* $x_{12}+x_{21} = y_{12}+y_{21}$, $J(\vec{x}) \le J(\vec{y})$.

Lemma 5 *The following results hold:*

1. $J(\vec{x}) = J(x_{12}, x_{11}, x_{22}, x_{21})$.
2. *For any two feasible states $\vec{x}$ and $\vec{y}$ such that* $\text{queue}(\vec{x}) = \text{queue}(\vec{y})$, $\max_i \text{diff}_i(\vec{x}) \le \max_i \text{diff}_i(\vec{y})$, $\text{maxsum}(\vec{x}) \le \text{maxsum}(\vec{y})$, $J(\vec{x}) \le J(\vec{y})$.

Since Lemmas 4 and 5 hold for all $\beta < 1$, it follows from (7) that the job acceptance policy of PSMD for zero remaining storage is optimal under equal arrival rates. From Lemma 5 and (6) for equal arrival rates, when only one queue can be *scheduled, the optimal scheduling is to schedule the largest queue.* The intuition behind both these lemmas is that job loss probability decreases if the difference functions are reduced without increasing the queue occupancies. We prove these results more formally now.

Consider two feasible states: $\vec{x}$ and $\vec{y}$ such that $x_{11} + x_{22} = y_{11} + y_{22}$, and $x_{12} + x_{21} = y_{12} + y_{21}$. Then $|\text{diff}_i(\vec{x}) - \text{diff}_i(\vec{y})|$ is even for $i = 1, 2$. Proofs of lemmas 4 and 5 use this fact. The proof of Lemma 4 needs partial symmetry assumptions only, i.e., $\lambda_{11} = \lambda_{22}$ and $\lambda_{12} = \lambda_{21}$.

Proof of Lemma 4: The results hold trivially if $J(\vec{x}) = 0$ for all $\vec{x}$. We will assume all these properties for the function $J(\vec{x})$ and prove these for the function $TJ(\vec{x})$.

Proof of part (1): Now we show the induction case. We first prove the first equality in property (1) for $TJ(\vec{x})$. Note that $\phi(\vec{x}) = \phi(x_{22}, x_{12}, x_{21}, x_{11})$, from part (1) of hypothesis and (17). From induction hypothesis, (18) and (19) it follows that $\theta_{11}(\vec{x}) = \theta_{22}(x_{22}, x_{12}, x_{21}, x_{11})$, $\theta_{22}(\vec{x}) = \theta_{11}(x_{22}, x_{12}, x_{21}, x_{11})$, and $\theta_{ij}(\vec{x}) = \theta_{ij}(x_{22}, x_{12}, x_{21}, x_{11})$ if $i \neq j$. The result follows since $\lambda_{11} = \lambda_{22}$. The argument is similar for the second equality. There we show that $J(\vec{x}) = J(x_{11}, x_{21}, x_{12}, x_{22})$. We show that $\phi(\vec{x}) = \phi(x_{11}, x_{21}, x_{12}, x_{22})$, from part (1) of hypothesis and (17). From induction hypothesis, (18) and (19) it follows that $\theta_{12}(\vec{x}) = \theta_{21}(x_{11}, x_{21}, x_{12}, x_{22})$, $\theta_{21}(\vec{x}) = \theta_{12}(x_{11}, x_{21}, x_{12}, x_{22})$, and $\theta_{ii}(\vec{x}) = \theta_{ii}(x_{11}, x_{21}, x_{12}, x_{22})$ for $i = 1, 2$. The result follows since $\lambda_{12} = \lambda_{21}$.

Proof of part (2): From part (1), it is sufficient to consider the case when $x_{11} - x_{22}$, $y_{11} - y_{22}$ and $x_{12} - x_{21}$, $y_{12} - y_{21}$ have the same signs. Thus without loss of generality, we assume that $x_{11} \geq x_{22}$, $x_{12} \geq x_{21}$, $y_{11} \geq y_{22}$ and $y_{12} \geq y_{21}$. Since $x_{11} + x_{22} = y_{11} + y_{22}$, $\text{diff}_1(\vec{x}) = x_{11} - x_{22}$ (since $x_{11} \geq x_{22}$), $\text{diff}_1(\vec{y}) = y_{11} - y_{22}$ (since $y_{11} \geq y_{22}$) and $\text{diff}_1(\vec{x}) \leq \text{diff}_1(\vec{y})$, if $x_{22} = 0$, $y_{22} = 0$ and $x_{11} = y_{11}$. Similarly, if $x_{21} = 0$, $y_{21} = 0$ and $x_{12} = y_{12}$.

First we will show that $\phi(\vec{x}) \leq \phi(\vec{y})$. First consider the case that $\max(\min(x_{11}, x_{22}), \min(x_{12}, x_{21})) = 0$. This means $x_{22} = x_{21} = 0$. From the previous argument, $\vec{x} = \vec{y}$. Thus $\phi(\vec{x}) = \phi(\vec{y})$ trivially. Now let $\min(x_{11}, x_{22}) > \min(x_{12}, x_{21}) = 0$. Thus $y_{12} = x_{12}$ and $y_{21} = x_{21} = 0$. Assume that $y_{22} > 0$. ¿From (24), $\phi(\vec{x}) = \beta J(\vec{p})$ and $\phi(\vec{y}) = \beta J(\vec{q})$, where $p_{11} = x_{11} - 1, p_{22} = x_{22} - 1$, $p_{ij} = x_{ij}$ if $i \neq j$, $q_{11} = y_{11} - 1$, $q_{22} = y_{22} - 1$, $q_{ij} = y_{ij}$ if $i \neq j$. It follows that $p_{11} + p_{22} = q_{11} + q_{22}$ and $p_{12} + p_{21} = q_{12} + q_{21}$. Also, $\text{diff}_i(\vec{p}) = \text{diff}_i(\vec{x})$ and $\text{diff}_i(\vec{q}) = \text{diff}_i(\vec{y})$. Thus $\text{diff}_i(\vec{p}) \leq \text{diff}_i(\vec{q})$ for $i = 1, 2$. From induction hypothesis, $J(\vec{p}) \leq J(\vec{q})$, and the desired inequality involving the ϕ function follows. The proof is similar for the case $\min(x_{12}, x_{21}) > \min(x_{11}, x_{22}) = 0$. Now consider the case that $\min(x_{12}, x_{21}) > 0$ and $\min(x_{11}, x_{22}) > 0$, i.e., $x_{ij} > 0$ for all i. First assume $\max(\min(y_{11}, y_{22}), \min(y_{12}, y_{21})) = 0$. It follows that $y_{22} = y_{21} = 0$. Let $\vec{p} = \vec{x} - \vec{e}_{11}, \vec{q} = \vec{y} - \vec{e}_{11}, \vec{r} = \vec{y} - \vec{e}_{22}$ and $\vec{s} = \vec{y} - \vec{e}_{22}$. Since $x_{ij} > 0$ for all i, j $y_{11} > 0, y_{12} > 0$. From (17) and Lemma 2, $\phi(\vec{y}) = \beta \min(J(\vec{r}), J(\vec{s}))$ and $\phi(\vec{x}) \leq \beta \min(J(\vec{p}, J(\vec{q}))$. Note that $p_{11} + p_{22} = r_{11} + r_{22}$, $p_{12} + p_{21} = r_{12} + r_{21}$, $\text{diff}_i(\vec{r}) = \text{diff}_i(\vec{y}) - 1$, $|\text{diff}_i(\vec{p}) - \text{diff}_i(\vec{x})| = 1$, $\text{diff}_i(\vec{x}) \leq \text{diff}_i(\vec{y}) - 2$, $i = 1, 2$. The last inequality follows since $\text{diff}_i(\vec{x}) < \text{diff}_i(\vec{y})$ as $x_{ij} > 0$ for all i, j, $y_{22} = y_{21} = 0$ and hence $y_{11} = x_{11} + x_{22}$ and $y_{12} = x_{12} + x_{21}$. It follows that $J(\vec{p}) \leq J(\vec{r})$. By similar arguments $J(\vec{q}) \leq J(\vec{s})$. It follows that $\phi(\vec{x}) \leq \phi(\vec{y})$ in this case. Now consider the case that $\max(\min(y_{11}, y_{22}), \min(y_{12}, y_{21})) > 0$. In this case, either $\min(y_{11}, y_{22}) > 0$ or $\min(y_{12}, y_{21}) > 0$. Let $\vec{p} = \vec{x} - \vec{e}_{11} - \vec{e}_{22}, \vec{q} = \vec{y} - \vec{e}_{12} - \vec{e}_{21}, \vec{r} = \vec{y} - \vec{e}_{11} - \vec{e}_{22}$ and $\vec{s} = \vec{y} - \vec{e}_{12} - \vec{e}_{21}$. From (24) to (26) either $\phi(\vec{y}) = \beta J(\vec{r})$ or $\phi(\vec{y}) = \beta J(\vec{s})$. From (26) $\phi(\vec{x}) = \beta \min(J(\vec{p}), J(\vec{q}))$. Note that $p_{11} + p_{22} = r_{11} + r_{22}$, $p_{12} + p_{21} = r_{12} + r_{21}$, $\text{diff}_i(\vec{p}) = \text{diff}_i(\vec{x})$, $\text{diff}_i(\vec{r}) = \text{diff}_i(\vec{y})$, $i = 1, 2$. Hence $\text{diff}_i(\vec{p}) \leq \text{diff}_i(\vec{r})$, $i = 1, 2$. It follows that $J(\vec{p}) \leq J(\vec{r})$. By similar arguments $J(\vec{q}) \leq J(\vec{s})$. It follows that $\phi(\vec{x}) \leq \phi(\vec{y})$ in this case. We thus conclude that

$$\phi(\vec{x}) \leq \phi(\vec{y}) \tag{34}$$

Now we show that $\theta_{ij}(\vec{x}) \leq \theta_{ij}(\vec{y})$. First consider the case that $\text{queue}(\vec{x}) < B$. It follows that $\text{queue}(\vec{y}) < B$. From (18) $\theta_{ij}(\vec{x}) = \beta J(\vec{x} + \vec{e}_{ij})$ and $\theta_{ij}(\vec{y}) = \beta J(\vec{y} + \vec{e}_{ij})$. Let $\vec{p} = \vec{x} + \vec{e}_{ij}$ and $\vec{q} = \vec{y} + \vec{e}_{ij}$. Clearly $p_{11} + p_{22} = q_{11} + q_{22}$ and $p_{12} + p_{21} = q_{12} + q_{21}$. If $\text{diff}_1(\vec{x}) = \text{diff}_1(\vec{y})$ then $x_{11} = y_{11}$ and $x_{22} = y_{22}$. This follows since $x_{11} + x_{22} = y_{11} + y_{22}$, $x_{11} \geq x_{22}$ and $y_{11} \geq y_{22}$. Thus, if $\text{diff}_1(\vec{x}) = \text{diff}_1(\vec{y})$ then $p_{11} = q_{11}$ and $p_{22} = q_{22}$, and hence $\text{diff}_1(\vec{p}) = \text{diff}_1(\vec{q})$. If $\text{diff}_1(\vec{x}) \neq \text{diff}_1(\vec{y})$, $\text{diff}_1(\vec{y}) \geq \text{diff}_1(\vec{x}) + 2$, as argued before. Note that $|\text{diff}_i(\vec{x}) - \text{diff}_i(\vec{p})| = 1$ and $|\text{diff}_i(\vec{y}) - \text{diff}_i(\vec{q})| = 1$, $i = 1, 2$. Thus $\text{diff}_1(\vec{p}) \leq \text{diff}_1(\vec{q})$. Using similar arguments, it can be shown that $\text{diff}_2(\vec{p}) \leq \text{diff}_2(\vec{q})$. Thus from induction hypothesis $J(\vec{p}) \leq J(\vec{q})$. Thus $\theta_{ij}(\vec{x}) \leq \theta_{ij}(\vec{y})$.

Now we consider the case $\text{queue}(\vec{x}) = B$. It follows that $\text{queue}(\vec{y}) = B$. We will show that $\theta_{ij}(\vec{x}) \leq \theta_{ij}(\vec{y})$, for $i = j$. The result follows similarly for $i \neq j$. First consider the case $i = j = 1$. ¿From (19) $\theta_{11}(\vec{y}) = \beta k$ where $k \in \{J(\vec{y}), J(\vec{y} + \vec{e}_{11} - \vec{e}_{22}), J(\vec{y} + \vec{e}_{11} - \vec{e}_{12}), J(\vec{y} + \vec{e}_{11} - \vec{e}_{21})\}$. From (19), $\theta_{11}(\vec{x}) \leq \beta J(\vec{x})$. From induction hypothesis $J(\vec{x}) \leq J(\vec{y})$. Let $k = J(\vec{y} + \vec{e}_{11} - \vec{e}_{22})$. Let $\vec{p} = \vec{y} + \vec{e}_{11} - \vec{e}_{22}$. Note that $p_{11} + p_{22} = y_{11} + y_{22}$ and $p_{12} + p_{21} = y_{12} + y_{21}$. It follows that $p_{11} + p_{22} = x_{11} + x_{22}$ and $p_{12} + p_{21} = x_{12} + x_{21}$. Also, $\text{diff}_1(\vec{p}) = \text{diff}_1(\vec{y}) + 2$ since $y_{11} \geq y_{22}$, and $\text{diff}_2(\vec{p}) = \text{diff}_2(\vec{y})$. Thus $\text{diff}_i(\vec{x}) \leq \text{diff}_i(\vec{p})$. It follows that $J(\vec{x}) \leq J(\vec{p})$. In both these cases $\theta_{11}(\vec{x}) \leq \theta_{11}(\vec{y})$. Now let $k = J(\vec{y} + \vec{e}_{11} - \vec{e}_{12})$. If $x_{12} > 0$ then from (19), $\theta_{11}(\vec{x}) \leq \beta J(\vec{x} + \vec{e}_{11} - \vec{e}_{12})$. If $x_{12} = 0$, then $x_{21} = 0$ and hence $y_{12} = y_{21} = 0$. But then $k \neq J(\vec{y} + \vec{e}_{11} - \vec{e}_{12})$ as $\vec{y} + \vec{e}_{11} - \vec{e}_{12}$ is not a valid state. Thus $x_{12} > 0$. Let $\vec{p} = \vec{x} + \vec{e}_{11} - \vec{e}_{12}$ and $\vec{q} = \vec{y} + \vec{e}_{11} - \vec{e}_{12}$. It is possible to show that $p_{11} + p_{22} = q_{11} + q_{22}$ and $p_{12} + p_{21} = q_{12} + q_{21}$. Also, $\text{diff}_1(\vec{p}) = \text{diff}_1(\vec{x}) + 1$ and $\text{diff}_1(\vec{q}) = \text{diff}_1(\vec{y}) + 1$. Thus

$\mathrm{diff}_1(\vec{p}) \le \mathrm{diff}_1(\vec{q})$. If $\mathrm{diff}_2(\vec{x}) = \mathrm{diff}_2(\vec{y})$, then $x_{12} = y_{12}$ and $x_{21} = y_{21}$ and thus $p_{12} = q_{12}$ and $p_{21} = q_{21}$. It follows that $\mathrm{diff}_2(\vec{q}) = \mathrm{diff}_2(\vec{p})$. Otherwise $\mathrm{diff}_2(\vec{x}) \le \mathrm{diff}_2(\vec{y}) - 2$. We know that $|\mathrm{diff}_i(\vec{x}) - \mathrm{diff}_i(\vec{p})| = 1$ and $|\mathrm{diff}_i(\vec{y}) - \mathrm{diff}_i(\vec{q})| = 1, i = 1, 2$. Thus $\mathrm{diff}_1(\vec{p}) \le \mathrm{diff}_1(\vec{q})$. Thus $J(\vec{p}) \le J(\vec{q})$ by induction hypothesis. Thus $\theta_{11}(\vec{x}) \le \theta_{11}(\vec{y})$ in this case. Finally let $k = J(\vec{y} + \vec{e}_{11} - \vec{e}_{21})$. This means $y_{21} > 0$. Thus $x_{12} + x_{21} \ge y_{21} > 0$ and hence $x_{12} > 0$ since $x_{12} \ge x_{21}$. Thus from (19), $\theta_{11}(\vec{x}) \le \beta J(\vec{x} + \vec{e}_{11} - \vec{e}_{12})$. Let $\vec{p} = \vec{x} + \vec{e}_{11} - \vec{e}_{12}$ and $\vec{q} = \vec{y} + \vec{e}_{11} - \vec{e}_{21}$. It is possible to show that $p_{11} + p_{22} = q_{11} + q_{22}$ and $p_{12} + p_{21} = q_{12} + q_{21}$. Also, $\mathrm{diff}_1(\vec{p}) = \mathrm{diff}_1(\vec{x}) + 1$ and $\mathrm{diff}_1(\vec{q}) = \mathrm{diff}_1(\vec{y}) + 1$. Thus $\mathrm{diff}_1(\vec{p}) \le \mathrm{diff}_1(\vec{q})$. Here, $\mathrm{diff}_2(\vec{q}) = \mathrm{diff}_2(\vec{y}) + 1$. Now, $\mathrm{diff}_2(\vec{p}) \in \{\mathrm{diff}_2(\vec{x}) + 1, \mathrm{diff}_2(\vec{x}) - 1\}$. It follows that $\mathrm{diff}_2(\vec{p}) \le \mathrm{diff}_2(\vec{q})$. Thus $J(\vec{p}) \le J(\vec{q})$ by induction hypothesis. Thus $\theta_{11}(\vec{x}) \le \theta_{11}(\vec{y})$ in this case.

Now consider the case $i = j = 2$. From (19) $\theta_{22}(\vec{y}) = \beta k$ where $k \in \{J(\vec{y}), J(\vec{y} + \vec{e}_{22} - \vec{e}_{11}), J(\vec{y} + \vec{e}_{22} - \vec{e}_{12}), J(\vec{y} + \vec{e}_{22} - \vec{e}_{21})\}$. From (19), $\theta_{11}(\vec{y}) \le \beta J(\vec{x})$. From induction hypothesis $J(\vec{x}) \le J(\vec{y})$. Let $k = J(\vec{y} + \vec{e}_{22} - \vec{e}_{11})$. Let $\vec{q} = \vec{y} + \vec{e}_{22} - \vec{e}_{11}$. First let $x_{11} \in \{x_{22}, x_{22} + 1\}$. Note that $q_{11} + q_{22} = x_{11} + x_{22}$ and $q_{12} + q_{21} = x_{12} + x_{21}$ and $\mathrm{diff}_2(\vec{q}) = \mathrm{diff}_2(\vec{y}) \ge \mathrm{diff}_2(\vec{x})$. Note that $\mathrm{diff}_1(\vec{x}) = 0$ if $x_{11} = x_{22}$ and thus $\mathrm{diff}_1(\vec{x}) \le \mathrm{diff}_1(\vec{q})$ in this case. If $x_{11} = x_{22} + 1$ then either $y_{11} = y_{22} + 1$ or $y_{11} \ge y_{22} + 3$ (since $\mathrm{diff}_2(\vec{y}) = 1, 3, \ldots$ in this case). In the first case $\mathrm{diff}_1(\vec{x}) = \mathrm{diff}_1(\vec{q}) = 1$. In the second case $\mathrm{diff}_1(\vec{x}) = 1$ and $\mathrm{diff}_1(\vec{q}) = \mathrm{diff}_1(\vec{y}) - 2 \ge 1$. It follows that $\mathrm{diff}_1(\vec{x}) \le \mathrm{diff}_1(\vec{q})$. Thus $J(\vec{x}) \le J(\vec{q})$ by induction hypothesis. Thus $\theta_{22}(\vec{x}) \le \theta_{22}(\vec{y})$ in this case. Now let $x_{11} > x_{22} + 1$. It follows that $x_{11} > 0$. Thus from (19), $\theta_{22}(\vec{x}) \le \beta J(\vec{x} + \vec{e}_{22} - \vec{e}_{11})$. Let $\vec{p} = \vec{x} + \vec{e}_{11} - \vec{e}_{22}$. Note that $p_{11} + p_{22} = q_{11} + q_{22}$ and $p_{12} + p_{21} = q_{12} + q_{21}$ and $\mathrm{diff}_2(\vec{p}) = \mathrm{diff}_2(\vec{x}) \le \mathrm{diff}_2(\vec{y}) = \mathrm{diff}_2(\vec{q})$. Since $x_{11} > x_{22} + 1, y_{11} > y_{22} + 1$ as $\mathrm{diff}_1(\vec{x}) \ge \mathrm{diff}_1(\vec{y})$ and $y_{11} \ge y_{22}$. Thus $\mathrm{diff}_1(\vec{p}) = \mathrm{diff}_1(\vec{x}) - 2$ and $\mathrm{diff}_1(\vec{q}) = \mathrm{diff}_1(\vec{y}) - 2$. It follows that $\mathrm{diff}_1(\vec{p}) \le \mathrm{diff}_1(\vec{q})$. Thus $J(\vec{p}) \le J(\vec{q})$ by induction hypothesis. Thus $\theta_{22}(\vec{x}) \le \theta_{22}(\vec{y})$ in this case. Now let $k = J(\vec{y} + \vec{e}_{22} - \vec{e}_{12})$. Thus $y_{12} > 0$ and hence $x_{12} > 0$. From (19), $\theta_{22}(\vec{x}) \le \beta J(\vec{x} + \vec{e}_{22} - \vec{e}_{12})$. Let $\vec{p} = \vec{x} + \vec{e}_{22} - \vec{e}_{12}$ and $\vec{q} = \vec{y} + \vec{e}_{22} - \vec{e}_{12}$. It is possible to show that $p_{11} + p_{22} = q_{11} + q_{22}$ and $p_{12} + p_{21} = q_{12} + q_{21}$. Let $x_{11} = x_{22}$. If $y_{11} = y_{22}$ then $\mathrm{diff}_1(\vec{p}) = \mathrm{diff}_1(\vec{q}) = 1$. Otherwise, $y_{11} \ge y_{22} + 2$. Thus $\mathrm{diff}_1(\vec{q}) = \mathrm{diff}_1(\vec{y}) - 1 \ge 1$ and $\mathrm{diff}_1(\vec{p}) = 1$. Thus $\mathrm{diff}_1(\vec{p}) \le \mathrm{diff}_1(\vec{q})$. If $x_{11} > x_{22}, y_{11} > y_{22}$. Thus $\mathrm{diff}_1(\vec{q}) = \mathrm{diff}_1(\vec{y}) - 1$ and $\mathrm{diff}_1(\vec{p}) = \mathrm{diff}_1(\vec{x}) - 1$. It follows that $\mathrm{diff}_1(\vec{p}) \le \mathrm{diff}_1(\vec{q})$. Using similar arguments it can be shown that $\mathrm{diff}_2(\vec{p}) \le \mathrm{diff}_2(\vec{q})$. Thus $J(\vec{p}) \le J(\vec{q})$ by induction hypothesis. Thus $\theta_{22}(\vec{x}) \le \theta_{22}(\vec{y})$ in this case. Finally let $k = J(\vec{y} + \vec{e}_{22} - \vec{e}_{21})$. This means $y_{21} > 0$. Thus $x_{12} + x_{21} \ge y_{21} > 0$ and hence $x_{12} > 0$ since $x_{12} \ge x_{21}$. Thus from (19), $\theta_{22}(\vec{x}) \le \beta J(\vec{x} + \vec{e}_{11} - \vec{e}_{12})$. Let $\vec{p} = \vec{x} + \vec{e}_{22} - \vec{e}_{12}$ and $\vec{q} = \vec{y} + \vec{e}_{22} - \vec{e}_{21}$ It is possible to show that $p_{11} + p_{22} = q_{11} + q_{22}$ and $p_{12} + p_{21} = q_{12} + q_{21}$. As in the previous case, $\mathrm{diff}_1(\vec{p}) \le \mathrm{diff}_1(\vec{q})$. Here, $\mathrm{diff}_2(\vec{q}) = \mathrm{diff}_2(\vec{y}) + 1$. Now, $\mathrm{diff}_1(\vec{p}) \in \{\mathrm{diff}_2(\vec{x}) + 1, \mathrm{diff}_2(\vec{x}) - 1\}$. Thus $\mathrm{diff}_2(\vec{p}) \le \mathrm{diff}_2(\vec{q})$. Thus $J(\vec{p}) \le J(\vec{q})$ by induction hypothesis. Thus $\theta_{22}(\vec{x}) \le \theta_{22}(\vec{y})$ in this case. Thus

$$\theta_{ij}(\vec{x}) \le \theta_{ij}(\vec{y}) \tag{35}$$

The result follows from (34) and (35).

Proof of Lemma 5: The results hold trivially if $J(\vec{x}) = 0$ for all $\vec{x}$. Now we show the induction case. We will assume all these properties for the function $J(\vec{x})$ and prove these for the function $TJ(\vec{x})$.

Proof of part (1): Note that $\phi(\vec{x}) = \phi(x_{12}, x_{11}, x_{22}, x_{21})$, from part (1) of hypothesis and (17). From induction hypothesis, (18) and (19) it follows that $\theta_{i1}(\vec{x}) = \theta_{i2}(x_{12}, x_{11}, x_{22}, x_{21})$, and $\theta_{i2}(\vec{x}) = \theta_{i1}(x_{12}, x_{11}, x_{22}, x_{21})$, $i = 1, 2$. The result follows since $\lambda_{i1} = \lambda_{i2}, i = 1, 2$.

Proof of part (2): From part (1), it is sufficient to consider the case when $\max_i \mathrm{diff}_i(\vec{x}) = |x_{11} - x_{22}|$, and $\max_i \mathrm{diff}_i(\vec{y}) = |y_{11} - y_{22}|$. Also, from part (1) of Lemma 2 without loss of generality, we assume that $x_{11} \ge x_{22}$, $x_{12} \ge x_{21}, y_{11} \ge y_{22}$ and $y_{12} \ge y_{21}$. It follows that

$$\begin{aligned}
x_{11} + x_{12} &= y_{11} + y_{12} - c, c \ge 0 && (36)\\
x_{22} + x_{21} &= y_{22} + y_{21} + c && (37)\\
&\quad \text{(from (36) and since}\\
&\quad \mathrm{queue}(\vec{x}) = \mathrm{queue}(\vec{y}))\\
\sum_{i=1}^{2} \mathrm{diff}_i(\vec{x}) &= \sum_{i=1}^{2} \mathrm{diff}_i(\vec{y}) - 2c && (38)\\
&\quad \text{(from (36) and (37))}
\end{aligned}$$

We make a few observations using (36) to (38). If $\max_i \mathrm{diff}_i(\vec{x}) = \min_i \mathrm{diff}_i(\vec{x})$, then either $\vec{x} = \vec{y}$ or $\max_i \mathrm{diff}_i(\vec{y}) > \max_i \mathrm{diff}_i(\vec{x})$ and $\max_i \mathrm{diff}_i(\vec{y}) > \max_i \mathrm{diff}_i(\vec{x})$. Let $\max_i \mathrm{diff}_i(\vec{y}) = \max_i \mathrm{diff}_i(\vec{x})$. Then from (38), $\min_i \mathrm{diff}_i(\vec{y}) = \min_i \mathrm{diff}_i(\vec{x})$. Thus $x_{ij} = x$ and $y_{ij} = y$. Since $\mathrm{queue}(\vec{x}) = \mathrm{queue}(\vec{y})$ $x = y$ and thus $\vec{x} = \vec{y}$. Since the proof is only required for $\vec{x} \ne \vec{y}$, we will always assume that $\max_i \mathrm{diff}_i(\vec{y}) > \max_i \mathrm{diff}_i(\vec{x})$. If $\max_i \mathrm{diff}_i(\vec{y}) = \max_i \mathrm{diff}_i(\vec{x})$, since $\max_i \mathrm{diff}_i(\vec{x}) = \min_i \mathrm{diff}_i(\vec{x})$, from (38) $\min_i \mathrm{diff}_i(\vec{y}) = \min_i \mathrm{diff}_i(\vec{x})$. Since $\mathrm{queue}(\vec{x}) = \mathrm{queue}(\vec{y}), \vec{x} = \vec{y}$.

The next observation is as follows. If $\min_i \mathrm{diff}_i(\vec{x}) = 0$, then either $\vec{x} = \vec{y}$ or $\mathrm{maxsum}(\vec{x}) < \mathrm{maxsum}(\vec{y})$, and $\max_i \mathrm{diff}_i(\vec{y}) > 0$. If $\mathrm{maxsum}(\vec{x}) = \mathrm{maxsum}(\vec{y})$ then (38) holds with $c = 0$. Since $\max_i \mathrm{diff}_i(\vec{y}) \ge \max_i \mathrm{diff}_i(\vec{x})$, this means $\min_i \mathrm{diff}_i(\vec{y}) \le \min_i \mathrm{diff}_i(\vec{x}) = 0$. Thus $\min_i \mathrm{diff}_i(\vec{y}) = \min_i \mathrm{diff}_i(\vec{x})$ and since $c = 0$ in (38) $\max_i \mathrm{diff}_i(\vec{y}) = \max_i \mathrm{diff}_i(\vec{x})$. It follows that $\vec{x} = \vec{y}$. Thus we can rule this case out and assume that $\mathrm{maxsum}(\vec{x}) < \mathrm{maxsum}(\vec{y})$. If $\max_i \mathrm{diff}_i(\vec{y}) = 0$, then it follows that $\vec{x} = \vec{y} = (x, x, x, x)$ for some $x \ge 0$. Thus we can rule this case out and assume that $\max_i \mathrm{diff}_i(\vec{y}) > 0$.

We first show that $\phi(\vec{x}) \leq \phi(\vec{y})$. From (37) $\max(\min(y_{11}, y_{22}), \min(y_{12}, y_{21})) = 0$ if $\max(\min(x_{11}, x_{22}), \min(x_{12}, x_{21})) = 0$. First we assume that $\max(\min(x_{11}, x_{22}), \min(x_{12}, x_{21})) = 0$. Hence $\max(\min(y_{11}, y_{22}), \min(y_{12}, y_{21})) = 0$. Thus $\vec{x} = (x_{11}, x_{12}, 0, 0)$ and $\vec{y} = (y_{11}, y_{12}, 0, 0)$. In this case $c = 0$ in (36), (37) and (38). From (36) and the assumptions, $y_{11} \geq x_{11} \geq x_{12} \geq y_{12}$. If $x_{11} = 0$ then $\vec{x} = \vec{y} = \vec{0}$. If $y_{11} = x_{11}$, then from (36) $y_{12} = x_{12}$ and thus $\vec{x} = \vec{y}$. In both cases, $\phi(\vec{x}) = \phi(\vec{y})$. So we assume that $y_{11} > x_{11} > 0$. From (17), $\phi(\vec{x}) \leq \beta \min(J(\vec{p}))$ and $\phi(\vec{y}) = \beta k$, where $k \in \{J(\vec{q}), J(\vec{r})\}$, $\vec{p} = \vec{x} - \vec{e}_{11}$, $\vec{q} = \vec{y} - \vec{e}_{11}$ and $\vec{r} = \vec{y} - \vec{e}_{12}$. Thus $\vec{p} = (x_{11} - 1, x_{12}, 0, 0)$, $\vec{q} = (y_{11} - 1, y_{12}, 0, 0)$ and $\vec{r} = (y_{11}, y_{12} - 1, 0, 0)$. Clearly, $\mathrm{maxsum}(\vec{p}) = \mathrm{maxsum}(\vec{q})$ and $\mathrm{queue}(\vec{p}) = \mathrm{queue}(\vec{q})$. Now, $\max_i \mathrm{diff}_i(\vec{p}) \in \{x_{11} - 1, x_{11}\}$ and $\max_i \mathrm{diff}_i(\vec{q}) = y_{11} - 1$ (since $y_{11} > y_{12}$). We know that $y_{11} - 1 \geq x_{11}$. Thus $\max_i \mathrm{diff}_i(\vec{p}) \leq \max_i \mathrm{diff}_i(\vec{q})$. By induction hypothesis, $J(\vec{p}) \leq J(\vec{q})$. Now $\mathrm{maxsum}(\vec{p}) = \mathrm{maxsum}(\vec{r})$ and $\mathrm{queue}(\vec{p}) = \mathrm{queue}(\vec{r})$. Also, $\max_i \mathrm{diff}_i(\vec{r}) = y_{11}$ (since $y_{11} > y_{12}$). Since $y_{11} > x_{11}$, $\max_i \mathrm{diff}_i(\vec{p}) < \max_i \mathrm{diff}_i(\vec{r})$. If $\vec{r}$ is feasible, then by induction hypothesis, $J(\vec{p}) \leq J(\vec{r})$. Thus, $\phi(\vec{x}) \leq \phi(\vec{y})$.

Now let $\max(\min(y_{11}, y_{22}), \min(y_{12}, y_{21})) > 0$ and $\max(\min(x_{11}, x_{22}), \min(x_{12}, x_{21})) > 0$. From (17), $\phi(\vec{x}) = \beta J(\vec{a})$ where $\vec{a} \in \{\vec{p}, \vec{q}\}$ and $\phi(\vec{y}) = \beta J(\vec{b})$ where $\vec{b} \in \{\vec{r}, \vec{s}\}$, $\vec{p} = \vec{x} - \vec{e}_{11} - \vec{e}_{22}$, $\vec{q} = \vec{x} - \vec{e}_{12} - \vec{e}_{21}$, $\vec{r} = \vec{y} - \vec{e}_{11} - \vec{e}_{22}$, and $\vec{s} = \vec{y} - \vec{e}_{12} - \vec{e}_{21}$. Let $\vec{a} \in \{\vec{p}, \vec{q}\}$ and $\vec{b} \in \{\vec{r}, \vec{s}\}$. Note that $\mathrm{maxsum}(\vec{a}) = \mathrm{maxsum}(\vec{x}) - 1$ and $\mathrm{maxsum}(\vec{b}) = \mathrm{maxsum}(\vec{y}) - 1$. Also, $\mathrm{queue}(\vec{a}) = \mathrm{queue}(\vec{x}) - 2$ and $\mathrm{queue}(\vec{b}) = \mathrm{queue}(\vec{y}) - 2$, and $\mathrm{diff}_i(\vec{a}) = \mathrm{diff}_i(\vec{x})$, $\mathrm{diff}_i(\vec{b}) = \mathrm{diff}_i(\vec{y})$, $i = 1, 2$. It follows that $\mathrm{maxsum}(\vec{a}) = \mathrm{maxsum}(\vec{b})$, $\mathrm{queue}(\vec{a}) = \mathrm{queue}(\vec{b})$ and $\max_i \mathrm{diff}_i(\vec{a}) \leq \max_i \mathrm{diff}_i(\vec{b})$. By induction hypothesis, $J(\vec{a}) \leq J(\vec{b})$. Thus, $\phi(\vec{x}) \leq \phi(\vec{y})$.

Now consider the case that $\max(\min(x_{11}, x_{22}), \min(x_{12}, x_{21})) > 0$ and $\max(\min(y_{11}, y_{22}), \min(y_{12}, y_{21})) = 0$. Since $\vec{x} \neq \vec{0}$, $x_{11} > 0$. Now $y_{21} = y_{22} = 0$. Since $\vec{x} \neq \vec{0}$, $\vec{y} \neq \vec{0}$ and thus $y_{11} > 0$. From (17) $\phi(\vec{y}) = \beta J(\vec{a})$ where $\vec{a} \in \{\vec{q}, \vec{r}\}$, $\vec{q} = \vec{y} - \vec{e}_{11}$ and $\vec{r} = \vec{y} - \vec{e}_{12}$. From (10) $\phi(\vec{x}) \leq \beta J(\vec{p})$, where $\vec{p} = \vec{x} - \vec{e}_{11}$. Note that $\mathrm{queue}(\vec{p}) = \mathrm{queue}(\vec{q}) = \mathrm{queue}(\vec{r}) = \mathrm{queue}(\vec{x}) - 1$. Let $\max_i \mathrm{diff}_i(\vec{x}) > \min_i \mathrm{diff}_i(\vec{x})$. Thus $\mathrm{maxsum}(\vec{p}) = \mathrm{maxsum}(\vec{x}) - 1$ and $\mathrm{maxsum}(\vec{a}) \geq \mathrm{maxsum}(\vec{y}) - 1$. Thus $\mathrm{maxsum}(\vec{a}) \geq \mathrm{maxsum}(\vec{p})$. Now $\max_i \mathrm{diff}_i(\vec{p}) = \max_i \mathrm{diff}_i(\vec{x}) - 1$. Note that $\max_i \mathrm{diff}_i(\vec{a}) \geq \max_i \mathrm{diff}_i(\vec{y}) - 1$. Thus $\max_i \mathrm{diff}_i(\vec{a}) \geq \max_i \mathrm{diff}_i(\vec{p})$. By induction hypothesis, $J(\vec{p}) \leq J(\vec{a})$. Thus, $\phi(\vec{x}) \leq \phi(\vec{y})$. Now let $\max_i \mathrm{diff}_i(\vec{x}) = \min_i \mathrm{diff}_i(\vec{x})$. It follows that $\max_i \mathrm{diff}_i(\vec{y}) > \max_i \mathrm{diff}_i(\vec{x})$. Initially let $\min_i \mathrm{diff}_i(\vec{x}) > 0$. Now $\mathrm{maxsum}(\vec{p}) = \mathrm{maxsum}(\vec{x}) - 1$ and $\mathrm{maxsum}(\vec{a}) \geq \mathrm{maxsum}(\vec{y}) - 1$. Thus $\mathrm{maxsum}(\vec{a}) \geq \mathrm{maxsum}(\vec{p})$. Now $\max_i \mathrm{diff}_i(\vec{p}) = \max_i \mathrm{diff}_i(\vec{x})$ and $\max_i \mathrm{diff}_i(\vec{a}) \geq \max_i \mathrm{diff}_i(\vec{y}) - 1$. Since $\max_i \mathrm{diff}_i(\vec{y}) > \max_i \mathrm{diff}_i(\vec{x})$, $\max_i \mathrm{diff}_i(\vec{a}) \geq \max_i \mathrm{diff}_i(\vec{p})$. By induction hypothesis, $J(\vec{p}) \leq J(\vec{a})$. Thus, $\phi(\vec{x}) \leq \phi(\vec{y})$. Finally, consider $\min_i \mathrm{diff}_i(\vec{x}) = 0$. It follows that $\mathrm{maxsum}(\vec{y}) > \mathrm{maxsum}(\vec{x})$. From (38) it follows that $\sum_i \mathrm{diff}_i(\vec{y}) \geq 2$. Thus either $\mathrm{diff}_i(\vec{y}) \geq 1$, $i = 1, 2$ or $\max_i \mathrm{diff}_i(\vec{y}) \geq 2$. Now $\mathrm{maxsum}(\vec{p}) = \mathrm{maxsum}(\vec{x})$ and $\mathrm{maxsum}(\vec{a}) \geq \mathrm{maxsum}(\vec{y}) - 1$. Thus $\mathrm{maxsum}(\vec{a}) \geq \mathrm{maxsum}(\vec{p})$ since $\mathrm{maxsum}(\vec{y}) > \mathrm{maxsum}(\vec{x})$. Here $\max_i \mathrm{diff}_i(\vec{p}) = 1$, $\max_i \mathrm{diff}_i(\vec{a}) \geq 1$ since either $\mathrm{diff}_i(\vec{y}) \geq 1$, $i = 1, 2$ or $\max_i \mathrm{diff}_i(\vec{y}) \geq 2$. Thus $\max_i \mathrm{diff}_i(\vec{a}) \geq \max_i \mathrm{diff}_i(\vec{p})$. By induction hypothesis, $J(\vec{p}) \leq J(\vec{a})$. Thus, $\phi(\vec{x}) \leq \phi(\vec{y})$.

We thus conclude that

$$\phi(\vec{x}) \leq \phi(\vec{y}) \tag{39}$$

Now we show that $\theta_{ij}(\vec{x}) \leq \theta_{ij}(\vec{y})$. We consider each (i, j) separately.

First let $(i, j) = (1, 1)$. Consider the case that $\mathrm{queue}(\vec{x}) < B$. It follows that $\mathrm{queue}(\vec{y}) < B$. From (18), $\theta_{11}(\vec{x}) = \beta J(\vec{x} + \vec{e}_{11})$ and $\theta_{11}(\vec{y}) = \beta J(\vec{y} + \vec{e}_{11})$. Let $\vec{p} = \vec{x} + \vec{e}_{11}$ and $\vec{q} = \vec{y} + \vec{e}_{11}$. Note that $\mathrm{maxsum}(\vec{p}) = \mathrm{maxsum}(\vec{x}) + 1$ and $\mathrm{maxsum}(\vec{q}) = \mathrm{maxsum}(\vec{y}) + 1$. Also, $\mathrm{queue}(\vec{p}) = \mathrm{queue}(\vec{x})$ and $\mathrm{queue}(\vec{q}) = \mathrm{queue}(\vec{y})$. Also, $\max_i \mathrm{diff}_i(\vec{p}) = \max_i \mathrm{diff}_i(\vec{x}) + 1$, $\max_i \mathrm{diff}_i(\vec{q}) = \max_i \mathrm{diff}_i(\vec{y}) + 1$. It follows that $\mathrm{maxsum}(\vec{p}) = \mathrm{maxsum}(\vec{q})$, $\mathrm{queue}(\vec{p}) = \mathrm{queue}(\vec{q})$ and $\max_i \mathrm{diff}_i(\vec{p}) \leq \max_i \mathrm{diff}_i(\vec{q})$. By induction hypothesis, $J(\vec{p}) \leq J(\vec{q})$. Thus, $\theta_{11}(\vec{x}) \leq \theta_{11}(\vec{y})$. Let $\mathrm{queue}(\vec{x}) = B$ and $\mathrm{queue}(\vec{y}) = B$. From (19) and Lemma 4, $\theta_{11}(\vec{x}) \leq \beta J(\vec{x})$ and $\theta_{11}(\vec{y}) = \beta k$ where $k \in \{J(\vec{y}), J(\vec{q})\}$, where $\vec{q} = \vec{y} + \vec{e}_{11} - \vec{e}_{12}$. Note that $\mathrm{queue}(\vec{q}) = \mathrm{queue}(\vec{x}) = \mathrm{queue}(\vec{y})$. By induction hypothesis, $J(\vec{x}) \leq J(\vec{y})$. Now, $\mathrm{maxsum}(\vec{q}) \geq \mathrm{maxsum}(\vec{y})$ and $\max_i \mathrm{diff}_i(\vec{q}) = \max_i \mathrm{diff}_i(\vec{y}) + 1$. By induction hypothesis, $J(\vec{y}) \leq J(\vec{q})$. Thus, $J(\vec{x}) \leq J(\vec{q})$. It follows that $\theta_{11}(\vec{x}) \leq \theta_{11}(\vec{y})$.

Now let $(i, j) = (1, 2)$. Consider the case that $\mathrm{queue}(\vec{x}) < B$. It follows that $\mathrm{queue}(\vec{y}) < B$. From (18), $\theta_{12}(\vec{x}) = \beta J(\vec{x} + \vec{e}_{12})$ and $\theta_{11}(\vec{y}) = \beta J(\vec{y} + \vec{e}_{12})$. Let $\vec{p} = \vec{x} + \vec{e}_{12}$ and $\vec{q} = \vec{y} + \vec{e}_{12}$. Note that $\mathrm{maxsum}(\vec{p}) = \mathrm{maxsum}(\vec{x}) + 1$ and $\mathrm{maxsum}(\vec{q}) = \mathrm{maxsum}(\vec{y}) + 1$. Also, $\mathrm{queue}(\vec{p}) = \mathrm{queue}(\vec{x})$ and $\mathrm{queue}(\vec{q}) = \mathrm{queue}(\vec{y})$. It follows that $\mathrm{maxsum}(\vec{p}) = \mathrm{maxsum}(\vec{q})$, $\mathrm{queue}(\vec{p}) = \mathrm{queue}(\vec{q})$. Let $\max_i \mathrm{diff}_i(\vec{x}) > \min_i \mathrm{diff}_i(\vec{x})$, then $\max_i \mathrm{diff}_i(\vec{p}) = \max_i \mathrm{diff}_i(\vec{x})$, while $\max_i \mathrm{diff}_i(\vec{q}) \geq \max_i \mathrm{diff}_i(\vec{y})$. Thus $\max_i \mathrm{diff}_i(\vec{p}) \leq \max_i \mathrm{diff}_i(\vec{q})$. By induction hypothesis, $J(\vec{p}) \leq J(\vec{q})$. Thus, $\theta_{12}(\vec{x}) \leq \theta_{12}(\vec{y})$. Now let $\max_i \mathrm{diff}_i(\vec{x}) = \min_i \mathrm{diff}_i(\vec{x})$. It follows that $\max_i \mathrm{diff}_i(\vec{y}) > \max_i \mathrm{diff}_i(\vec{x})$. Now, $\max_i \mathrm{diff}_i(\vec{p}) \leq \max_i \mathrm{diff}_i(\vec{x}) + 1$, and $\max_i \mathrm{diff}_i(\vec{q}) \geq \max_i \mathrm{diff}_i(\vec{y})$. Thus $\max_i \mathrm{diff}_i(\vec{p}) \leq \max_i \mathrm{diff}_i(\vec{q})$. By induction hypothesis, $J(\vec{p}) \leq J(\vec{q})$. Thus, $\theta_{12}(\vec{x}) \leq \theta_{12}(\vec{y})$. Let

$\mathrm{queue}(\vec{x}) = B$ and $\mathrm{queue}(\vec{y}) = B$. From (19) and Lemma 4, $\theta_{12}(\vec{x}) \leq \beta J(\vec{x})$ and $\theta_{12}(\vec{y}) = \beta k$ where $k \in \{J(\vec{y}), J(\vec{q})\}$, where $\vec{q} = \vec{y} + \vec{e}_{12} - \vec{e}_{11}$. Note that $\mathrm{queue}(\vec{q}) = \mathrm{queue}(\vec{x}) = \mathrm{queue}(\vec{y})$. By induction hypothesis, $J(\vec{x}) \leq J(\vec{y})$. Note that $\mathrm{maxsum}(\vec{q}) \geq \mathrm{maxsum}(\vec{y})$. Thus, $\mathrm{maxsum}(\vec{q}) \geq \mathrm{maxsum}(\vec{x})$. First let $\max_i \mathrm{diff}_i(\vec{y}) > \max_i \mathrm{diff}_i(\vec{x})$. In this case, $\max_i \mathrm{diff}_i(\vec{q}) \geq \max_i \mathrm{diff}_i(\vec{y}) - 1 \geq \max_i \mathrm{diff}_i(\vec{x})$. By induction hypothesis, $J(\vec{x}) \leq J(\vec{q})$. Thus, $\theta_{12}(\vec{x}) \leq \theta_{12}(\vec{y})$. Now let $\max_i \mathrm{diff}_i(\vec{y}) = \max_i \mathrm{diff}_i(\vec{x})$. First assume $\max_i \mathrm{diff}_i(\vec{x}) \leq \min_i \mathrm{diff}_i(\vec{x}) + 1$. From (38), $\max_i \mathrm{diff}_i(\vec{y}) \leq \min_i \mathrm{diff}_i(\vec{y}) + 1$. It follows that $\max_i \mathrm{diff}_i(\vec{q}) \geq \max_i \mathrm{diff}_i(\vec{y}) \geq \max_i \mathrm{diff}_i(\vec{x})$. By induction hypothesis, $J(\vec{x}) \leq J(\vec{q})$. Thus, $\theta_{12}(\vec{x}) \leq \theta_{12}(\vec{y})$. Now let $\max_i \mathrm{diff}_i(\vec{x}) > \min_i \mathrm{diff}_i(\vec{x}) + 1 \geq 1$. It follows that $x_{11} > 0$. ¿From (19), $\theta_{12}(\vec{x}) = \beta \min(J(\vec{p}), J(\vec{x}))$ where $\vec{p} = \vec{x} + \vec{e}_{12} - \vec{e}_{11}$. Note that $\mathrm{queue}(\vec{p}) = \mathrm{queue}(\vec{q})$. Since $\max_i \mathrm{diff}_i(\vec{y}) \geq \max_i \mathrm{diff}_i(\vec{x}) \geq 2$, $\mathrm{maxsum}(\vec{q}) = \mathrm{maxsum}(\vec{y})$, and $\mathrm{maxsum}(\vec{p}) = \mathrm{maxsum}(\vec{x})$. Now $\max_i \mathrm{diff}_i(\vec{q}) \geq \max_i \mathrm{diff}_i(\vec{y}) - 1$. Since $\max_i \mathrm{diff}_i(\vec{x}) > \min_i \mathrm{diff}_i(\vec{x}) + 1$, $\max_i \mathrm{diff}_i(\vec{p}) = \max_i \mathrm{diff}_i(\vec{x}) - 1$. It follows that $\mathrm{maxsum}(\vec{p}) = \mathrm{maxsum}(\vec{q})$ and $\max_i \mathrm{diff}_i(\vec{p}) \geq \max_i \mathrm{diff}_i(\vec{q})$. By induction hypothesis, $J(\vec{p}) \leq J(\vec{q})$. Thus, $\theta_{12}(\vec{x}) \leq \theta_{12}(\vec{y})$.

Now let $(i,j) = (2,1)$. Consider the case that $\mathrm{queue}(\vec{x}) < B$. It follows that $\mathrm{queue}(\vec{y}) < B$. From (18), $\theta_{21}(\vec{x}) = \beta J(\vec{x} + \vec{e}_{21})$ and $\theta_{21}(\vec{y}) = \beta J(\vec{y} + \vec{e}_{21})$. Let $\vec{p} = \vec{x} + \vec{e}_{21}$ and $\vec{q} = \vec{y} + \vec{e}_{21}$. Clearly, $\mathrm{queue}(\vec{q}) = \mathrm{queue}(\vec{p})$. Let $\min_i \mathrm{diff}_i(\vec{x}) > 0$. It follows that $\mathrm{maxsum}(\vec{p}) = \mathrm{maxsum}(\vec{x})$ and $\mathrm{maxsum}(\vec{q}) \geq \mathrm{maxsum}(\vec{y})$. Thus $\mathrm{maxsum}(\vec{q}) \geq \mathrm{maxsum}(\vec{p})$. Note that $\max_i \mathrm{diff}_i(\vec{q}) \geq \max_i \mathrm{diff}_i(\vec{y})$ and $\max_i \mathrm{diff}_i(\vec{p}) = \max_i \mathrm{diff}_i(\vec{x})$. Thus $\max_i \mathrm{diff}_i(\vec{q}) \geq \max_i \mathrm{diff}_i(\vec{p})$. By induction hypothesis, $J(\vec{p}) \leq J(\vec{q})$. Thus, $\theta_{21}(\vec{x}) \leq \theta_{21}(\vec{y})$. Let $\min_i \mathrm{diff}_i(\vec{x}) = 0$. It follows that $\mathrm{maxsum}(\vec{x}) < \mathrm{maxsum}(\vec{y})$ and $\max_i \mathrm{diff}_i(\vec{y}) > 0$. Now $\mathrm{maxsum}(\vec{p}) = \mathrm{maxsum}(\vec{x}) + 1$ and $\mathrm{maxsum}(\vec{q}) \geq \mathrm{maxsum}(\vec{y})$. Thus $\mathrm{maxsum}(\vec{q}) \geq \mathrm{maxsum}(\vec{p})$. Now, $\max_i \mathrm{diff}_i(\vec{q}) \geq \max_i \mathrm{diff}_i(\vec{y}) > 0$. Note that $\max_i \mathrm{diff}_i(\vec{p}) \leq \max(\max_i \mathrm{diff}_i(\vec{x}), 1)$. Thus $\max_i \mathrm{diff}_i(\vec{q}) \geq \max_i \mathrm{diff}_i(\vec{p})$. By induction hypothesis, $J(\vec{p}) \leq J(\vec{q})$. Thus, $\theta_{21}(\vec{x}) \leq \theta_{21}(\vec{y})$. Now we consider the case that $\mathrm{queue}(\vec{x}) = B$ and $\mathrm{queue}(\vec{y}) = B$. This excludes $\vec{x} = \vec{0}$. Thus $x_{11} > 0$. From (19) and Lemma 4, $\theta_{21}(\vec{x}) \leq \beta J(\vec{p})$ and $\theta_{21}(\vec{y}) = \beta J(\vec{a})$ where $\vec{a} \in \{\vec{y}, \vec{q}, \vec{r}\}$, $\vec{p} = \vec{x} + \vec{e}_{21} - \vec{e}_{11}$, $\vec{q} = \vec{y} + \vec{e}_{21} - \vec{e}_{11}$ and $\vec{r} = \vec{y} + \vec{e}_{21} - \vec{e}_{12}$. Note that $\mathrm{queue}(\vec{p}) = \mathrm{queue}(\vec{q}) = \mathrm{queue}(\vec{r}) = \mathrm{queue}(\vec{y})$. Let $\min_i \mathrm{diff}_i(\vec{x}) > 0$. It follows that $\mathrm{maxsum}(\vec{p}) = \mathrm{maxsum}(\vec{x}) - 1$, $\mathrm{maxsum}(\vec{a}) \geq \mathrm{maxsum}(\vec{y}) - 1$. Thus $\mathrm{maxsum}(\vec{p}) \leq \mathrm{maxsum}(\vec{a})$. Note that $\max_i \mathrm{diff}_i(\vec{p}) = \max_i \mathrm{diff}_i(\vec{x}) - 1$, and $\max_i \mathrm{diff}_i(\vec{a}) \geq \max_i \mathrm{diff}_i(\vec{y}) - 1$. Thus $\max_i \mathrm{diff}_i(\vec{p}) \geq \max_i \mathrm{diff}_i(\vec{a})$. By induction hypothesis, $J(\vec{p}) \leq J(\vec{a})$. Thus, $\theta_{21}(\vec{x}) \leq \theta_{21}(\vec{y})$. Now let $\min_i \mathrm{diff}_i(\vec{x}) = 0$. It follows that that $\mathrm{maxsum}(\vec{y}) > \mathrm{maxsum}(\vec{x})$ and $\max_i \mathrm{diff}_i(\vec{y}) > 0$. We first consider the case that $\max_i \mathrm{diff}_i(\vec{x}) > 0$. The case $\max_i \mathrm{diff}_i(\vec{y}) = 1$ can be ruled out. To see this assume that $\max_i \mathrm{diff}_i(\vec{y}) = 1$. If $\min_i \mathrm{diff}_i(\vec{y}) = 0$ then $\vec{x} = \vec{y}$. If $\min_i \mathrm{diff}_i(\vec{y}) = 1$ then $\sum_i \mathrm{diff}_i(\vec{y}) = 2$ and $\sum_i \mathrm{diff}_i(\vec{x}) = 1$ ($\max_i \mathrm{diff}_i(\vec{x}) = 1$ in this case), but from (38) $\sum_i \mathrm{diff}_i(\vec{y}) - \sum_i \mathrm{diff}_i(\vec{x})$ must be even. Thus $\max_i \mathrm{diff}_i(\vec{y}) > 1$. Note that $\mathrm{maxsum}(\vec{p}) = \mathrm{maxsum}(\vec{x})$, $\mathrm{maxsum}(\vec{a}) \geq \mathrm{maxsum}(\vec{y}) - 1$. Since $\mathrm{maxsum}(\vec{y}) > \mathrm{maxsum}(\vec{x})$, $\mathrm{maxsum}(\vec{p}) \leq \mathrm{maxsum}(\vec{a})$. Now $\max_i \mathrm{diff}_i(\vec{p}) = \max(\max_i \mathrm{diff}_i(\vec{x}) - 1, 1)$ and $\max_i \mathrm{diff}_i(\vec{a}) \geq \max_i \mathrm{diff}_i(\vec{y}) - 1$. Thus $\max_i \mathrm{diff}_i(\vec{p}) \leq \max_i \mathrm{diff}_i(\vec{a})$ since $\max_i \mathrm{diff}_i(\vec{y}) > 1$. By induction hypothesis, $J(\vec{p}) \leq J(\vec{a})$. Thus, $\theta_{21}(\vec{x}) \leq \theta_{21}(\vec{y})$. Now let $\max_i \mathrm{diff}_i(\vec{x}) = 0$. From (19), $\theta_{12}(\vec{x}) \leq \beta J(\vec{x})$. Note that $\mathrm{maxsum}(\vec{a}) \geq \mathrm{maxsum}(\vec{y}) - 1$. Since $\mathrm{maxsum}(\vec{y}) > \mathrm{maxsum}(\vec{x})$, $\mathrm{maxsum}(\vec{x}) \leq \mathrm{maxsum}(\vec{a})$. Also $\max_i \mathrm{diff}_i(\vec{a}) \geq \max_i \mathrm{diff}_i(\vec{x}) = 0$. By induction hypothesis, $J(\vec{x}) \leq J(\vec{a})$. Thus, $\theta_{21}(\vec{x}) \leq \theta_{21}(\vec{y})$.

Finally let $(i,j) = (2,2)$. Consider the case that $\mathrm{queue}(\vec{x}) < B$. It follows that $\mathrm{queue}(\vec{y}) < B$. From (18), $\theta_{22}(\vec{x}) = \beta J(\vec{x} + \vec{e}_{22})$ and $\theta_{22}(\vec{y}) = \beta J(\vec{y} + \vec{e}_{22})$. Let $\vec{p} = \vec{x} + \vec{e}_{22}$ and $\vec{q} = \vec{y} + \vec{e}_{22}$. Clearly, $\mathrm{queue}(\vec{q}) = \mathrm{queue}(\vec{p})$. Let $\max_i \mathrm{diff}_i(\vec{x}) > \min_i \mathrm{diff}_i(\vec{x})$. Note that $\max_i \mathrm{diff}_i(\vec{q}) \geq \max_i \mathrm{diff}_i(\vec{y}) - 1$ and $\max_i \mathrm{diff}_i(\vec{p}) = \max_i \mathrm{diff}_i(\vec{x}) - 1$. Thus $\max_i \mathrm{diff}_i(\vec{q}) \geq \max_i \mathrm{diff}_i(\vec{p})$. Here $\max_i \mathrm{diff}_i(\vec{x}) > 0$. Hence $\mathrm{maxsum}(\vec{p}) = \mathrm{maxsum}(\vec{x})$ and $\mathrm{maxsum}(\vec{q}) \geq \mathrm{maxsum}(\vec{y})$. Thus $\mathrm{maxsum}(\vec{q}) \geq \mathrm{maxsum}(\vec{p})$. By induction hypothesis, $J(\vec{p}) \leq J(\vec{q})$. Thus, $\theta_{22}(\vec{x}) \leq \theta_{22}(\vec{y})$. Now let $\max_i \mathrm{diff}_i(\vec{x}) = \min_i \mathrm{diff}_i(\vec{x})$. It follows that $\max_i \mathrm{diff}_i(\vec{y}) > \max_i \mathrm{diff}_i(\vec{x})$. Let $\min_i \mathrm{diff}_i(\vec{x}) > 0$. Here, $\mathrm{maxsum}(\vec{p}) = \mathrm{maxsum}(\vec{x})$ and $\mathrm{maxsum}(\vec{q}) \geq \mathrm{maxsum}(\vec{y})$. Thus $\mathrm{maxsum}(\vec{q}) \geq \mathrm{maxsum}(\vec{p})$. Note that $\max_i \mathrm{diff}_i(\vec{q}) \geq \max_i \mathrm{diff}_i(\vec{y}) - 1$ and $\max_i \mathrm{diff}_i(\vec{p}) = \max_i \mathrm{diff}_i(\vec{x})$. Now $\max_i \mathrm{diff}_i(\vec{q}) \geq \max_i \mathrm{diff}_i(\vec{p})$ since $\max_i \mathrm{diff}_i(\vec{y}) > \max_i \mathrm{diff}_i(\vec{x})$. By induction hypothesis, $J(\vec{p}) \leq J(\vec{q})$. Thus, $\theta_{22}(\vec{x}) \leq \theta_{22}(\vec{y})$. Finally, consider the case $\max_i \mathrm{diff}_i(\vec{x}) = \min_i \mathrm{diff}_i(\vec{x}) = 0$. It follows that $\mathrm{maxsum}(\vec{x}) < \mathrm{maxsum}(\vec{y})$ and $\max_i \mathrm{diff}_i(\vec{y}) > 0$. The case $\max_i \mathrm{diff}_i(\vec{y}) = 1$ and $\min_i \mathrm{diff}_i(\vec{y}) = 0$ can be ruled out as then $\sum_i \mathrm{diff}_i(\vec{y}) - \sum_i \mathrm{diff}_i(\vec{x}) = 1$ which contradicts (38). Here, $\mathrm{maxsum}(\vec{p}) = \mathrm{maxsum}(\vec{x}) + 1$ and $\mathrm{maxsum}(\vec{q}) \geq \mathrm{maxsum}(\vec{y})$. Since $\mathrm{maxsum}(\vec{x}) < \mathrm{maxsum}(\vec{y})$, $\mathrm{maxsum}(\vec{q}) \geq \mathrm{maxsum}(\vec{p})$. Now $\max_i \mathrm{diff}_i(\vec{p}) = 1$, and $\max_i \mathrm{diff}_i(\vec{q}) \geq 1$, since $\max_i \mathrm{diff}_i(\vec{y}) > 0$ and $(\mathrm{diff}_1(\vec{y}), \mathrm{diff}_2(\vec{y})) \neq (1,0)$. By induction hypothesis, $J(\vec{p}) \leq J(\vec{q})$. Thus, $\theta_{22}(\vec{x}) \leq \theta_{22}(\vec{y})$. Now we consider the case that $\mathrm{queue}(\vec{x}) = B$ and $\mathrm{queue}(\vec{y}) = B$. This excludes $\vec{x} = \vec{0}$. Thus $x_{11} > 0$. From (19) and Lemma 4, $\theta_{22}(\vec{y}) = \beta J(\vec{a})$ where $\vec{a} \in \{\vec{y}, \vec{q}, \vec{r}\}$, $\vec{q} = \vec{y} + \vec{e}_{22} - \vec{e}_{11}$ and $\vec{r} = \vec{y} + \vec{e}_{22} - \vec{e}_{12}$. Let $\max_i \mathrm{diff}_i(\vec{x}) > 1 + \min_i \mathrm{diff}_i(\vec{x})$. From (19), $\theta_{22}(\vec{x}) \leq \beta J(\vec{p})$, where $\vec{p} = \vec{x} + \vec{e}_{22} - \vec{e}_{11}$. Note that $\mathrm{queue}(\vec{p}) = \mathrm{queue}(\vec{q}) = \mathrm{queue}(\vec{r}) = \mathrm{queue}(\vec{y})$. It follows that $\mathrm{maxsum}(\vec{p}) = \mathrm{maxsum}(\vec{x}) - 1$, $\mathrm{maxsum}(\vec{a}) \geq \mathrm{maxsum}(\vec{y}) - 1$. Thus $\mathrm{maxsum}(\vec{p}) \leq \mathrm{maxsum}(\vec{a})$. Note that $\max_i \mathrm{diff}_i(\vec{p}) = \max_i \mathrm{diff}_i(\vec{x}) - 2$, and $\max_i \mathrm{diff}_i(\vec{a}) \geq \max_i \mathrm{diff}_i(\vec{y}) - 2$. Thus $\max_i \mathrm{diff}_i(\vec{p}) \geq \max_i \mathrm{diff}_i(\vec{a})$. By induction hypothesis, $J(\vec{p}) \leq J(\vec{a})$. Thus, $\theta_{22}(\vec{x}) \leq \theta_{22}(\vec{y})$. Now let $\max_i \mathrm{diff}_i(\vec{x}) \leq 1 +$

$\min_i \operatorname{diff}_i(\vec{x})$. From (38) we can assume $\max_i \operatorname{diff}_i(\vec{y}) > \max_i \operatorname{diff}_i(\vec{x})$ without any loss of generality. First assume that $\min_i \operatorname{diff}_i(\vec{x}) > 0$. Thus $x_{12} > 0$. From (19), $\theta_{22}(\vec{x}) \leq \beta J(\vec{t})$, where $\vec{t} = \vec{x} + \vec{e}_{22} - \vec{e}_{12}$. Again $\operatorname{queue}(\vec{t}) = \operatorname{queue}(\vec{q}) = \operatorname{queue}(\vec{r}) = \operatorname{queue}(\vec{y})$. It follows that $\operatorname{maxsum}(\vec{t}) = \operatorname{maxsum}(\vec{x}) - 1$, $\operatorname{maxsum}(\vec{a}) \geq \operatorname{maxsum}(\vec{y}) - 1$. Thus $\operatorname{maxsum}(\vec{t}) \leq \operatorname{maxsum}(\vec{a})$. Note that $\max_i \operatorname{diff}_i(\vec{t}) = \max_i \operatorname{diff}_i(\vec{x}) - 1$, and $\max_i \operatorname{diff}_i(\vec{a}) \geq \max_i \operatorname{diff}_i(\vec{y}) - 2$. Thus $\max_i \operatorname{diff}_i(\vec{t}) \geq \max_i \operatorname{diff}_i(\vec{a})$ since $\max_i \operatorname{diff}_i(\vec{y}) > \max_i \operatorname{diff}_i(\vec{x})$. By induction hypothesis, $J(\vec{t}) \leq J(\vec{a})$. Thus, $\theta_{22}(\vec{x}) \leq \theta_{22}(\vec{y})$. Now consider $\min_i \operatorname{diff}_i(\vec{x}) = 0$. From (19), $\theta_{22}(\vec{x}) \leq \beta J(\vec{x})$. Now, $\operatorname{queue}(\vec{x}) = \operatorname{queue}(\vec{q}) = \operatorname{queue}(\vec{r}) = \operatorname{queue}(\vec{y})$. Here, $\max_i \operatorname{diff}_i(\vec{x}) \in \{0, 1\}$. First let $\max_i \operatorname{diff}_i(\vec{x}) = 0$. It follows that $\operatorname{maxsum}(\vec{y}) > \operatorname{maxsum}(\vec{x})$. Now using (38) $\sum_i \operatorname{diff}_i(\vec{y}) \geq 2$. Now, $\operatorname{maxsum}(\vec{a}) \geq \operatorname{maxsum}(\vec{y}) - 1 \geq \operatorname{maxsum}(\vec{x})$. Note that $\max_i \operatorname{diff}_i(\vec{a}) \geq 0 = \max_i \operatorname{diff}_i(\vec{x})$. By induction hypothesis, $J(\vec{x}) \leq J(\vec{a})$. Thus, $\theta_{22}(\vec{x}) \leq \theta_{22}(\vec{y})$. Now let $\max_i \operatorname{diff}_i(\vec{x}) = 1$. It follows that $\operatorname{maxsum}(\vec{y}) > \operatorname{maxsum}(\vec{x})$. Now using (38) $\sum_i \operatorname{diff}_i(\vec{y}) \geq 3$. Thus $\max_i \operatorname{diff}_i(\vec{y}) \geq 2$. Now, $\operatorname{maxsum}(\vec{a}) \geq \operatorname{maxsum}(\vec{y}) - 1 \geq \operatorname{maxsum}(\vec{x})$. Note that $\max_i \operatorname{diff}_i(\vec{a}) \geq 1 = \max_i \operatorname{diff}_i(\vec{x})$. The first inequality follows since either $\max_i \operatorname{diff}_i(\vec{y}) = 2, \min_i \operatorname{diff}_i(\vec{y}) = 1$ or $\max_i \operatorname{diff}_i(\vec{y}) \geq 3$. By induction hypothesis, $J(\vec{x}) \leq J(\vec{a})$. Thus, $\theta_{22}(\vec{x}) \leq \theta_{22}(\vec{y})$. Thus

$$\theta_{ij}(\vec{x}) \leq \theta_{ij}(\vec{y}) \tag{40}$$

The result follows from (39) and (40).

Lemma 6 *Let* $\vec{x} = (i-1, j, k, l-1)$ *and* $\vec{y} = (i, j-1, k-1, l)$, *where* $\min(i, j, k, l) > 0$. *Then* $J(\vec{x}) = J(\vec{y})$.

Since Lemma 6 holds for all $\beta < 1$, from (6), the following result holds for equal arrival rates. *When both pairs can be scheduled, then the optimal strategy can choose arbitrarily between them.* The intuition behind the result is that in this case the difference functions and the queue occupancies are not affected by the selection of the pairs, and thus this choice does not the affect job loss rate. Analytically, the result follows from an application of Lemma 5.

Proof: Note that $\operatorname{queue}(\vec{x}) = \operatorname{queue}(\vec{y}) = i+j+k+l-2$, $\operatorname{maxsum}(\vec{x}) = \operatorname{maxsum}(\vec{y}) = \max(i, l) + \max(j, k) - 1$. Also, $\operatorname{diff}_1(\vec{x}) = \operatorname{diff}_1(\vec{y}) = |i - l|$ and $\operatorname{diff}_2(\vec{x}) = \operatorname{diff}_2(\vec{y}) = |j - k|$. Thus $\max_i \operatorname{diff}_i(\vec{x}) = \max_i \operatorname{diff}_i(\vec{y})$. From Lemma 5 $J(\vec{x}) \leq J(\vec{y})$ and $J(\vec{y}) \leq J(\vec{x})$. The result follows.

Theorem 1 follows from Lemmas 2 to 6.

B Proof of the properties of the optimal strategy for $N = 2$ and unequal arrival rates

Lemmas 2 and 3 hold for arbitrary arrival rates λ_{ij}. The properties mentioned for the optimal policy in Section 3.2 follow from these lemmas.

C Proof of the heavy traffic lower bound for the optimal discounted cost function for arbitrary N and arrival rates (Lemma 1)

We prove Lemma 1 in this section. As in Section A we show that the heavy traffic lower bound holds for $J_{\beta,k}(\vec{x})$ in iteration (8) for all k, and β sufficiently close to 1 using induction, if initial value of the iterates is $J_{\beta,0}(\vec{x}) = \operatorname{cong}(\vec{x})$. Since initial choice does not affect the limiting value ($k \to \infty$) of these functions, it follows that the heavy traffic lower bound holds for $J_{\beta,\text{OPT}}(\vec{x})$ for all β sufficiently close to 1. For simplicity we will drop the index β and k henceforth.

Recall that $c(\vec{x}, \vec{y}, i, j) = 1 + \sum_{i=1}^{N} \sum_{j=1}^{N} (x_{ij} - y_{ij})$. Let $\operatorname{cong}_{\text{inp}}(\vec{x})$ be the maximum number of jobs waiting at any input, i.e., $\operatorname{cong}_{\text{inp}}(\vec{x}) = \max_i \sum_{j=1}^{N} x_{ij}$. Similarly, $\operatorname{cong}_{\text{op}}(\vec{x})$ is the maximum number of jobs waiting for any output, i.e., $\operatorname{cong}_{\text{op}}(\vec{x}) = \max_j \sum_{i=1}^{N} x_{ij}$. Clearly, $\operatorname{cong}(\vec{x}) = \max\left(\operatorname{cong}_{\text{inp}}(\vec{x}), \operatorname{cong}_{\text{op}}(\vec{x})\right)$.

Proof: Let $\beta_0 = \frac{B}{B + \min(\min_i \sum_{j=1}^{N} \lambda_{ij}, \min_j \sum_{i=1}^{N} \lambda_{ij}) - \mu}$. We prove in two parts. First we show that $J(\vec{x}) \geq \operatorname{cong}_{\text{inp}}(\vec{x})$ for all $\beta \in [\beta_0, 1)$, and all states $\vec{x}$. Next we show that $J(\vec{x}) \geq \operatorname{cong}_{\text{op}}(\vec{x})$ for all $\beta \in [\beta_0, 1)$ and all states $\vec{x}$. The results hold trivially if $J(\vec{x}) = \operatorname{cong}(\vec{x})$ for all $\vec{x}$. We just show the induction cases.

We assume that $J(\vec{x}) \geq \operatorname{cong}_{\text{inp}}(\vec{x})$ for all β sufficiently close to 1 ($\beta < 1$)and all states $\vec{x}$ and show the property for $TJ(\vec{x})$. For every state $\vec{y}$ in $S(\vec{x})$, $\operatorname{cong}_{\text{inp}}(\vec{y}) \geq \operatorname{cong}_{\text{inp}}(\vec{x}) - 1$. This happens because the scheduling can remove at most one job from any input. An input i is most congested if it has the maximum number of jobs waiting for transmission, i.e., $i = \arg\max_l \sum_{j=1}^{N} x_{lj}$. We assume that the most congested input is unique (the extension to the case when multiple inputs have the maximum congestion is straight-forward). For every state $\vec{y}$ in $\cup_{j=1}^{N} A(\vec{x}, i, j)$, $\operatorname{cong}_{\text{inp}}(\vec{y}) + c(\vec{x}, \vec{y}, i, j) \geq \operatorname{cong}_{\text{inp}}(\vec{x}) + 1$. Either the new job is accepted or it is dropped. In the former case, $\operatorname{cong}_{\text{inp}}(\vec{y}) = \operatorname{cong}_{\text{inp}}(\vec{x}) + 1$, $c(\vec{x}, \vec{y}, i, j) \geq 0$. In the latter case, $c(\vec{x}, \vec{y}, i, j) \geq 1$ and $\operatorname{cong}_{\text{inp}}(\vec{y}) = \operatorname{cong}_{\text{inp}}(\vec{x})$. For any other state $\vec{y}$ in $A(\vec{x}, l, j)$, $i \neq l$, $\operatorname{cong}_{\text{inp}}(\vec{y}) + c(\vec{x}, \vec{y}, i, j) \geq \operatorname{cong}_{\text{inp}}(\vec{x})$. This happens because the arrival at one input does not alter the buffer occupancy at a different input. Thus congestion at input i does not change. Hence, $\operatorname{cong}_{\text{inp}}(\vec{y}) \geq \operatorname{cong}_{\text{inp}}(\vec{x})$ and $c(\vec{x}, \vec{y}, i, j) \geq 0$.

$$
\begin{aligned}
& TJ(\vec{x}) \\
= \ & \sum_{k=1}^{N}\sum_{j=1}^{N} \min_{\vec{y}\in A(\vec{x},k,j)} \lambda_{kj}\left(c(\vec{x},\vec{y},k,j)+\beta J(\vec{y})\right) \\
& +\beta\mu \min_{\vec{y}\in S(\vec{x})} J(\vec{y}) \\
\geq \ & \beta\sum_{k=1}^{N}\sum_{j=1}^{N} \min_{\vec{y}\in A(\vec{x},k,j)} \lambda_{kj}\left(c(\vec{x},\vec{y},k,j)+\mathrm{cong}_{\mathrm{inp}}(\vec{y})\right) \\
& +\beta\mu \min_{\vec{y}\in S(\vec{x})} \mathrm{cong}_{\mathrm{inp}}(\vec{y})) \text{ (by induction hypothesis)} \\
\geq \ & \beta\sum_{j=1}^{N} \min_{\vec{y}\in A(\vec{x},i,j)} \lambda_{ij}\left(c(\vec{x},\vec{y},i,j)+\mathrm{cong}_{\mathrm{inp}}(\vec{y})\right) \\
& +\beta\sum_{\substack{1\leq k\leq N\\ k\neq i}}\sum_{j=1}^{N} \min_{\vec{y}\in A(\vec{x},k,j)} \lambda_{kj}\left(c(\vec{x},\vec{y},k,j)\right. \\
& \left.+\mathrm{cong}_{\mathrm{inp}}(\vec{y})\right)+\beta\mu\left(\mathrm{cong}_{\mathrm{inp}}(\vec{x})-1\right) \\
\geq \ & \beta\sum_{j=1}^{N}\lambda_{ij}\left(1+\mathrm{cong}_{\mathrm{inp}}(\vec{x})\right) \\
& +\beta\sum_{\substack{1\leq k\leq N\\ k\neq i}}\sum_{j=1}^{N}\lambda_{kj}\mathrm{cong}_{\mathrm{inp}}(\vec{x}) \\
& +\beta\mu\left(\mathrm{cong}_{\mathrm{inp}}(\vec{x})-1\right) \\
= \ & \beta\mathrm{cong}_{\mathrm{inp}}(\vec{x})+\beta(\sum_{j=1}^{N}\lambda_{ij}-\mu) \\
& \qquad \text{(since } \mu+\sum_{i=1}^{N}\sum_{j=1}^{N}\lambda_{ij}=1) \\
\geq \ & \mathrm{cong}_{\mathrm{inp}}(\vec{x})
\end{aligned}
$$

The last inequality follows since $\sum_{j=1}^{N}\lambda_{ij} > \mu \forall i$, $\mathrm{cong}(\vec{x}) \leq B, \forall\vec{x}$, and $\beta \geq \beta_0$.

Now we prove the second part, i.e., $J(\vec{x}) \geq \mathrm{cong}_{\mathrm{op}}(\vec{x})$ for all $\beta \in [\beta_0, 1)$ and all states $\vec{x}$. We assume that $J(\vec{x}) \geq \mathrm{cong}_{\mathrm{op}}(\vec{x})$ for all $\beta \in [\beta_0, 1)$ and all states $\vec{x}$, and show the property for $TJ(\vec{x})$. We assume this lower bound for $J(\vec{x})$ and show the property for $TJ(\vec{x})$. As in the first part, for every state $\vec{y}$ in $S(\vec{x})$, $\mathrm{cong}_{\mathrm{op}}(\vec{y}) \geq \mathrm{cong}_{\mathrm{op}}(\vec{x}) - 1$. The most congested output is the one for which the maximum number of jobs wait, i.e., $j = \arg\max_k \sum_{i=1}^{N} x_{ik}$. For every state $\vec{y}$ in $\cup_{i=1}^{N} A(\vec{x}, i, j)$ $\mathrm{cong}_{\mathrm{op}}(\vec{y}) + c(\vec{x},\vec{y},i,j) \geq \mathrm{cong}_{\mathrm{op}}(\vec{x}) + 1$. This happens because either the new job is accepted or it is rejected. In the former case $\mathrm{cong}_{\mathrm{op}}(\vec{y}) = \mathrm{cong}_{\mathrm{op}}(\vec{x})+1$, $c(\vec{x},\vec{y},i,j) \geq 0$ and in the latter case $\mathrm{cong}_{\mathrm{op}}(\vec{y}) = \mathrm{cong}_{\mathrm{op}}(\vec{x})$, $c(\vec{x},\vec{y},i,j) \geq 1$. For any i, and every state $\vec{y}$ in $A(\vec{x}, i, k)$, $k \neq j$, $\mathrm{cong}_{\mathrm{op}}(\vec{y}) + c(\vec{x},\vec{y},i,j) \geq \mathrm{cong}_{\mathrm{op}}(\vec{x})$. There are two possibilities: (a) a job waiting for output j is dropped, then $\mathrm{cong}_{\mathrm{op}}(\vec{y}) \geq \mathrm{cong}_{\mathrm{op}}(\vec{x}) - 1$, $c(\vec{x},\vec{y},i,j) \geq 1$ (b)a job waiting for output j is not dropped, then $\mathrm{cong}_{\mathrm{op}}(\vec{y}) \geq \mathrm{cong}_{\mathrm{op}}(\vec{x})$, $c(\vec{x},\vec{y},i,j) \geq 0$. Thus $\mathrm{cong}_{\mathrm{op}}(\vec{y}) + c(\vec{x},\vec{y},i,j) \geq \mathrm{cong}_{\mathrm{op}}(\vec{x})$ in either case. The relations are similar to the previous case, and the inequality follows likewise.

Lemma 2 holds for arbitrary N and arbitrary arrival rates as well. The proof is similar to that for the $N = 2$ case as described in appendix A and is omitted for brevity. Since Lemma 2 holds for all $\beta < 1$, (6) shows that the optimal memory management decision is to accept new jobs without any pushouts as long as the input buffer has space.