Computer Science Department Technical Report
University of California
Los Angeles, CA 90024-1596

OPTIMAL ROUTING FOR CLOSED QUEUEING NETWORKS

William C. Cheng
Richard R. Muntz

September 1989
CSD-890054

Optimal Routing for Closed Queueing Networks[1]

William C. Cheng Richard R. Muntz

UCLA Computer Science Department

September 1989
Revised, December 1989

[1]This work was partially supported by a MICRO grant from the University of California and the Hughes Aircraft Company.

Abstract

Optimal routing is an important subclass of resource allocation and load balancing problems that has applications in file allocation, distributed database systems, local area networks, etc. In this paper, we present a generalization of the vertex allocation theorem of Tripathi and Woodside. The vertex allocation theorem applies to routing of single customer chains in closed product form networks. Basically, the theorem states that to maximize the sum of throughputs, each customer should consistently use the same server for each request type rather than probabilistically choose among alternatives for each request. We show that vertex allocation is valid for any network of quasi-reversible queues and that the objective function can be any Markov reward function on the state space of the queueing network model.

In many optimization problems the customers are not all known a priori but rather enter the system dynamically. We address the problem of optimally choosing the routing for a new customer being added to an operational system. We model the system as a closed multi-chain queueing network and show that a simple "greedy" algorithm can be used to determine the optimal routing for a restricted type of routing behavior. A purely analytic application of the greedy algorithm requires solution of several queueing models. A less costly approximation is proposed which uses measurements of the operational system to avoid expensive calculations.

1 Introduction

Queueing network models have been widely used to model distributed computer system and communication networks. We will be concerned in this paper with system optimization problems that can be expressed in terms of a closed queueing network model of the system. In order to optimize some performance measure, it is assumed that some model parameters can be controlled, subject to specified constraints. Some of the most common model parameters that are taken as control variables are: routing of customers; service capacities of servers; and scheduling disciplines. Constraints are often expressed in terms of total cost of the system, maximum response time, minimum throughput, fairness criteria, etc. We will be concerned with the optimization of systems in which the decision variables control only the routing of customers. This covers a wide variety of problems since routing is a way of determining which resources are to be used to provide the services required by a customer, and many problems fit in this framework, e.g., file allocation, load balancing, packet routing, etc. We allow, as objective functions, any *reward function* on the states of the model. This is general enough to include the most common objective functions, e.g., the weighted sum of throughputs of all customers.

There has been considerable research on the use of queueing network models for system optimization [7] [8] [10] [15] [20] [22] [23] [25] [26]. Most of these studies deal with systems with workloads that are modeled as open customer chains with Poisson arrivals. For other applications closed chains are more accurate. Closed chains are used to represent a customer population which is small enough that congestion in the system has an appreciable effect on the rate of new demands for service. In this paper, we will assume that the system of interest can be reasonably modeled as a closed product form queueing network. Closed queueing networks have been used to model virtual communication channels with window flow control, LANs with a moderate number of workstations, a mix of multiprogrammed processes, etc. There are considerably fewer results available on the optimization of closed queueing networks due to the more complex steady state solution for closed models which have a finite state space and require normalization. We are concerned in this paper mostly with static routing or load balancing, in which routing decisions are fixed at the time a customer is introduced into the system. Below we briefly review the work most closely related to ours and indicate the distinguishing characteristics of our results.

Kobayashi and Gerla [14] considered closed queueing networks and the problem of selecting routing to optimize a weighted sum of throughputs. They showed that for a single chain closed network the throughput is a convex function of the routing decisions and that a modified form of the Flow Deviation algorithm [12] could be used to determine the optimal routing. For a multi-chain network the throughput is not a convex function of the routing variables and heuristics must be employed. For the class of networks treated in [14] there are multiple customers per chain and the transition probabilities are restricted to be the same for all customers in the same chain.

In a recent paper [21], Tripathi and Woodside considered the case in which the routing decisions for each individual customer can be independently chosen. From another point of view this is equivalent to letting each customer be represented by a separate (single customer) chain. Under certain restrictions (which we will discuss shortly) they proved the so called *vertex allocation* theorem. Consider a queueing network with multiple chains, each with a single customer, in which a customer can be in different "phases". Customer phases are a useful device for modeling the requests for certain services by a job. For each type of request (a phase the customer can be in), there may be several candidate servers that can provide the service the customer requests. The problem is to assign routing probabilities to these candidate servers so as to optimize some specified performance measures. In general, if there are several servers that can provide the required function then each time a job may choose to use a server according to assigned probabilities. In fact, in models in which there is more than one customer per chain, an optimal choice for the transition probabilities (for some objective function) will involve such a "probabilistic routing". However, when each customer can be independently routed, it is shown in [21] that there exists a *vertex allocation* that gives optimal throughput, i.e. in which each phase of service (request type) is executed at one particular server each time the phase is entered. In other words, the transition probabilities describing which server is to be used are assigned values of 0 or 1. Thus vertex allocation implies a deterministic choice of a server each time the phase is entered.

The proof of the vertex allocation theorem in [21] is algebraic in nature. Although it covers many of the most common applications, a more general result is presented in this paper. The result in [21] is based on an algebraic form for the solution of the queueing network model. The algebraic form assumed in the paper is only valid for a subset of

product form queueing networks. For a network of quasi-reversible queues, the algebraic form used in [21] may not be satisfied; however, the steady-state probability distribution still has a product form. The MSCCC center in [2] is an example of a center which can be included in a product form network but which does not have the algebraic form assumed in [21]. In addition, the only performance measure considered in [21] is a weighted sum of throughputs. There are other performance measures of interest that can be expressed as Markov reward functions on the state space of the queueing network model. For example, in [9] a product form queueing network is used to analyze system availability. One performance measure of interest is the steady state probability that the system is operational. This can be expressed as a reward function in which the operational states have reward rate 1 and the failed states have a reward of 0.

In the first part of this paper, we will show that the vertex allocation theorem holds for any network of quasi-reversible queues [13] and for *any* Markov reward function. This result is clearly a direct extension of the result in [21]. The major claim to originality for this result is that it applies to a more general class of queueing networks and to a wider class of objective functions. We also believe the proof method is of some interest in itself. Also, the approach taken here has the potential to provide a basis for heuristic optimization since the development has a probabilistic interpretation.

In the preceding discussion it is assumed that all jobs which are to be routed are known a priori. In many optimization problems the customers are not all known a priori but rather enter the system dynamically. For example, consider a model of a LAN in which disk-less workstations are added one at a time. When a workstation is added, decisions have to be made such as where to obtain the kernel and swap space, where to locate files, etc. A similar situation exists for choosing display servers for high speed graphics display terminals. We continue to assume that the system can be modeled as a closed product form queueing network. We will show that under a restricted, yet quite general, type of routing (which we call the "phase-type" routing), any Markov reward function is a quasi-linear function of the routing decisions variables, i.e. has only one local maximum and one local minimum. Furthermore, since the objective function is quasi-linear and the optimal solution lies at a vertex of the search space, we will show that a simple "greedy" algorithm can be used to determine the optimal routing. If the resource demands on all queues for all chains are known exactly, the greedy algorithm can give the exact optimal

routing; however, it can be computationally expensive. Furthermore, resource demands of individual chains in the operational system may not be known exactly. An approximation based on the formal development is proposed in Section 5; it is less costly and requires less detailed information. The information required is easily obtainable from measurements of the operational system and estimates of service demands of the new customer.

In Section 2 we formally define the model and the notation used in the remainder of the paper. In Section 3 we present our generalization of the vertex allocation theorem and outline the proof. Section 4 considers the case of adding a new customer to an existing system and proves that a greedy algorithm can be used to determine the optimal routing. In Section 5 we develop an approximation for the quantities used in the greedy algorithm from measurements of the operational system. In Section 6 we present a numerical example and Section 7 concludes with a summary and discussion.

2 Queueing Network Model

For convenience we list below the notation which is used in the remainder of the paper.

J	=	number of service centers.
K	=	total number of chains in the network.
$j(k)$	=	a specified (reference) service center visited by chain k.
θ_{jk}	=	visit ratio of a chain k customer at center j, scaled such that $\theta_{j(k)k} = 1$.
T_{jk}	=	mean service time of a chain k customer at center j.
$\vec{x}_j$	=	$(x_{j1}, x_{j2}, \ldots, x_{jK})$ = state of center j where x_{jk} indicates the state of the chain k customer with respect to center j. Specifically, $x_{jk} = 0$ if the chain k customer is not at center j and $x_{jk} = c$ if the chain k customer is at center j in class c.
$\underline{\vec{x}}$	=	$(\vec{x}_1, \vec{x}_2, \ldots, \vec{x}_J)$ = state of the network model.
λ_{jk}	=	$\theta_{jk} \cdot \lambda_k$ = throughput of chain k customers at center j, where $\lambda_k = \lambda_{j(k)k}$.
L_{jk}	=	mean number of chain k customers at center j.
W_{jk}	=	mean waiting time (queueing time + service time) of a chain k customer at center j.
$\pi(\underline{\vec{x}})$	=	steady state probability that the network is in state $\underline{\vec{x}} \in \underline{S}$, where $\underline{S}$ is the state space of the network.

$\vec{e}_k$ = K-dimensional vector whose k^{th} element is one and whose other elements are zeroes.

The queueing network models being considered are multiple chain networks of quasi-reversible queues [13] (or equivalently, queues that satisfy the $M \Rightarrow M$ condition [16]) with Markovian routing for customers. There is only one customer per chain unless otherwise stated.

The major difference from standard queueing network models concerns the routing decision variables that are introduced. Below, we introduce a type of model that incorporates general class of routing decisions. (In Section 4, we will introduce another type of routing which is more restrictive but still quite general and which permits efficient solution to important optimization problems.)

Consider a single customer chain k, in a network of quasi-reversible queues. Let $\mathcal{C}_k$ be the set of classes that customer k can take on. For convenience we will assume that: (a) the classes associated with different customers are distinct and, (b) a customer never visits two distinct centers in the same class. These assumptions are for convenience in exposition and can always be accommodated simply by renaming the classes. Let p_{c_1,c_2} denote the probability that a customer in class c_1 next moves to class c_2. (Due to the use of disjoint class names, the chain, the source, and the destination service center are implied by the subscripts.) The transition behavior of a customer k is then described by a matrix $P_k = [\![p_{c_i,c_j}]\!]$, c_i and $c_j \in \mathcal{C}_k$. Since P is a stochastic matrix the rows must sum to 1, and therefore, for each chain k:

$$\sum_{c_j \in \mathcal{C}_k} p_{c_i,c_j} = 1, \quad \forall\, c_i \in \mathcal{C}_k \tag{1}$$

In our case, since we are interested in optimal routing, not all the elements in P_k are fixed. In fact, we allow element $p_{c_i,c_j} = a_{i,j} + v_{i,j}$ where $a_{i,j}$ is a non-negative constant and $v_{i,j}$ is a non-negative variable. When $a_{i,j} > 0$, this places a lower bound on the transition probability p_{c_i,c_j}. Clearly, $v_{i,j}$ is be a decision variable in our optimization problems.

The variables $v_{i,j}$ are all distinct and independent, except for the constraints embodied in Eq. (1). The fact that some transition probabilities can be zero or non-zero, as a function of the decision variables, means that the Markov chain associated with a customer's class

transitions can be decomposable for some values of the decision variables and can also contain "transient classes". To deal with this issue we assume that for each customer there is a *distinguished class* and, for a well formed model, for any feasible choice of values for the routing variables, the set of classes that are reachable from the distinguished class forms an irreducible Markov chain. For any assignment of values to the routing variables, the queueing network model has customers whose transition behavior is described by the associated irreducible Markov chain. This rather cumbersome definition is required to avoid pathological cases. In practice, it is not a problem to accommodate the definition and in fact, the requirement is naturally satisfied.

In the next section we describe a proof for the vertex allocation theorem for queueing networks with the type of routing described above.

3 General Proof of the Vertex Allocation Theorem

In this section we describe our proof for the vertex allocation theorem. For convenience in this discussion, we assume that the lower bounds on transition rates (i.e., the $a_{i,j}$'s in the previous section) are all zero. A more complete and more detailed proof is available in [3].

We proceed by showing that if all but one of the routing variables are fixed, then an optimal solution for the remaining decision exists with values for the transition probabilities set to 0 or 1. Once this is proved the vertex allocation theorem follows easily by contradiction as follows. Suppose there is no optimal vertex allocation. Then start with an optimal solution and select one decision variable that is neither 0 nor 1. Considering the other variables to be fixed and applying the previous result, an optimal solution exists with the selected decision variable set to 0 or 1. Proceeding through the decision variables in this way, they can all be "forced" to a vertex.

Consider a queueing network model as described in Section 2. Let $R(\underline{\vec{x}})$ denote the reward rate assigned to state $\underline{\vec{x}}$, and let $R = \sum_{\underline{\vec{x}} \in \underline{S}} \pi(\underline{\vec{x}}) R(\underline{\vec{x}})$ be the expected reward rate. Consider a particular customer (chain) k and let $\mathcal{C}_k = \{1, 2, \ldots, C^k\}$. We can partition the state space $\underline{S}$ according to which class the chain k customer is in. Let $A_c^k = \{\underline{\vec{x}} | x_{jk} = c\}$ be the set of states in which chain k customer is in class c for $1 \leq c \leq C^k$ and where the center that corresponds to c is j. (Recall that in the model description, a customer class c uniquely

identifies the chain and the location of the customer). For simplicity in notation, we will drop the superscript k in A_c^k and C^k where the chain is understood from the context. It is clear that the sets $A_1, \ldots, A_C$ form a partition of $\underline{S}$. Let $\mathbf{Q}$ be the transition rate matrix of the model and partition the state space according to $A_1, \ldots, A_C$. By ordering the states, $\mathbf{Q}$ can be put in the form shown in Figure 1. A principal submatrix Q_{cc} corresponds to

$$\begin{bmatrix} Q_{11} & Q_{12} & Q_{13} & \cdots & Q_{1C} \\ Q_{21} & Q_{22} & Q_{23} & \cdots & Q_{2C} \\ & \ddots & & & \\ Q_{C1} & Q_{C2} & & & Q_{CC} \end{bmatrix}$$

Figure 1: Rate matrix.

the set of states A_c where the chain k customer is in class c, and transitions between states in A_c correspond to transitions which do not involve the chain k customer. A submatrix Q_{cd}, for $c \neq d$, corresponds to transitions in which the chain k customer moves from class c to class d.

Let $\pi(\underline{\vec{x}}|A_c) = \pi(\underline{\vec{x}}) / \sum_{\underline{\vec{x}}' \in A_c} \pi(\underline{\vec{x}}')$ be the steady state probability of state $\underline{\vec{x}}$ conditioned on being in some state of A_c. Let $w_c = \sum_{\underline{\vec{x}}' \in A_c} \pi(\underline{\vec{x}}')$ be the probability that the model is in some state in partition A_c. We can express the steady state reward rate as follows,

$$\begin{aligned} R &= \sum_{\underline{\vec{x}} \in S} \pi(\underline{\vec{x}}) R(\underline{\vec{x}}) \\ &= w_1 \sum_{\underline{\vec{x}} \in A_1} \pi(\underline{\vec{x}}|A_1) R(\underline{\vec{x}}) + \ldots + w_C \sum_{\underline{\vec{x}} \in A_C} \pi(\underline{\vec{x}}|A_C) R(\underline{\vec{x}}) \end{aligned} \tag{2}$$

In the development which follows, we will show that for a network of quasi-reversible queues, for all classes c and all chains k, the conditional state probability $\pi(\underline{\vec{x}}|A_c)$ is independent of the visit ratios for class c, and since routing decisions only affect the visit ratios for class c, $\pi(\underline{\vec{x}}|A_c)$ is independent of the routing decisions.

Consider a single quasi-reversible queue [13] driven by Poisson arrivals with rates $\lambda_1, \ldots, \lambda_{C'}$ (open model) where C' is the total number of classes. Let $\vec{x}$ denote a state of the queue and $\vec{n}(\vec{x}) = (n_1(\vec{x}), \ldots, n_{C'}(\vec{x}))$ be the population vector where $n_c(\vec{x})$ is the number of class c customers in the queue or in service in state $\vec{x}$. Let $\pi(\vec{x})$ denote the equilibrium state

probability of state $\vec{x}$, S denote the state space of the queue, and $Q = [\![q(\vec{x}, \vec{x}')]\!]$ denote the transition rate matrix of the corresponding Markov process where $q(\vec{x}, \vec{x}')$ is the transition rate from state $\vec{x}$ to $\vec{x}'$. Consider a subset $A_c(n) \subset S$ where $A_c(n) = \{\vec{x} | n_c(\vec{x}) = n\}$ is the set of states with n class c customers. Let

$$\pi(\vec{x}|A_c(n)) = \frac{\pi(\vec{x})}{\sum_{\vec{x}' \in A_c(n)} \pi(\vec{x}')} \tag{3}$$

be the state probability of state $\vec{x}$ conditioned on being in some state of $A_c(n)$. We have the following lemma.

Lemma 1 *For a quasi-reversible queue,* $\pi(\vec{x}|A_c(0))$ *is* independent *of* λ_c.

Proof:

From global balance, we have, $\forall\ \vec{x} \in A_c(0)$,

$$\pi(\vec{x}) \sum_{\vec{x}' \in A_c(0)} q(\vec{x}, \vec{x}') + \pi(\vec{x}) \sum_{\vec{y} \in A_c(1)} q(\vec{x}, \vec{y}) = \sum_{\vec{x}' \in A_c(0)} \pi(\vec{x}') q(\vec{x}', \vec{x}) + \sum_{\vec{y} \in A_c(1)} \pi(\vec{y}) q(\vec{y}, \vec{x})$$

But the second terms on each side of the equation are equal for a quasi-reversible queue (see equation (3.11) of [13]), or equivalently, for a queue satisfying the $M \Rightarrow M$ condition [16].

Therefore, for a quasi-reversible queue,

$$\forall\ \vec{x} \in A_c(0), \quad \pi(\vec{x}) \sum_{\vec{x}' \in A_c(0)} q(\vec{x}, \vec{x}') = \sum_{\vec{x}' \in A_c(0)} \pi(\vec{x}') q(\vec{x}', \vec{x}) \tag{4}$$

Now consider a matrix $Q^* = [\![q^*(\vec{x}, \vec{x}')]\!]$ which represents the Markov chain with state space truncated to $A_c(0)$. The entries in Q^* are defined as follows.

$$q^*(\vec{x}, \vec{x}') = q(\vec{x}, \vec{x}') \quad \text{if } \vec{x} \neq \vec{x}', \text{ for } \vec{x}, \vec{x}' \in A_c(0)$$

The diagonal terms of Q^* are chosen such that all rows in Q^* sum to 0. Clearly, all entries in Q^* are independent of λ_c. Let $\pi^* = [\![\pi^*(\vec{x})]\!]$ be the solution to the Markov process that corresponds to Q^*, namely $\pi^* Q^* = 0$. Since all entries in Q^* are independent of λ_c, $\pi^*(\vec{x})$ must be *independent* of λ_c. The global balance equations for the Markov process represented by Q^* are:

$$\forall\ \vec{x} \in A_c(0), \quad \pi^*(\vec{x}) \sum_{\vec{x}' \in A_c(0)} q^*(\vec{x}, \vec{x}') = \sum_{\vec{x}' \in A_c(0)} \pi^*(\vec{x}')q^*(\vec{x}', \vec{x}) \tag{5}$$

Clearly, $G\pi(\vec{x})$ for $\pi(\vec{x})$ satisfying Eq. (4) is a solution to Eq. (5), where G is the normalizing constant, i.e., $1/\sum_{\vec{x}' \in A_c(0)} \pi(\vec{x}')$. Therefore, $\pi^*(\vec{x})$ is a solution to Q^* where $\pi^*(\vec{x}) = \pi(\vec{x})/\sum_{\vec{x}' \in A_c(0)} \pi(\vec{x}') = \pi(\vec{x}|A_c(0))$. Notice that both G and $\pi(\vec{x})$ can be functions of λ_c's and we will denote G by $G(\lambda_c)$.

Since $\pi^*(\vec{x})$ is independent of λ_c, $\pi(\vec{x}|A_c(0))$ is independent of λ_c. □

We also need the following lemma.

Lemma 2 *For a quasi-reversible queue,* $\pi(\vec{x}|A_c(1))$ *is* independent *of* λ_c [1].

Proof:

From global balance, we have, $\forall\ \vec{x} \in A_c(1)$,

$$\pi(\vec{x}) \sum_{\vec{x}' \in A_c(1)} q(\vec{x}, \vec{x}') + \pi(\vec{x}) \sum_{\vec{y} \in A_c(0)} q(\vec{x}, \vec{y}) + \pi(\vec{x}) \sum_{\vec{z} \in A_c(2)} q(\vec{x}, \vec{z}) = \sum_{\vec{x}' \in A_c(1)} \pi(\vec{x}')q(\vec{x}', \vec{x}) + \sum_{\vec{y} \in A_c(0)} \pi(\vec{y})q(\vec{y}, \vec{x}) + \sum_{\vec{z} \in A_c(2)} \pi(\vec{z})q(\vec{z}, \vec{x})$$

Again, the last term on each side of the equation cancel out for quasi-reversible queues. Therefore, $\forall\ \vec{x} \in A_c(1)$,

$$\pi(\vec{x}) \sum_{\vec{x}' \in A_c(1)} q(\vec{x}, \vec{x}') + \pi(\vec{x}) \sum_{\vec{y} \in A_c(0)} q(\vec{x}, \vec{y}) = \sum_{\vec{x}' \in A_c(1)} \pi(\vec{x}')q(\vec{x}', \vec{x}) + \sum_{\vec{y} \in A_c(0)} \pi(\vec{y})q(\vec{y}, \vec{x}) \tag{6}$$

In the last term of the above equation, each $q(\vec{y}, \vec{x})$ corresponds to an arrival of a class c customer. Therefore, λ_c can be factored out of each $q(\vec{y}, \vec{x})$, namely, $q(\vec{y}, \vec{x}) = \lambda_c q'(\vec{y}, \vec{x})$, where $q'(\vec{y}, \vec{x})$ is independent of λ_c. Furthermore, from Lemma 1, $\pi(\vec{y})$ can be expressed as $\pi^*(\vec{y})/G(\lambda_c)$. Eq. (6) becomes, $\forall\ \vec{x} \in A_c(1)$,

[1] This lemma and the previous one suggest that a general proof of independence of $\pi(\vec{x}|A_c(n))$ is possible by induction. This is indeed true, but in the interest of space, we only give the proof for $A_c(0)$ and $A_c(1)$.

$$\sum_{\vec{x}' \in A_c(1)} \pi(\vec{x}')q(\vec{x}', \vec{x}) - \pi(\vec{x}) \left[\sum_{\vec{x}' \in A_c(1)} q(\vec{x}, \vec{x}') + \sum_{\vec{y} \in A_c(0)} q(\vec{x}, \vec{y}) \right] =$$

$$- \frac{\lambda_c}{G(\lambda_c)} \sum_{\vec{y} \in A_c(0)} \pi^*(\vec{y})q'(\vec{y}, \vec{x}) \qquad (7)$$

In the above equation, the first sum corresponds to transition into state $\vec{x}$ due to the arrivals and departures of other classes of customers. The second sum corresponds to transitions out of state $\vec{x}$ due to the arrivals and departures of other classes of customers. The third sum corresponds to transitions out of state $\vec{x}$ due to the departure of a class c customers; All of these transition rates are independent of λ_c. Furthermore, $\pi^*(\vec{y})$ and $q'(\vec{y}, \vec{x})$ are independent of λ_c. The above equation in matrix form then becomes:

$$\vec{\pi}\hat{Q} = \frac{\lambda_c}{G(\lambda_c)}\vec{b}$$

where $\vec{\pi} = [\![\pi(\vec{x})]\!]$ for $\vec{x} \in A_c(1)$, $\hat{Q}$ corresponds to the first three sum terms in Eq. (6), and $\vec{b} = [\![b(\vec{x})]\!]$ where $b(\vec{x}) = -\sum_{\vec{y} \in A_c(0)} \pi^*(\vec{y})q'(\vec{y}, \vec{x})$. $\hat{Q}$ and $\vec{b}$ are independent of λ_c. $-\hat{Q}$ is a diagonally dominant matrix with some rows that do not sum to zero. It can be shown that the matrix $C = I - D^{-1}\left[-\hat{Q}\right]$ is a substochastic matrix where D is the diagonal matrix of $-\hat{Q}$ [24]. It follows that $\hat{Q}^{-1}$ exists and $\hat{Q}^{-1} \geq 0$. Multiplying both side of the above equation by $\hat{Q}^{-1}$, we obtain,

$$\vec{\pi} = \frac{\lambda_c}{G(\lambda_c)}\vec{b}\left[\hat{Q}^{-1}\right]$$

Thus for any $\vec{x} \in A_c(1)$, $\pi(\vec{x})$ is a constant vector times $\lambda_c/G(\lambda_c)$. Therefore, $\pi(\vec{x}|A_c(1)) = \pi(\vec{x})/\sum_{\vec{x}' \in A_c(1)} \pi(\vec{x}')$ is *independent* of λ_c. □

Using Lemmas 1 and 2, we will show that $\pi(\underline{\vec{x}}|A_c)$, the conditional state probability in a closed queueing model, is independent of the routing of the chain k customer. Consider the model described in Section 2; let $\sigma(c)$ denote the distinguished center corresponding to chain c and let θ_c denote the visit ratio for class c at center $\sigma(c)$. We have the following lemma.

Lemma 3 *For a network of quasi-reversible queues with Markovian routing,* $\pi(\underline{\vec{x}}|A_c)$ *is* independent *of* θ_c.

Proof:

Since we have a network of quasi-reversible queues, the equilibrium state probability has a product form [13] [16]; namely, $\pi(\underline{\vec{x}}) = (1/G)\prod_{j=1}^{J}\pi_j(\vec{x}_j)$, where G is the normalization constant, and $\pi_j(\vec{x}_j)$ is the equilibrium state probability of being in state $\vec{x}_j$ for center j when it is driven by Poisson arrivals with rates $\theta_1, \ldots, \theta_{C^*}$ in isolation, where θ_c is the visit ratio for class c customer at center $j = \sigma(c)$ in the closed network model. C^* denotes the total number of classes over all chains. From the definition of $\pi(\underline{\vec{x}}|A_c)$, we have,

$$\pi(\underline{\vec{x}}|A_c) = \frac{\pi(\underline{\vec{x}})}{\sum_{\underline{\vec{y}}\in A_c}\pi(\underline{\vec{y}})} = \frac{\prod_{i=1}^{J}\pi_i(\vec{x}_i)}{\sum_{\underline{\vec{y}}\in A_c}\prod_{i=1}^{J}\pi_i(\vec{y}_i)} = \left[\sum_{\underline{\vec{y}}\in A_c}\prod_{i=1}^{J}\frac{\pi_i(\vec{y}_i)}{\pi_i(\vec{x}_i)}\right]^{-1} \tag{8}$$

In Eq. (8), if we can show that $\pi_i(\vec{y}_i)/\pi_i(\vec{x}_i)$ is independent of θ_c for $i = 1, \ldots, J$, then $\pi(\underline{\vec{x}}|A_c)$ is independent of θ_c. Extending the notation in Eq. (3) by adding subscript j, we have,

$$\pi(\vec{x}_j|A_c(n)) = \frac{\pi_j(\vec{x}_j)}{\sum_{\vec{x}'_j\in A_c(n)}\pi(\vec{x}'_j)} \tag{9}$$

In Eq. (8), if $i \neq \sigma(c)$, then $n_c(\vec{y}_i) = n_c(\vec{x}_i) = 0$, and

$$\frac{\pi_i(\vec{y}_i)}{\pi_i(\vec{x}_i)} = \frac{\pi_i(\vec{y}_i|A_c(0))}{\pi_i(\vec{x}_i|A_c(0))} \qquad \text{if } i \neq \sigma(c)$$

which is independent of θ_c by Lemma 1. (Notice that the normalizing terms in Eq. (9) cancel.) Similarly, if $i = \sigma(c)$ then $n_c(\vec{y}_i) = n_c(\vec{x}_i) = 1$, and

$$\frac{\pi_i(\vec{y}_i)}{\pi_i(\vec{x}_i)} = \frac{\pi_i(\vec{y}_i|A_c(1))}{\pi_i(\vec{x}_i|A_c(1))} \qquad \text{if } i = \sigma(c)$$

which is also independent of θ_c by Lemma 2. Hence for any center i, $\pi_i(\vec{y}_i)/\pi_i(\vec{x}_i)$ is independent of θ_c; therefore, $\pi(\underline{\vec{x}}|A_c)$ is independent of θ_c. □

Since routing of chain k can only affect the visit ratios θ_c for $c \in \mathcal{C}_k$ and the conditional state probability $\pi(\vec{\underline{x}}|A_c)$ is independent of θ_c, $\pi(\vec{\underline{x}}|A_c)$ is independent of the routing of chain k.

Refering back to Eq. (2), $R(\vec{\underline{x}})$, the reward rate for a given state $\vec{\underline{x}}$, is a constant, and therefore, it is independent of the routing decisions of chain k. We have also just proved that $\pi(\vec{\underline{x}}|A_c)$ is independent of the routing decisions of chain k; therefore, the summation terms in Eq. (2) are independent of chain k's routing. Let $R_c = \sum_{\vec{\underline{x}} \in A_c} \pi(\vec{\underline{x}}|A_c) R(\vec{\underline{x}})$, Eq. (2) can be written as

$$R = \sum_{c=1}^{C} w_c R_c \tag{10}$$

In this equation the only dependence of the steady state reward rate on the routing decisions is through the w_c terms, i.e., the steady state probability that the chain k customer is in class c.

Without loss of generality, let's examine the case in which the chain k customer leaves class C. Let the probability of moving from class C to class d be $v_{C,d}$ for $d = 1, \ldots, D$, and let the probability of moving to any other class be 0. For simplicity in notation, we will drop the subscript C in $v_{C,d}$ where the class is clear from the context. Clearly, every element in Q_{Cd} is either equal to 0 or v_d multiplied by a constant which corresponds to the rate at which the chain k customer leaves class C; therefore, v_d can be factored out of Q_{Cd}, or $Q_{Cd} = v_d Q'_{Cd}$ where Q'_{Cd} is a constant matrix. The rate matrix in Figure 1 can then be represented as shown in Figure 2.

$$\begin{bmatrix} Q_{11} & Q_{12} & & \cdots & & Q_{1C} \\ Q_{21} & Q_{22} & & \cdots & & Q_{2C} \\ & \vdots & & \ddots & & \\ v_1 Q'_{C1} & v_2 Q'_{C2} & \cdots & v_D Q'_{CD} & Q_{C(D+1)} & \cdots & Q_{CC} \end{bmatrix}$$

Figure 2: Rate matrix with routing decisions factored.

We can construct the exact aggregate matrix $\mathbf{Q}^{Ag}$ of the matrix $\mathbf{Q}$, where $\mathbf{Q}^{Ag}$ is a C by C matrix with $Q^{Ag}_{cd} = \vec{\pi}_c Q_{cd} \vec{1}^T$, where $\vec{\pi}_c$ denotes the vector of probabilities $[\![\pi(\vec{\underline{x}}|A_c)]\!]$ (confer [5] [17]). $\mathbf{Q}^{Ag}$ is shown in Figure 3. Let $\vec{w} = (w_1, \ldots, w_C)$ denote the solution to

$$\begin{bmatrix} Q_{11}^{Ag} & Q_{12}^{Ag} & & \cdots & Q_{1C}^{Ag} \\ Q_{21}^{Ag} & Q_{22}^{Ag} & & \cdots & Q_{2C}^{Ag} \\ & \vdots & & \ddots & \vdots \\ v_1\vec{\pi}_C Q'_{C1}\vec{1}^T & v_2\vec{\pi}_C Q'_{C2}\vec{1}^T & \cdots\ v_D\vec{\pi}_C Q'_{CD}\vec{1}^T & Q_{C(D+1)}^{Ag} & \cdots\ Q_{CC}^{Ag} \end{bmatrix}$$

Figure 3: $\mathbf{Q}^{Ag}$ – the exact aggregate of $\mathbf{Q}$.

$\mathbf{Q}^{Ag}$. Then w_c is just the probability of being in the aggregate state which corresponds to Q_{cc}^{Ag} (the solution to $\mathbf{Q}^{Ag}$), and R_c is the conditional reward of the aggregated state corresponding to Q_{cc}^{Ag}.

Let $\mathbf{Q}^{Ag}(d)$ denote the matrix obtained from $\mathbf{Q}^{Ag}$ by setting $v_d = 1$ and $v_{d'} = 0$ for $d' \neq d$ and $1 \leq d, d' \leq D$, and let $\vec{w}(d)$ denote the solution to $\mathbf{Q}^{Ag}(d)$. Let $R(d) = \sum_{c=1}^{C} w_c(d)R_c$ be the steady state reward rate if class C customer always changes to class d. We have the following lemma.

Lemma 4 $\min_d\{R(d)\} \leq R \leq \max_d\{R(d)\}$ *for* $1 \leq d \leq D$.

Proof:

In [6], it is shown that for a matrix such as $\mathbf{Q}^{Ag}$ above, its left eigenvector $\vec{w}$ is in the convex hull of $\{\vec{w}(d),\ 1 \leq d \leq D\}$, i.e., $\vec{w}$ can be expressed as:

$$\vec{w} = \sum_{d=1}^{D} \beta_d \vec{w}(d) \tag{11}$$

where $\sum \beta_d = 1$. Combining Eq. (10) and (11), we have,

$$R = \sum_c w_c R_c = \sum_c \left[\sum_d \beta_d w_c(d)\right] R_c = \sum_d \beta_d \left[\sum_c w_c(d) R_c\right] = \sum_d \beta_d R(d) \tag{12}$$

Eq. (12) says that the total reward R for the model is a convex combination of the $R(d)$'s. Clearly, $\min_d\{R(d)\} \leq R \leq \max_d\{R(d)\}$. □

Lemma 4 states that the steady state reward rate can be bounded above and below by the reward rate obtained from routing the class C customer to a class d deterministically. Now we will prove the vertex allocation theorem.

Let C^* denote the total number of classes over all chains, or $C^* = \sum_{k=1}^{K} C^k$, let $c, d \in \mathcal{C}_k$ for some chain k, and let $\phi_c = [\![\phi_{c,d}]\!]$ be a probability assignment to $[\![v_{c,d}]\!]$ such that Eq. (1) is satisfied. Let $\Phi_c = \{\phi_c\}$ be the set of all possible probability assignment for class c, and let $V_c \subset \Phi_c$ be the set of all possible *vertex assignment* for class c which is defined as follows,

$$V_c = \{\phi_c \mid \phi_c \in \Phi_c \text{ and } \phi_{c,d} \in \{0,1\}\}$$

Let $\phi = [\![\phi_1, \ldots, \phi_{C^*}]\!]$ be a probability assignment to all classes of all chains, and let $\Phi = \{\phi\}$ be the set of all possible assignments for all classes of all chains. $V \subset \Phi$ is call the *vertex set* of the routing decision space and is defined as follows:

$$V = \{\phi \mid \phi \in \Phi \text{ and } \forall c,\ 1 \leq c \leq C^*,\ \phi_c \in V_c\}$$

Theorem 1 *For a queueing network model described in Section 2, an optimal steady state reward rate can be obtained by a probability assignment in the vertex set.*

Proof:

Suppose there is no optimal probability assignment in the vertex set. Let $\phi \in \Phi$ and $\phi \notin V$ be a probability assignment that gives the optimal reward, say $R(\phi)$. Find a class c such that $\phi_c \notin V_c$. Consider all other variables to be fixed; from Lemma 4, we can improve the expected reward rate with a new assignment ϕ' such that $\phi_d = \phi'_d$ for $d \neq c$ and $\phi'_c \in V_c$. Set $\phi \leftarrow \phi'$ and repeat this procedure until $\phi \in V$. Therefore, a probability assignment in the vertex set exists that gives the optimal reward. □

4 Routing of Closed Chains

Theorem 1 provides the means to prune the search space from a continuum of values for decision variables to a finite number of possibilities, but the complexity of determining an optimal routing is still exponential in the number of decision variables. In the general, case there is no known efficient solution. In this section we consider a special case in which only

a single chain (customer) is being routed in the network where there are other customers present. As mentioned in the introduction, this corresponds to optimally routing a new customer which is being introduced into an operational system. For the "phase-type" routing described below, we can show that the total reward has only one local maximum and one local minimum in the routing decision space. In this case, we are able to devise an efficient hill climbing algorithm to find an optimal solution.

4.1 Phase-Type Routing

In this section, we consider a particular type of routing which we call the "phase-type" routing. This is a more restrictive form of routing than that described in Section 2. Phase-type routing is basically hierarchical in nature. At the higher level, the behavior of the job (or chain) is described via a Markov chain in which states are associated with "phases" or "functions" that the job executes. At the lower level each "phase" or "function" is associated with a set of open subnetworks which describe alternative ways for performing the function. By way of example, consider the description of a chain k customer as shown in Figure 4(a). Each state in the Markov chain in Figure 4(a) represents a phase of

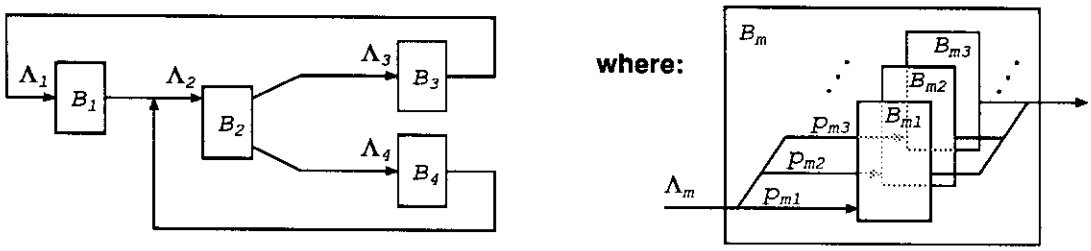

Figure 4: Phase-Type Routing for Chain k.

service. For example, B_2 might represent a phase of service which can be provided by the alternative service networks $B_{21}, B_{22}, \ldots, B_{2I_2}$ (see Figure 4(b)), where I_m is the total number of alternatives for phase m. The routing decision variables are the probabilities with which alternative service networks are selected, i.e. p_{mi} is the probability that the alternative service network B_{mi} is selected when the customer requests service in phase m. As before, we assume p_{mi} can be expressed as a non-negative constant plus a non-negative

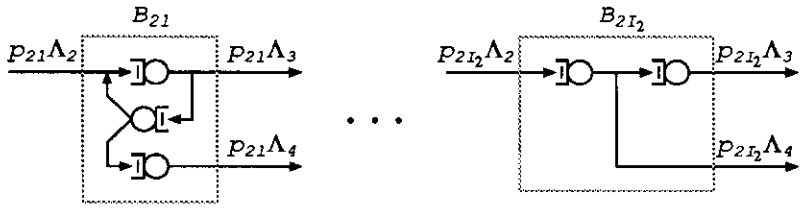

Figure 5: Alternative Service Networks for B_2.

variable subject to the constraint

$$\sum_i p_{mi} = 1, \quad \forall\, m \tag{13}$$

Figure 5 illustrates that the alternative service networks which provide the same function can have different topologies. The relative arrival rate to each phase can be computed by solving the job flow balance equations corresponding to the "phase transition diagram" such as the one shown in Figure 4(a). Let Λ_m denote the relative arrival rate to phase m. Then $p_{mi}\Lambda_m$ is the relative arrival rate into alternative service network B_{mi}. Let λ_{mij} denote the expected number of visits to center j in the i^{th} alternative service network of phase m for each entry to the i^{th} alternative network. Then in the corresponding queueing network model, it is easy to see that the visit ratios are given by:

$$\theta_{jK} = \sum_{m=1}^{M} \Lambda_m \sum_{i=1}^{I_m} p_{mi}\lambda_{mij} \tag{14}$$

The main restriction inherent in phase-type routing is that the Λ_m's are *independent* of the routing decisions variables $\{p_{mi}\}$. Since the Λ_m's are independent of the routing decisions and the λ_{mij}'s only depend on the topology within alternative B_{mi}, it is clear that each θ_{jK} is a *linear* function of the decision variables $\{p_{mi}\}$.

Note that the resources in the various alternatives for a phase are not necessarily disjoint and in fact, it is common for the same resource to be present in multiple alternatives.

In the next section we consider models with phase-type routing. We show that a Markov reward function results in the total reward being a quasi-linear function of the routing decision variables and develop a greedy algorithm for finding the optimal choice of routing.

4.2 Optimal Routing for a Single Chain

Consider the case of adding a new customer to an existing system where the routings of all existing chains are fixed and the route of the new customer has to be decided to optimize some performance measures. We assume that we have a BCMP type network (for the explanation of this restriction, refer to the discussion on the computation of the mean sojourn time below) and the performance measure of interest can be expressed as a Markov reward function. Let the new customer be chain K, and the other customers be chains 1 through $K-1$.

We will show that if chain K has the phase-type routing described in Section 2, the reward function R can be expressed as a ratio of linear functions of the routing decision variables, i.e. R is a *fractional linear function*; therefore, R is both pseudo-convex and pseudo-concave in the routing decision space [1], and a simple "greedy" algorithm that hill climbs from one vertex to another will find the optimum. For ease of exposition, we will assume that we only have binary decisions. The arguments presented are easily generalized to n-way routing decisions.

Let's focus our attention on Eq. (10) which is true even for the model with the general type of routing. The conditional reward rate R_c is independent of chain K's routing, and w_c is the probability that the chain K customer is in class c. Let $\sigma(c)$ denote the center that corresponds to class c. Then w_c can be expressed as [18],

$$w_c = \frac{\theta_{cK} l_c}{\sum_{c'=1}^{C} \theta_{c'K} l_{c'}} \tag{15}$$

where θ_{cK} is the relative frequency of chain K being in class c and l_c is chain K's mean sojourn time per visit to center $\sigma(c)$ when it is in class c. Clearly, the relative throughput (visit ratio) of chain K customer at center j is $\theta_{jK} = \sum_{c,\sigma(c)=j} \theta_{cK}$, and $l_j = \sum_{c,\sigma(c)=j} (\theta_{cK}/\theta_{jK}) l_c$ is chain K's mean sojourn time per visit to center j. Let w_j denote the probability that the chain K customer is at center $\sigma(c)$, then

$$w_j = \sum_{c,\sigma(c)=j} w_c = \frac{\sum_{c,\sigma(c)=j} \theta_{cK} l_c}{\sum_{c'=1}^{C} \theta_{c'K} l_{c'}} = \frac{\theta_{jK} l_j}{\sum_{j'=1}^{J} \theta_{j'K} l_{j'}} \tag{16}$$

The mean sojourn time of chain K at center $\sigma(c)$ is just the reciprocal of the conditional throughput of chain K if chain K is conditioned to be in class c self-looping at center j [2].

[2]This is well-known for BCMP network [4]. We are currently extending the proof for network of quasi-reversible queues.

Since the conditional throughput can be expressed as $\sum_{\underline{\vec{x}} \in A_c^K} \pi(\underline{\vec{x}}|A_c^K)R'(\underline{\vec{x}})$, where $R'(\underline{\vec{x}})$ measures chain K's throughput rate in state $\underline{\vec{x}}$, l_c is independent of chain K's routing.

Notice that Equations (15) and (16) have the same form. For convenience and economy of space, we assume that for each chain, the mean service time at center j is the same for all classes. This makes l_j independent of the routing decisions in the discussion follows. The development can be easily extended to use Eq. (15).

Using Eq. (16), the total reward R can be expressed as,

$$R = \frac{\sum_{j=1}^{J} \theta_{jK} l_j R_j}{\sum_{i=1}^{J} \theta_{iK} l_i} \tag{17}$$

in which the routing of the chain K customer only affects the visit ratios θ_{jK}'s. Below we concentrate on the nature of the dependence of the θ_{jK}'s on the routing variables. Basically, for the cases in which each θ_{jK} is a linear function of the routing variables, R is easily seen to be a fractional linear function of the routing variables. The phase-type routing introduced in this section is exactly such a case. Using the notation introduced above, each visit ratio, i.e., θ_{jK}, is a *linear* function of the decision variables $\{p_{mi}\}$ (confer Eq. (14)). Therefore, the reward function is a fractional linear function. Furthermore,

$$\frac{\partial \theta_{jK}}{\partial p_{mi}} = \Lambda_m \sum_{i=1}^{I_m} \lambda_{mij} \tag{18}$$

There are on the order of $I_m \times M$ systems of linear equations which need to be solved to obtain the λ_{mij}'s. ($2 \times M$ for the binary decision case.) One additional set of linear equations must be solved to obtain the Λ_m's. Furthermore, each conditional reward rate R_j and mean sojourn time l_j can be obtained by solving a queueing network model with chain K customer self-looping at center j, and the number of such models to be solved is equal to the number of centers visited by chain K.

From Eq. (17), the rate of change of the total reward with respect to a decision variable becomes,

$$\frac{\partial R}{\partial p_{mi}} = \frac{\left[\sum_j \theta_{jK} l_j\right]\left[\sum_j l_j R_j \frac{\partial \theta_{jK}}{\partial p_{mi}}\right] - \left[\sum_j \theta_{jK} l_j R_j\right]\left[\sum_j l_j \frac{\partial \theta_{jK}}{\partial p_{mi}}\right]}{\left[\sum_j \theta_{jK} l_j\right]^2} \tag{19}$$

Since θ_{jK} is a linear function of $\{p_{mi}\}$ for phase-type routing, $\partial \theta_{jK}/\partial p_{mi}$ is a constant with respect to p_{mi}. We can use the current allocation to compute θ_{iK}'s and obtain the

direction of greatest improvement on the total reward. This determines the next vertex allocation.

4.3 Greedy Algorithm

Here we summarize the greedy algorithm for finding the vertex allocation which maximizes the total reward for a model with phase-type routing.

Step 1: For each center j that may be visited by chain K, solve the network with chain K looping at center j. Set R_j to the total reward for this network; also set l_j to be the reciprocal of chain K's throughput at center j.

Step 2: Compute $\partial\theta_{jK}/\partial p_{mi}$ according to Eq. (18), $\forall\, j, m, i$.

Step 3: Let $n = 0$ and let $\vec{p}^{(0)}$ be an initial vertex allocation.

Step 4: Compute θ_{jK}, $\forall\, j$ using $\vec{p}^{(n)}$. Then compute $\partial R/\partial p_{mi}$ using Eq. (19).

Step 5: For each m, find $1 \leq i^* \leq I_m$ such that $\partial R/\partial p_{mi^*}$ is the "maximum" [3].

Step 6: If $p_{mi^*} = p_{mi}^{(n)} = 1$ for all m, then $\vec{p}^{(n)}$ is the optimal allocation and we stop. Otherwise, set $p_{mi}^{(n+1)} = 1$ if $i = i^*$, or 0 if $i \neq i^*$. Let $n = n + 1$ and go to Step 4.

In Step 1 of the above algorithm, the number of models need to be solved is equal to the number of centers visited by chain K. $2 \times M$ systems of linear equations need to be solved to obtain the partial derivatives $\partial\theta_{jK}/\partial p_{mi}$'s. Compared with solving 2^M queueing models, this is quite an improvement.

5 Approximations

The algorithm presented in the previous section requires computation of the reward rate, for each center, conditioned on the chain K customer being at that center as well as the mean sojourn time of chain K customer at that center. (For convenience, we still assume that the mean service time at a center is the same regardless of which class a customer is in.) If the parameters for all chains are known, these quantities can be calculated for a given center by solving a model with the chain K customer looping at the given

[3] Depending on the current allocation $\vec{p}^{(n)}$, "maximum" means most negative if $p_{mi}^{(n)} = 1$ and most positive if $p_{mi}^{(n)} = 0$.

center. However, if the number of chains is large, it can be expensive to compute the exact values by solving the queueing networks (one for each choice of center at which the chain K customer loops). It is of interest, therefore, to consider efficient approximations. In the approximation described below we assume that measurements of the system with the current population (before adding the new customer) are available and these measurements can provide an (approximately) alternative to solving the queueing network models. Our discussion here is limited to queueing networks with SSFR servers. Delay servers could easily be included but we choose not to for clarity in presentation.

As an example, consider the case where a large number of workstations are activated one at a time and routing decisions are made for each workstation in order to optimize the sum of throughputs. In such a model, the number of centers would be at least equal to the number of workstations plus the total number of servers. Therefore, K, the total number of chains, and J, the total number of centers, can both be large. Also, the system is somewhat dynamic in the sense that users come and go, so the resource demands for all the existing chains may not be known. We assume that it is possible to collect statistics to estimate, for example, the mean queue length and the mean load for each chain at each center. At chain K's arrival to the operational system, i.e., a new workstation coming on-line, these statistics (load indices) can be used in the heuristics to choose the customer's routing. In the following we develop an approximate version of the exact optimization algorithm that is based on Schweitzer's approximation [19] and the availability of estimates for current queue lengths and loads. The objective function assumed for this approximation is the sum of throughputs.

We will assume that the parameters (or at least estimates) for the new customer are known. To apply the algorithm presented in the previous section requires estimates of the reward rates R_j and the sojourn time l_j conditioned on the new customer being at center j, for all j. In the following we develop an approximation for these quantities.

Let $\vec{N}$ denote the population vector when chains 1 through K are present in the network. The reward rate (sum of throughputs in this case) conditioned on the new (chain K) customer being at center j can be expressed as:

$$R_j \;=\; \sum_{k=1}^{K-1} \lambda_k(\vec{N}) \;+\; \lambda_K(j) \tag{20}$$

where

$$\lambda_K(j) \;=\; 0 \quad \text{if } j(k) \neq j \;\text{ and }\; = \frac{1}{l_j} \;\text{ otherwise} \tag{21}$$

and where (for $k \neq K$)

$$\lambda_k(\vec{N}) \;=\; \frac{1}{\sum_{i=1}^{J} \theta_{ik} W_{ik}(\vec{N})} \tag{22}$$

In the following discussion we develop an approximation for the throughput $\lambda_k(\vec{N})$ where it is understood that the chain K customer is treated as looping at the j^{th} center. Rather than complicating the notation, we assume that the center j will be understood. Of course, the approximation must be computed for each choice of j to obtain all the required terms.

For product form queueing networks of fixed rate servers it is well known that the visit ratios can be suppressed in the above equation by defining $W_{ik}(\vec{N})$ as the mean sojourn time per cycle of customer k at center i (the cycle is measured relative to some specified center). Eq. (22) then becomes:

$$\lambda_k(\vec{N}) \;=\; \frac{1}{\sum_{i=1}^{J} W_{ik}(\vec{N})} \tag{23}$$

The unknowns that have to be estimated are therefore the mean sojourn times at each center per cycle. To estimate these quantities we will make use of the MVA equation:

$$W_{ik}(\vec{N}) \;=\; T_{ik}\left[1 + L_i(\vec{N} - \vec{e}_k)\right] \tag{24}$$

where T_{ik} is the load at the i^{th} center due to the chain k customer, per cycle of the chain k customer. We assume that these loads are known for each customer. Then the mean queue lengths that appear in Eq. (24) are the only unknowns. Below we develop an approximation for these mean queue lengths based on Schweitzer's approximation.

In words, Schweitzer's approximation says that when a given customer is removed from the network, the other customers remain distributed among the centers in approximately the same proportion as with the customer present. More formally, this is:

$$L_{ik'}(\vec{N} - \vec{e}_k) \;\approx\; L_{ik'}(\vec{N}) \quad 1 \leq k' \leq K-1, \quad k' \neq k$$

In our case we are adding a customer and the equivalent approximation is that the customers that were in the network will (approximately) remain distributed as they were before the new customer is added. More formally, let $\vec{N}$ be the population vector after adding the new customer and $\vec{N} - \vec{e}_K$ be the population vector prior to adding the new, chain K customer. Then the approximation is,

$$L_{ik'}(\vec{N} - \vec{e}_K) \approx L_{ik'}(\vec{N}) \quad 1 \leq k' \leq K-1$$

Together the above equations yield:

$$L_{ik'}(\vec{N} - \vec{e}_k) \approx L_{ik'}(\vec{N} - \vec{e}_K) \quad k' \neq k$$

It follows that:

$$L_i(\vec{N} - \vec{e}_k) = \sum_{k' \neq k} L_{ik'}(\vec{N} - \vec{e}_k) \approx \sum_{k' \neq k} L_{ik'}(\vec{N} - \vec{e}_K) \quad \text{for } i \neq j \text{ and } k \neq K \tag{25}$$

or

$$L_i(\vec{N} - \vec{e}_k) \approx \sum_{k' \neq k} L_{ik'}(\vec{N} - \vec{e}_K) + \delta_{ij} \quad \forall\, i \text{ and } k \neq K \tag{26}$$

where $\delta_{ij} = 1$ for $i \neq j$ and 0 otherwise.

Note that the δ_{ij} term accounts for the chain K customer looping at center j; it adds one to the queue length at center j. Putting this result into the MVA equation we obtain:

$$W_{ik}(\vec{N}) \approx T_{ik}\left[1 + \sum_{k' \neq k} L_{ik'}(\vec{N} - \vec{e}_K) + \delta_{ij}\right] \quad \text{for } k \neq K \tag{27}$$

In words, the approximation is that a chain k customer arriving at the i^{th} center, conditioned on the chain K customer being at center j, "sees", on the average, the mean number of other customers (i.e., excluding itself) at center i with population $\vec{N} - \vec{e}_K$, plus the chain K customer if i is where the chain K customer is conditioned to be.

Finally we note that l_j is simply $1/\lambda_K(\vec{N})$ where:

$$\lambda_K(\vec{N}) = \frac{1}{W_{jK}(\vec{N})} \tag{28}$$

$W_{jK}(\vec{N})$ can be "computed" *exactly* from measured quantities of the operational system:

$$W_{jK}(\vec{N}) = T_{jK}\left[1 + L_j(\vec{N} - \vec{e}_K)\right] \tag{29}$$

At this point it is simply a matter of applying the greedy algorithm given previously using the approximate values for R_j and l_j to find the optimal routing. In the next section we report on experiments using this approximation. Before concluding this section we discuss several other approximations that are possible using the same basic approach.

It is interesting to note that the above approximation can be very easily extended to form the basis of a heuristic for dynamic load balancing. Suppose the current system is modeled as a closed queueing network with K single customer chains, and we want to evaluate the possibility of rerouting a customer, say chain K. To reduce the problem to the routing problem that was just discussed, we need estimates for $L_{jk}(\vec{N} - \vec{e}_K)$ for all centers j and all chains $k < K$. Schweitzer's approximation suggests using

$$L_{jk}(\vec{N} - \vec{e}_K) \approx L_{jk}(\vec{N})$$

At this point the previous approximation can be invoked.

The approximation that is described above can be viewed as a more formal approach to selecting and justifying load indices for load balancing heuristics. In a recent paper [11], Ferrari and Zhou describe the derivation and use of a certain load index for routing (or assigning) a new command in a workstation environment. The objective function that is to be *minimized* is the new jobs's response time. (Note that effects on other jobs are not included in the objective function.) It is easy to see that minimizing the response time of this new job is equivalent to maximizing its throughput. For the sake of brevity, lets consider a very simple case in which there are J single server queues where the job can be executed and also that the job is initiated on a workstation on which there is no contention. The throughput of the job is measured relative to the workstation. Referring back to Eq. (17), it is straightforward to deduce that θ_{jK} is 1 when j corresponds to the workstation of job K, $R_j = 0$ for j not equal to the workstation, and when j does equal the workstation then $l_j = 1/R_j$. The result is that the total reward is:

$$R = \frac{1}{\sum_{i=1}^{J} \theta_{iK} l_i} \tag{30}$$

where the sum is over the possible servers. Since we know that vertex allocation applies to this problem, one of the θ_{iK} will be 1 and the rest are 0. It is clear that server i^* which yields the smallest value for l_i should always be chosen and server with the smallest mean queue length with population $\vec{N}$ will yield the smallest value for l_i. The result in [11]

includes delay servers and several different functions (with a set of possible servers for each function), but the result in these cases also follows as a special case of our development.

6 Numerical Examples

Consider a local area network with disk-less workstations and servers connected to a communication network. When the workstation comes on-line, routing decisions must be made concerning where to obtain certain types of services. Figure 6 depicts such a routing problem. There are 3 servers and 5 workstations in the network and workstation 5 is coming

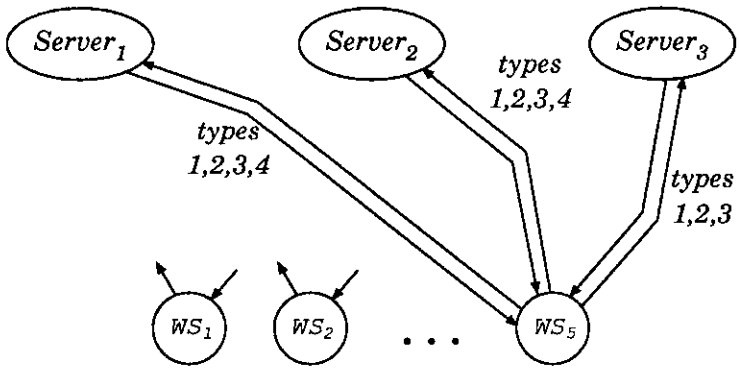

Figure 6: A routing example.

on-line. The user of WS_5 requests 4 types of services from the servers. 3 types of service can be obtained from any of the servers and the last type can only be provided by servers 1 and 2. After the job executes at a server, it always returns to the workstation. Let p_{mj} denote the probability that server j is selected in request type m. The relative frequency of the occurrence of the types of requests, which are the Λ_m's as described in Section 2 for phase-type routing, are $\Lambda_1 = 0.12$, $\Lambda_2 = 0.15$, $\Lambda_3 = 0.5$, $\Lambda_4 = 0.23$, and $\Lambda_{WS_5} = 1$. Table 1 shows the relative loads of the other workstations and the service time of the new customer.

In order to compare the results of the exact algorithm and the approximation procedure presented in Section 5, we assume that the system has reached steady state when the new customer joins the network, and therefore, the mean queue lengths of the network without the new customer can be obtained from solving the network with chains 1 through 4. Then the approximation presented in Section 5 is applied to obtain the conditional reward rates

chain (k)	$Server_1$	$Server_2$	$Server_3$	WS_k
relative loads of other chains				
1	0.4	0.25	0.25	3
2	0.8	0.25	0.35	4
3	0.35	0.9	0.9	4
4	0.15	0.3	0.4	3
service time of chain K				
K	0.3	0.3	0.2	3

Table 1: Reletive loads and service time.

and mean sojourn times. The exact conditional reward rates and mean sojourn times are solved using MVA with chain K looping at each center in turn. These values are shown in Table 2. Notice that each alternative service network contains only a single server (since

	WS_5	$Server_1$	$Server_2$	$Server_3$
exact solution				
R_j	1.1389	0.7393	0.7429	0.7319
l_j	3.0000	0.4225	0.4174	0.2922
approximate solution				
R_j	1.1351	0.7442	0.7472	0.7396
l_j	3.0000	0.4224	0.4173	0.2923

Table 2: Conditional reward rates and mean sojourn time comparisons.

the job returns to the workstation right after it visits a server) and the resources do not overlap in any of the alternative service networks, $\sum_{i=1}^{I_m} \lambda_{mij} = 1$ in Eq. (18) (if center j is visited by chain K), and therefore, $\partial\theta_{jK}/\partial p_{mj} = \Lambda_m$, $\forall\, i$. Table 3 shows the values for the $\partial\theta_j/\partial p_{mj}$'s. Table 4 shows the rate of change of the reward with respect to the decision variables $\{p_{mj}\}$, starting with an initial allocation where all phases execute at server 1. In both the exact and the approximate algorithm, the optimum is found in one step, which is the case where types 1 through 3 of services are obtained at server 3 and type 4 service is obtained at server 2. The * in Table 4 denotes the direction of the steepest ascent for the reward function.

	$Server_1$	$Server_2$	$Server_3$
$m = 1$	0.12	0.12	0.12
$m = 2$	0.15	0.15	0.15
$m = 3$	0.5	0.5	0.5
$m = 4$	0.23	0.23	N/A

Table 3: $\partial\theta_j/\partial p_{mj}$'s.

	$Server_1$	$Server_2$	$Server_3$	$Server_1$	$Server_2$	$Server_3$
exact $\partial R/\partial p_{mj}$						
	step 1			step 2		
$m = 1$	-0.0304	-0.0297	-0.0215*	-0.0304	-0.0297	-0.0214*
$m = 2$	-0.0380	-0.0371	-0.0268*	-0.0379	-0.0371	-0.0268*
$m = 3$	-0.1266	-0.1266	-0.0894*	-0.1264	-0.1265	-0.0893*
$m = 4$	-0.0582	-0.0570*	N/A	-0.0582	-0.0569*	N/A
approximated $\partial R/\partial p_{mj}$						
	step 1			step 2		
$m = 1$	-0.0297	-0.0291	-0.0208*	-0.0297	-0.0291	-0.0208*
$m = 2$	-0.0371	-0.0364	-0.0260*	-0.0371	-0.0364	-0.0260*
$m = 3$	-0.1238	-0.1238	-0.0868*	-0.1237	-0.1237	-0.0867*
$m = 4$	-0.0570	-0.0558*	N/A	-0.0569	-0.0558*	N/A

Table 4: $\partial R/\partial p_{mj}$ as the greedy algorithm executes.

In the above example, Schweitzer's approximation works quite well for estimating the conditional reward rates and the mean sojourn times. Furthermore, the search converges in one step in the above example. A large number of random versions of this and other examples have been run with randomly chosen parameters. The heuristic worked very well: the optimal solution was found and in one step in over 95% of the cases. The maximum difference between the optimal reward rate and the reward rate selected by the heuristic was 3%.

7 Conclusion

Optimal routing for closed queueing networks has proved to be a difficult problem. In this paper, we have described a generalized version of the vertex allocation theorem of Tripathi and Woodside. The proof method has an interesting probabilistic interpretation that suggests the characteristics that should be present in load indices to be used in routing, and it also suggests heuristics for routing or load balancing.

We proved the vertex allocation theorem for a general class of routing problems and for any Markov reward objective function. While this helps to reduce the search space, there is still no efficient solution to the problem of optimally routing a set of customers. We concentrated in this paper on the special case of optimally adding a single customer to an existing system. We developed a greedy algorithms and discussed approximations which are relatively efficient. Experiments to date have indicated that the greedy type algorithm for these problems is quite efficient; the optimal solution is often found in one step. Further experimentation is desirable to determine how robust the approximations are.

References

[1] M. S. Bazaraa. *Nonlinear Programming.* John Wiley and Sons, Inc., 1979.

[2] J. Y. Le Boudec. A bcmp extension to multiserver stations with concurrent classes of customers. In *Proceedings ACM SIGMETRICS*, Raleigh, NC, May 1986.

[3] W. C. Cheng and R. R. Muntz. *A General Proof of the Vertex-Allocation Theorem.* Technical Report, UCLA Computer Science Department, 1989.

[4] A. E. Conway and N. D. Georganas. *Queueing Networks – Exact Computational Algorithms.* MIT Press, 1989.

[5] P. J. Courtois and P. Semal. Bounds for the position eigenvectors of non-negative matrices and for their approximations. *J. ACM*, 31(4):804–825, October 1984.

[6] P.J. Courtois and P. Semal. Computable bounds for conditional steady-state probabilities in large Markov chains and queueing models. *IEEE Journal on Selected Areas in Communications*, SAC-4:926–937, 1986.

[7] E. de Souza e Silva and M. Gerla. Load balancing in distributed systems with multiple classes and site constraints. In *Proc. Performance '84 and 1984 ACM SIGMETRICS Conf.*, pages 17–33, 1984.

[8] E. de Souza e Silva and M. Gerla. *Queueing Network Models for Load Balancing in Distributed Systems.* Technical Report CSD-870069, UCLA Computer Science Department, December 1987.

[9] E. de Souza e Silva and S.S. Lavenberg. Calculating joint queue length distributions in product form queueing networks. *Journal of the ACM*, 36:194–207, 1989.

[10] D.L. Eager, E.D. Lazowska, and J. Zahorjan. Adaptive load sharing in homogeneous distributed systems. *IEEE Transactions on Software Engineering*, 12:662–675, 1986.

[11] D. Ferrari and S. Zhou. A load index for dynamic load balancing. *Fall Joint Computer Conference*, 684–690, 1986.

[12] L. Fratta, M. Gerla, and L. Kleinrock. The flow deviation method - an approach to store-and-forward communication network design. *Networks*, 3:97–133, 1973.

[13] F. P. Kelly. *Reversibility and Stochastic Networks.* John Wiley and Sons, 1979.

[14] H. Kobayashi and M. Gerla. Optimal routing in closed queueing networks. *ACM Trans. on Computer Systems*, 1:294–310, 1983.

[15] K.J. Lee and D. Towsley. A comparison of priority-based decentralized load balancing policies. In *Proc. Performance '86 and 1986 ACM SIGMETRICS Conf.*, pages 70–77, 1986.

[16] R. R. Muntz. *Poisson Departure Processes and Queueing Networks.* Technical Report RC4145, IBM Thomas J. Watson Research Center, Yorktown Heights, N.Y., 1972.

[17] R. R. Muntz, E. de Souza e Silva, and A. Goyal. Bounding availability of repairable computer systems. *SIGMETRICS*, September 1988.

[18] S.M. Ross. *Stochastic Processes.* Wiley, 1983.

[19] P. Schweitzer. Approximate analysis of multiclass closed networks of queues. In *International Conference on Stochastic Control and Optimization*, Amsterdam, 1979.

[20] A.N. Tantawi and D. Towsley. Optimal load balancing in distributed computer systems. *Journal of the ACM*, 32:442–462, 1985.

[21] S. K. Tripathi and C. M. Woodside. A vertex-allocation theorem for resources in queueing networks. *J. ACM*, 35(1):221–230, January 1988.

[22] S.K. Tripathi, Y. Huang, and D. Towsley. On optimal file allocation with sharing. In *Proceedings of the International Seminar on Performance of Distributed and Parallel Systems*, pages 1–14, Kyoto, Japan, 1988.

[23] K.S. Trivedi and R.E. Kinicki. A model for computer configuration design. *Computer*, 13:47–54, 1980.

[24] R. Varga. *Matrix Iterative Analysis.* Prentice Hall, 1962.

[25] Y.T. Wang and R.J.T Morris. Load sharing in distributed systems. *IEEE Trans. on Computers*, 34:204–217, 1985.

[26] C.M. Woodside and S.K. Tripathi. Optimal allocation of file servers in a local network environment. *IEEE Transactions on Software Engineering*, 12:844–848, 1986.