A queue with skill based service under FCFS-ALIS : steady state, overloaded system, and behavior under abandonments

EURANDOM PREPRINT SERIES
2012-011

May 23, 2012

A queue with skill based service under FCFS-ALIS: steady state, overloaded system, and behavior under abandonments

Ivo Adan, Gideon Weiss
ISSN 1389-2355

A queue with skill based service under FCFS-ALIS: steady state, overloaded system, and behavior under abandonments

Ivo Adan[*] Gideon Weiss[†]

May 23, 2012

Abstract

We consider a queueing system with servers $\mathcal{S} = \{m_1, \ldots, m_J\}$, and with customer types $\mathcal{C} = \{a, b, \ldots\}$. A bipartite graph G describes which pairs of server - customer type are compatible. We consider FCFS-ALIS policy: A server always picks the first, longest waiting compatible customer, a customer is always assigned to the longest idle compatible server. We assume Poisson arrivals and exponential service times. We derive an explicit product-form expression for the steady state distribution of this system when service capacity is sufficient. We analyze the system under overload, when partial steady state exists. Finally we describe the behavior of the system with generally distributed abandonments, under many arrivals - fast service scaling.

Keywords: Service system; multi type customers; multi type servers; matching infinite sequences; product form solution; first come first served policy; assign longest idle server policy; complete resource pooling; abandonment.

1 Introduction

In this paper we study a service system which serves several types of customers, labeled $a, b, c, \ldots$, we denote the set of customer types by $\mathcal{C}$. Service is provided by J servers, labeled $m_1, \ldots, m_J$, we denote the set of servers by $\mathcal{S}$. Service is skill based, in the sense that each server m_j has a non-empty subset of customer types which it can serve, given by $\mathcal{C}(m_j)$ the union of which is $\mathcal{C}$, and customers of type c have a non-empty set of servers which can serve them, given by $\mathcal{S}(c)$. This can be described by a bipartite graph between the servers and the customer types, with directed arc (c, m_j) if c can be served by m_j. We assume that this graph is connected. The motivation for such systems is that while they provide custom tailored service to the various types of customers, the overlap of server skills allows for resource pooling and reduced congestion.

We assume that arrivals are Poisson and service is exponential. Customers arrive at the system in independent Poisson streams with rates λ_c, $c \in \mathcal{C}$. Servers work at rates $\mu_{m_1}, \ldots, \mu_{m_J}$, with independent exponential service times. Note that service durations of customers depend on the server which provides the service and not on the type of customer.

Service discipline in the system is a combination of first come first served (FCFS) and assign longest idle server (ALIS). Arriving customers which find no compatible server available wait in

[*]Department of Mechanical Engineering, Eindhoven University of Technology, P.O. Box 513, 5600 MB Eindhoven, the Netherlands; email *iadan@tue.nl* Research supported in part by the Netherlands Organization for Scientific Research (NWO).

[†]Department of Statistics, The University of Haifa, Mount Carmel 31905, Israel; email *gweiss@stat.haifa.ac.il* Research supported in part by Israel Science Foundation Grant 711/09.

a single queue, and are processed in a FCFS order as long as it is possible. This means that as soon as a server finishes a service it takes the first customer in the queue that it can serve, possibly skipping several earlier customers that it cannot serve. Customers which upon arrival find some available compatible servers, are assigned to the one of them that has been idle for the longest time, and else they join the end of the queue.

Under this combined FCFS-ALIS regime the system is Markovian. A full description of the system will then consist of three parts: first the list of all the customers in the system in their order of arrivals, including customers which are being served, second the location in this list of each of the busy servers, where we imagine that the servers are situated in the queue at the position of the customer that they are serving, and third the list of all idle servers ordered by increasing idle time.

To illustrate we consider a system with three servers and three customer types, as shown in Fig. 1: there are three job types a, b, c and three servers with $\mathcal{C}(m_1) = \{a, b\}$, $\mathcal{C}(m_2) = \{a, c\}$, $\mathcal{C}(m_3) = \{a\}$.

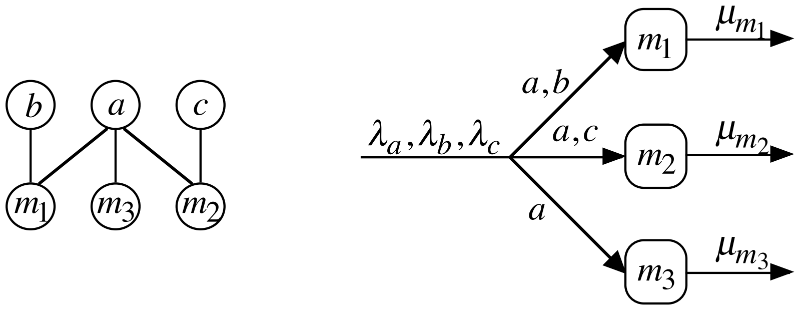

Figure 1: A system with three types of customers and three servers.

Three possible situations of this system are depicted in Fig. 2, which employs the following way to describe the state of the system: The customers are denoted by circles and the servers by rectangles. Customers in service have a rectangle drawn around them with the identity of the server inside. Idle servers are denoted by rectangles with a $*$ instead of a circle. The customers are ordered from left to right by increasing time of arrival, followed on the right by the idle servers, ordered from left to right by increasing idle times. One can visualize the dynamics of the system with customers arriving from the right, scanning the idle servers to pick the rightmost (longest idle) one that is compatible, and joining the queue at the rightmost position with this server, or without a server if none is compatible. Concurrently when a service is completed the customer is removed from the queue. The server that completed service moves to the right looking for the earliest waiting customer which is compatible, and starts serving it, or if no compatible customer is found the server joins the idle servers in the leftmost position. Note that waiting customers to the left of a server in this picture must be incompatible with that server, because of the FCFS rule.

In Fig. 2 part (i) there are 12 customers in the system, and all the servers are busy. Server m_1 is serving the first customer in line, which must therefore be either of type a or of type b. Following the queue to the right, server m_2 is serving the first customer in the line which it can serve, which is the 5th customer in the line, and must therefore be either type a or type c. Server m_3 is serving the first customer in the line (apart from customers 1 and 5) which it can serve, and must be of type a. There are 3 customers waiting between servers m_1 and m_2. These cannot be served by either server m_2 or by server m_3, so they must be type b customers. There are 4 customers waiting between m_2 and m_3, those cannot be served by server m_3, so they must be

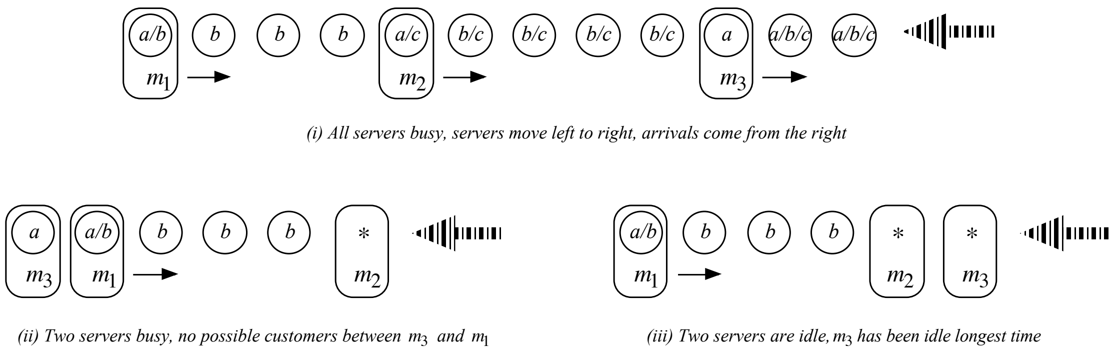
a/b
b
b
b
a/c
b/c
b/c
b/c
b/c
a
a/b/c
a/b/c
m_1
m_2
m_3
(i) All servers busy, servers move left to right, arrivals come from the right
a
a/b
b
b
b
*
m_3
m_1
m_2
(ii) Two servers busy, no possible customers between m_3 and m_1
a/b
b
b
b
*
*
m_1
m_2
m_3
(iii) Two servers are idle, m_3 has been idle longest time

Figure 2: A possible state for the system in Fig. 1.

of types b or c. Finally there are 2 customers at the tail of the queue, behind server m_3, which may be of types a, b, or c. The situation in part (ii) of Fig. 2 is that servers m_3, m_1 are busy, with server m_3 serving the earlier customer, while server m_2 is idle. There can be no customers waiting after server m_3 and before the next server, because servers m_1, m_2 combined can serve all types, and would have picked up the next customer after server m_3. The situation in part (iii) of Fig. 2 is that only server m_1 is busy, with 3 type b customers waiting and servers m_2, m_3 are idle, with server m_3 the one that became idle first. If the next arriving customer is of type a it will go to server m_3 (and not to m_2, because of ALIS). If it is of type c it will go to server m_2. If it is of type b the two servers will remain idle and the customer will join the queue in the last position.

We will actually aggregate some of the states in this detailed description, to simplify the model while retaining the Markovian property.

In the first part of this paper (Section 2) we will define the Markovian system, derive its transition rates, and set up partial balance equations. We will solve those to obtain conditions for ergodicity and an explicit product-form expression for the steady state distribution of the system. We will also derive the waiting time distribution for this system. Our system here has been analyzed in [24] under a different policy, in which service is FCFS but arriving customers are assigned to idle servers randomly, according to some assignment probability distributions. Surprisingly, we show that under these two different policies the stationary distributions of the system coincide.

In the second part of the paper (Section 3) we analyze the system under overload conditions. We define complete resource pooling of the system if the system is stable whenever the total arrival rate $\lambda = \sum_{c \in C} \lambda_c$ is less than the total service rate $\mu = \sum_{j=1}^{J} \mu_{m_j}$. Under complete resource pooling, if $\lambda > \mu$ the system is transient with the number of customers in the system growing without bound as $t \to \infty$. However, we show that at the same time, as $t \to \infty$ the servers stay together, and only the queue behind the last server grows without bound, while the state of the servers and the customers waiting between them converges to a stationary distribution which is the same as that obtained for FCFS infinite matching in [3]. We also consider the case when complete resource pooling does not hold. In that case the system under overload will decompose in a unique way to a partition of the servers, where each subset of servers stay together and serve a subset of the customers, and queues grow without bound between these groups of servers. Again, the state of each subset of servers and the customers waiting between them will converge to a stationary distribution given by [3].

In the third part of the paper (Section 4) we will consider the overloaded system with abandonments. We will assume that each customer type c has some general patience distribution F_c. We will then consider the system when λ and μ are large, so that many customers arrive and leave within a typical patience time span. We will then argue that under this scaling the system will stabilize. Again we will distinguish the case of complete resource pooling and of incomplete resource pooling. Under complete resource pooling there will be a single W such that all customers with patience exceeding W will be served after waiting for a total time W, while all customers with patience less than W will abandon. The system will contain a large number of customers, all arrivals over the last W length time interval, but the servers will progress together, with the distribution of servers and customers waiting between servers given by [3]. Without resource pooling the system will decompose uniquely into a partition of servers and of customer types, where each subset of servers with their customers will have their own value of W, and will behave again according to [3]. These results go some way towards confirming the conjectures stated in [23], and answering some of the questions posed in that paper.

Our model in this paper is described in the standard customer server language used for queueing models. However, it should find as much use also to describe the flow of jobs in a manufacturing system with non-homogenous machines, skill based routing of calls to operators in a call center, wireless messages to ad hoc nodes, calculations to processing chips, and so on. See in particular [5, 6, 8, 10, 17, 18, 19, 20, 21, 25].

2 The stable system

In this section we define a continuous time Markov chain to describe the dynamics of our system. We derive conditions for ergodicity for this Markov chain, and we obtain its stationary distribution, which is of product form. To define the Markov chain we aggregate some of the states in the detailed description, to simplify the model while retaining the Markovian property. We retain the identity and location of the busy servers, but we do not specify the type of customer which they are working on. Also we only record the number of jobs between the busy servers, and do not specify the string of job types. Finally, we retain the order of the idle servers. Thus the state of the system in Fig. 2(i) is denoted $(m_1, 3, m_2, 4, m_3, 0)$, the state in Fig. 2(ii) is $(m_3, 0, m_1, 3, m_2)$, and the state in Fig. 2(iii) is $(m_1, 3, m_2, m_3)$. Note that each busy server is followed by a number which counts how many (could be zero) customers are waiting behind him, while idle servers are not followed by a number.

We introduce the following notation:

M	$:=$	an arbitrary server M from the set of servers $\mathcal{S} = \{m_1, \ldots, m_J\}$. The capitalised M points to one of the servers m_j. Note that the names (or labels) of the servers m_j are not capitalised.
$\lambda_{\mathcal{X}}$	$:=$	$\sum_{c \in \mathcal{X}} \lambda_c$, where $\mathcal{X} \subset \mathcal{C}$
$\mu_{\mathcal{Y}}$	$:=$	$\sum_{m_j \in \mathcal{Y}} \mu_{m_j}$, where $\mathcal{Y} \subset \mathcal{S}$
$\mathcal{C}(\mathcal{Y})$	$:=$	total set of customer types that can be handled by the servers in $\mathcal{Y} \subset \mathcal{S}$, which is equal to $\bigcup_{m_j \in \mathcal{Y}} \mathcal{C}(m_j)$.
$\mathcal{U}(\mathcal{Y})$	$:=$	set of customer types unique to the servers in $\mathcal{Y} \subset \mathcal{S}$, thus the set of customer types that cannot be served by servers outside $\mathcal{Y}$. We have $\mathcal{U}(\mathcal{Y}) = \overline{\mathcal{C}(\overline{\mathcal{Y}})}$

$\mathcal{S}(\mathcal{X})$:= total set of servers that can serve customer types in $\mathcal{X} \subset \mathcal{C}$, which is equal to $\bigcup_{c\in\mathcal{X}} \mathcal{S}(c)$.

2.1 State definition and dynamics of the Markov chain

We define the state in general as:

$\boldsymbol{(M_1, n_1, \ldots, M_i, n_i, M_{i+1}, \ldots, M_J)}$: State in which there are i busy servers and $J-i$ idle servers with corresponding numbers of customers waiting between the busy servers. Here $M_1, \ldots, M_J$ is a permutation of $m_1, \ldots, m_J$. Servers $M_1, \ldots, M_i$ are serving customers of increasing arrival times, with n_j customers waiting between servers M_j and M_{j+1}, and servers $M_{i+1}, \ldots, M_J$ are idle, with increasing idle times. There is a total of $i + \sum_{j=1}^{i} n_i$ customers in the system.

The state space is denoted by $\mathfrak{S}$ and to simplify the notation we use $\mathfrak{s}$ to denote an arbitrary state $\mathfrak{s} = (M_1, n_1, \ldots, M_i, n_i, M_{i+1}, \ldots, M_J) \in \mathfrak{S}$. Fig. 3 shows a system in state $\mathfrak{s}$. There are a

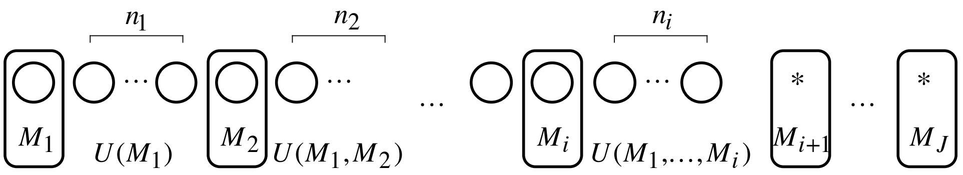

Figure 3: General system in state $\mathfrak{s} = (M_1, n_1, \ldots, M_i, n_i, M_{i+1}, \ldots, M_J)$.

few things that are important to note about this state description:

First, the waiting customers between servers M_j and M_{j+1} can only be handled by the servers $M_1, \ldots, M_j$ and not by any of the servers $M_{j+1}, \ldots, M_J$. This is due to the FCFS serving order. Thus waiting customers between servers M_j and M_{j+1} can only be of type $c \in \mathcal{U}(\{M_1, \ldots, M_j\})$. The n_i waiting customers in the back of the queue cannot be handled by any of the idle servers and have to be of type $c \in \mathcal{U}(\{M_1, \ldots, M_i\})$.

Second, since each part of the queue between two machines contains jobs from different subsets of job types it is necessary to keep these sets separated in the state description. It is not possible to aggregate the state description any further without losing the Markov property. However, each of the n_j customers within one of these queues is a customer of type c with probability $\frac{\lambda_c}{\lambda_{\mathcal{U}(\{M_1,\ldots,M_j\})}}$ independent of all the others. This is because the queue between servers M_j and M_{j+1} contains all the customers of types $\mathcal{U}(\{M_1, \ldots, M_j\})$ that arrived between the two customers served by M_j, M_{j+1}, in their original order of arrival.

Third, it is possible that the set of customer types $\mathcal{U}(\{M_1, \ldots, M_j\})$ is empty for a certain set of servers $\{M_1, \ldots, M_j\}$. In this case there are no customers which cannot be handled by any of the servers $M_{j+1}, \ldots, M_J$. Thus there can be no waiting jobs between M_j and M_{j+1}, and therefore n_j can only be equal to zero. In that case one also has that $n_1, n_2, \ldots, n_{j-1} = 0$.

Fourth, it is important to note that in this state description we lose customer type information about the customers that are in service, since we only denote the server that is handling the customer and not the type of the customer. This aggregation preserves the Markov property since all types are processed by server m_j at rate μ_{m_j}. We conjecture that specifying the customer types in service will destroy the possibility of a product form solution.

The dynamics of the Markov chain are as follows: when the system is in state $\mathfrak{s}$ the following transitions are possible:

(i) When a customer of type c arrives it will activate server M_J if $c \in \mathcal{C}(M_J)$, it will activate server M_j for $i+1 \le j < J$ if $c \in \mathcal{C}(M_j)\backslash\mathcal{C}(\{M_{j+1},\dots,M_J\})$. The customer will move to the end of the queue with the activated server which will then become M_{i+1} with $n_{i+1}=0$. If $c \in \mathcal{U}(\{M_1,\dots,M_i\})$ then the customer will move to the end of the queue and wait, the idle servers will remain unchanged, and n_i will increase by one.

(ii) When server M_j completes service, it will scan the customers in queue from left to right, starting with the n_j customers queued behind it, and continuing with the queues behind servers $M_{j+1},\dots,M_i$. It will skip a customer in the queue behind M_k if the customer type is $c \in \mathcal{U}(\{M_1,\dots,M_k\})\backslash\mathcal{C}(M_j)$, and will pick up the first customer of type $c \in \mathcal{U}(\{M_1,\dots,M_k\}) \cap \mathcal{C}(M_j)$. If it finds no customer to serve it will join the idle servers, in the leftmost position.

It is readily verified that the Markov chain on $\mathfrak{S}$ is irreducible (cf. Sect. 3.1 in [24]). In the next section we will specify the transitions and transitions rates of the Markov chain in detail.

2.2 Transitions and transition rates into a state $\mathfrak{s}$

In order to formulate the equilibrium equations it is convenient to list all the transitions into a state $\mathfrak{s}$, and obtain their transition rates. Transitions into $\mathfrak{s}$ can be caused by completion of service and a departure of a customer or by an arrival of a new customer.

(i) Transitions into state $\mathfrak{s} = (M_1, n_1, \dots, M_i, n_i, M_{i+1}, \dots, M_J) \in \mathfrak{S}$ due to completion of service and departure of customer:

When a service is completed a customer departs and the server scans queueing customers from left to right to find the first customer that it can serve. There are two different ways that state $\mathfrak{s} = (M_1, n_1, \dots, M_i, n_i, M_{i+1}, \dots, M_J) \in \mathfrak{S}$ can be reached.

(1) *The server does not find a customer to serve and becomes idle.* This transition is illustrated in Fig. 4. Such a transition is possible to state $\mathfrak{s} = (M_1, n_1, \dots, M_i, n_i, M_{i+1}, \dots, M_J)$ from state $(M_1, n_1, \dots, M_k, n_k - l, M_{i+1}, l, \dots, M_i, n_i, M_{i+2}, \dots, M_J)$, where server M_{i+1} is situated in the queue between M_k and M_{k+1}. We denote this state by $idle_{kl}(\mathfrak{s})$. In this state there are $n_k - l$

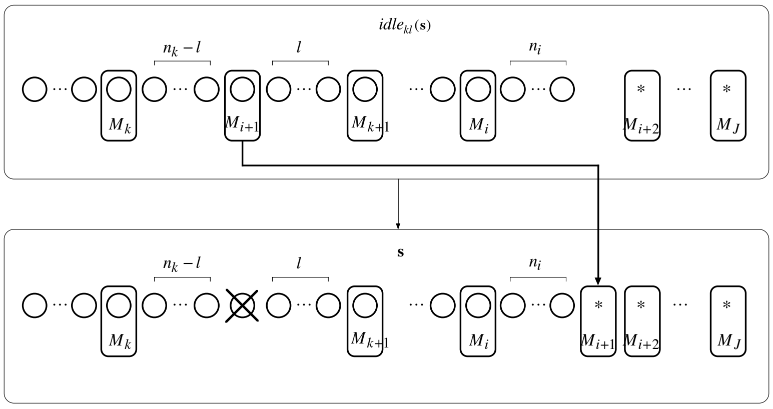

Figure 4: Transition from state $idle_{k,l}(\mathfrak{s})$ to state $\mathfrak{s}$

customers between M_k and M_{i+1} and l customers between M_{i+1} and M_{k+1}. The customers between servers M_{i+1} and M_{k+1} can only be of type $c \in \mathcal{U}(\{M_1, \ldots, M_k, M_{i+1}\})$, thus with probability

$$\frac{\lambda_{\mathcal{U}(\{M_1,\ldots,M_k\})}}{\lambda_{\mathcal{U}(\{M_1,\ldots,M_k,M_{i+1}\})}},$$

such a customer cannot be served by server M_{i+1}. A customer between server M_j and M_{j+1}, $j > k$, cannot be served by server M_{i+1} with probability

$$\frac{\lambda_{\mathcal{U}(\{M_1,\ldots,M_j\})}}{\lambda_{\mathcal{U}(\{M_1,\ldots,M_j,M_{i+1}\})}}.$$

Thus, if server M_{i+1} completes a service, then, with probability $p_{k,l}(\mathfrak{s})$, he will not find a customer in the queue that he can handle and will move to position $i + 1$ and become idle, where $p_{k,l}(\mathfrak{s})$ is defined as:

$$p_{k,l}(\mathfrak{s}) := \delta_k(M_{i+1})^l \delta_{k+1}(M_{i+1})^{n_{k+1}} \ldots \delta_i(M_{i+1})^{n_i},$$

with

$$\delta_j(M) := \frac{\lambda_{\mathcal{U}(\{M_1,\ldots,M_j\})}}{\lambda_{\mathcal{U}(\{M_1,\ldots,M_j,M\})}}, \qquad j = 1, 2, \ldots, J. \tag{1}$$

By convention we set $\delta_j(M) = 0$ when $\mathcal{U}(\{M_1, \ldots, M_j\}) = \mathcal{U}(\{M_1, \ldots, M_j, M\}) = \emptyset$. Thus with probability $p_{k,l}(\mathfrak{s})$ a transition is made from state $idle_{k,l}(\mathfrak{s})$ to state $\mathfrak{s}$, given that server M_{i+1} completes service.

In the special case of $k = 0$, before the transition into $\mathfrak{s}$ server M_{i+1} is serving the first customer in the queue, and in that case $l = 0$, since otherwise all customers between the first and second busy servers must be of type in $\mathcal{U}(\{M_{i+1}\})$, and server M_{i+1} will be able to serve the first of them and not become idle. Hence for $k = 0$ the only transition in which server M_{i+1} becomes idle is from $idle_{0,0}(\mathfrak{s})$ to $\mathfrak{s}$, and in fact:

$$p_{0,0}(\mathfrak{s}) = p_{1,n_1}(\mathfrak{s}).$$

(2) *The server finds a customer to serve and moves to the right in the queue.* This transition is illustrated in Fig. 5. In this case one of the busy servers completes a service and finds somewhere

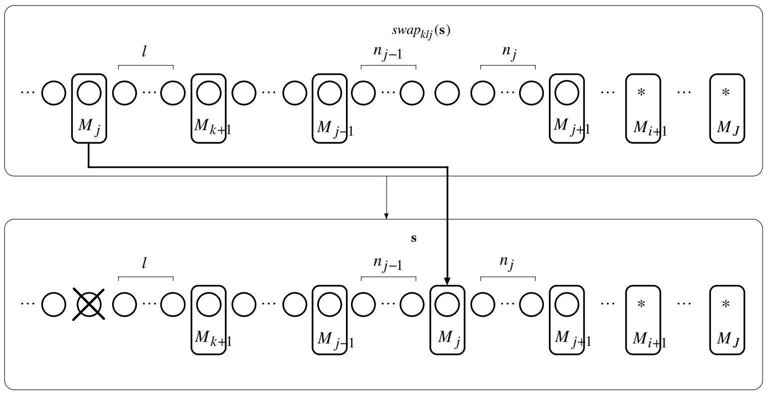

Figure 5: Transition from state $swap_{k,l,j}(\mathfrak{s})$ to state $\mathfrak{s}$.

in the queue a new customer that he can serve. The state $\mathfrak{s}$ can be reached by such a transition from state

$$(M_1, n_1, \ldots, M_k, n_k - l, M_j, l, \ldots, M_{j-1}, n_{j-1} + 1 + n_j, M_{j+1}, \ldots, M_J).$$

This state will be denoted by $swap_{k,l,j}(\mathfrak{s})$ with $j-1 \geq k$. In this state, server M_j is located between server M_k and M_{k+1}. Between servers M_{j-1} and M_{j+1} there are $n_{j-1}+1+n_j (>0)$ customers; note that this is not possible if $\mathcal{U}(\{M_1,\dots,M_j\})=\emptyset$ (since then there can be no customers between M_{j-1} and M_{j+1}). When server M_j completes service a transition is made to state $\mathfrak{s}$ if the first customer that M_j can serve is the $n_{j-1}+1$-th customer in the queue between M_{j-1} and M_{j+1}. The probability of this event is:

$$q_{k,l,j}(\mathfrak{s}) = \delta_k(M_j)^l \delta_{k+1}(M_j)^{n_{k+1}} \dots \delta_{j-1}(M_j)^{n_{j-1}} \left(1-\delta_{j-1}(M_j)\right),$$

with $\delta_j(S)$ defined in (1). The system makes a transition from state $swap_{k,l,j}(\mathfrak{s})$ to state $\mathfrak{s}$ with probability $q_{k,l,j}(\mathfrak{s})$, given that M_j completes service.

In the special case that $k=j-1$ server M_j starts and ends its move between servers M_{j-1} and M_{j+1}, and there are initially n_j+1+l customers between M_j and M_{j+1} and $n_{j-1}-l$ customers between M_{j-1} and M_j. This state is denoted by $swap_{j-1,l,j}(\mathfrak{s})$ and the probability that a transition is made from this state to state $\mathfrak{s}$ equals $q_{j-1,l,j}(\mathfrak{s})$.

Slightly different is the special case that $k=0$, where there are two possibilities: If $j=1$ then $k=j-1=0$, and the originating state is $swap_{0,0,1}(\mathfrak{s})$. In the transition from $swap_{0,0,1}(\mathfrak{s})$ to $\mathfrak{s}$ server M_1 was serving the first customer in the system, and there were n_1+1 customers queued between it and server M_2, and upon completion of service the server moves to the next customer. Because the next customer is in $\mathcal{U}(\{M_1\})$ the probability for this transition is

$$q_{0,0,1}(\mathfrak{s}) = 1.$$

Otherwise, if $j>1$ then we must have that in the originating state server M_j is serving the first customer in the system, and there are no customers between M_j and the next server M_1, so we have $l=0$. In that case the originating state is $swap_{0,0,j}(\mathfrak{s})$ and the transition probability is

$$q_{0,0,j}(\mathfrak{s}) = q_{1,n_1,j}(\mathfrak{s}).$$

(ii) Transitions into state $\mathfrak{s}=(M_1,n_1,\dots,M_i,n_i,M_{i+1},\dots,M_J)\in\mathfrak{S}$ due to arrival of a new customer:

When a customer arrives he will scan the idle servers from right to left to find a server, and will move to the end of the queue with or without a server. There are two different ways that state $\mathfrak{s}=(M_1,n_1,\dots,M_i,n_i,M_{i+1},\dots,M_J)\in\mathfrak{S}$ can be reached.

(1) *The customer does not find an idle server:* Such a transition is possible to state $\mathfrak{s}=(M_1,n_1,\dots,M_i,n_i,M_{i+1},\dots,M_J)$ from state $(M_1,n_1,\dots,\dots,M_i,n_i-1,M_{i+1},\dots,M_J)$, where $n_i>0$. We denote this state by $wait(\mathfrak{s})$. In this transition the new customer cannot be served by any of the idle servers, and simply joins the end of the queue to become the n_i-th customer behind server M_i. Transition from $wait(\mathfrak{s})$ to $\mathfrak{s}$ will happen if the customer is of type c and $c\in\mathcal{U}(\{M_1,\dots,M_i\})$. The rate of the transitions from $wait(\mathfrak{s})$ to $\mathfrak{s}$ is $\lambda_{\mathcal{U}(\{M_1,\dots,M_i\})}$.

(2) *The customer finds an idle server and activates it:* This transition is illustrated in Fig. 6. Such a transition is possible to state $\mathfrak{s}=(M_1,n_1,\dots,M_i,0,M_{i+1},\dots,M_J)$ in which M_i is active but $n_i=0$, from a state in which the last active server is M_{i-1} and server M_i is idle, and is located with the idle servers $M_{i+1},\dots,M_J$, where the possible positions for M_i are before M_k, where $k=i+1,\dots,J$, or after M_J. The originating states are then: $(M_1,n_1,\dots,M_{i-1},n_{i-1},M_{i+1},\dots,M_{k-1},M_i,M_k,\dots,M_J)$, where $k=i+1,\dots,J$, which we denote by $activate_k(\mathfrak{s})$, or from originating state $(J_1,n_1,\dots,J_{i-1},n_{i-1},M_{i+1},\dots,M_J,M_i)$, which we denote (as a convention) by $activate_{J+1}(\mathfrak{s})$. Transition from state $activate_{J+1}(\mathfrak{s})$ to state $\mathfrak{s}$ happens if a customer of type $c\in\mathcal{C}(M_i)$ arrives. Transition from state $activate_k(\mathfrak{s})$ to state $\mathfrak{s}$ happens if a customer of type c arrives and it cannot be served by $M_k,\dots,M_J$ but it can be served

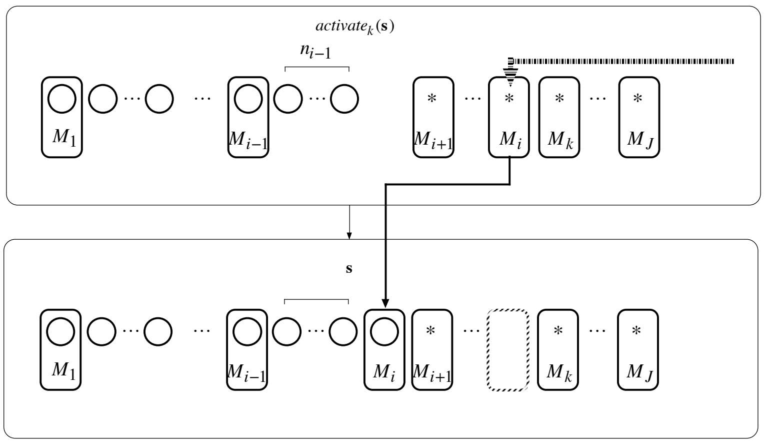

Figure 6: Transition from state $activate_k(\mathfrak{s})$ to state $\mathfrak{s}$.

by M_i, that is if $c \in \mathcal{C}(M_i) \backslash \mathcal{C}(\{M_k, \ldots, M_J\})$. The rate of transitions from $activate_{J+1}(\mathfrak{s})$ to $\mathfrak{s}$ is $\lambda_{\mathcal{C}(M_i)}$. The rate of transitions from $activate_k(\mathfrak{s})$ to $\mathfrak{s}$ is $\lambda_{\mathcal{C}(M_i) \backslash \mathcal{C}(\{M_k, \ldots, M_J\})}$, $k = i+1, \ldots, J$.

2.3 Equilibrium equations

We can now formulate the set of equilibrium equations. The equilibrium probability of being in the state $\mathfrak{s}$ is denoted by $\pi(\mathfrak{s})$. The state $\mathfrak{s}$ can be reached by (i-1) a departure of a customer from a server that then finds no other customer and becomes idle, (i-2) a departure of a customer from a server which then finds another customer in the queue, (ii-1) an arrival of a customer that finds no idle server and waits at the end of the queue, (ii-2) an arrival of a customer that finds an idle server to serve it and activates that server. The equilibrium equations display these four possibilities. The left hand side of the equations equals the total probability flux out of state $\mathfrak{s}$. The right hand side of the equations equals the probability flux into state $\mathfrak{s}$ and consist of four parts, corresponding to respectively (i-1), (i-2), (ii-1) and (ii-2).

In part (i-1) the busy server that becomes newly idle in the transition into $\mathfrak{s}$ must be server M_{i+1}, and we need to sum over all states with one more customer being served by M_{i+1}, so that the summation is over all k, l for which $idle_{k,l}(\mathfrak{s}) \in \mathfrak{S}$. In part (i-2) we need to sum over all busy servers $M_1, \ldots, M_i$ in $\mathfrak{s}$, and over all the positions of these servers in the queue before the transition, so it is over all states $swap_{k,l,j}$ where $k \leq j-1$ and $swap_{k,l,j}(\mathfrak{s}) \in \mathfrak{S}$. Part (ii-1) consists of a transition from the single state $wait(\mathfrak{s})$ of one less customer in the queue behind the last busy server, if indeed $n_i > 0$ in $\mathfrak{s}$, and if $\mathcal{U}(\{M_1, \ldots, M_i\}) \neq \emptyset$. In part (ii-2) we have that M_i is newly activated in $\mathfrak{s}$, and we need to sum over all positions in which M_i was among the idle machines prior to the transition, so we need to sum over states $activate_k(\mathfrak{s})$. The terms are non-zero only if $\lambda_{C(M_i) \backslash C(\{M_k, \ldots, M_J\})} \neq \emptyset$

The equilibrium equations are for all states $\mathfrak{s} = (M_1, n_1, \ldots, M_i, n_i, M_{i+1}, \ldots, M_J) \in \mathfrak{S}$ given by:

$$\begin{aligned} \left(\lambda_{\mathcal{C}} + \mu_{\{M_1,\ldots,M_i\}}\right) \pi(\mathfrak{s}) = \quad & \lambda_{\mathcal{U}(\{M_1,\ldots,M_i\})} \pi(wait(\mathfrak{s})) && \text{(ii-1)} \\ & + \mu_{M_{i+1}} \mathcal{P}_{M_{i+1}}(\mathfrak{s}) && \text{(i-1)} \\ & + \sum_{j=1}^{i} \mu_{M_j} \mathcal{Q}_{M_j}(\mathfrak{s}), \qquad n_i > 0 && \text{(i-2)} \end{aligned} \tag{2}$$

$$\begin{aligned}\left(\lambda_{\mathcal{C}} + \mu_{\{M_1,\dots,M_i\}}\right)\pi(\mathfrak{s}) = \; & \sum_{k=i+1}^{J} \lambda_{\mathcal{C}(M_i)\setminus\mathcal{C}(\{M_k,\dots,M_J\})}\pi(\mathit{activate}_k(\mathfrak{s})) \\ & +\lambda_{\mathcal{C}(M_i)}\pi(\mathit{activate}_{J+1}(\mathfrak{s})) && \text{(ii-2)} \\ & +\mu_{M_{i+1}}\mathcal{P}_{M_{i+1}}(\mathfrak{s}) && \text{(i-1)} \\ & +\sum_{j=1}^{i}\mu_{M_j}\mathcal{Q}_{M_j}(\mathfrak{s}), \qquad n_i = 0 && \text{(i-2)}\end{aligned} \tag{3}$$

where

$$\begin{aligned}\mathcal{P}_{M_{i+1}}(\mathfrak{s}) &= \sum_{k=1}^{i}\sum_{l=0}^{n_k} p_{k,l}(\mathfrak{s})\pi(\mathit{idle}_{k,l}(\mathfrak{s})) + p_{1,n_1}(\mathfrak{s})\pi(\mathit{idle}_{0,0}(\mathfrak{s})), \\ \mathcal{Q}_{M_j}(\mathfrak{s}) &= \sum_{k=1}^{j-1}\sum_{l=0}^{n_k} q_{k,l,j}(\mathfrak{s})\pi(\mathit{swap}_{k,l,j}(\mathfrak{s})) + q_{0,0,j}\pi(\mathit{swap}_{0,0,j}(\mathfrak{s})).\end{aligned}$$

The above definition of $\mathcal{Q}_{M_j}(\mathfrak{s})$ assumes $\mathcal{U}(\{M_1,\dots,M_j\}) \neq \emptyset$. If this set is empty, the states $\mathit{swap}_{k,l,j}(\mathfrak{s})$ are not feasible (see (i-2) in Sect. 2.2), and then $\mathcal{Q}_{M_j}(\mathfrak{s})$ is set to 0.

2.4 Partial balance equations

We will in fact show that the process satisfies partial balance equations as follows, for state $\mathfrak{s}$:

(i) The total probability flux out of state $\mathfrak{s}$ due to an arrival of a customer that activates a server (with arrival intensity $\lambda_{C(\{M_{i+1},\dots,M_J\})}$) equals the total probability flux into state $\mathfrak{s}$ due to a departure of a customer of server M_{i+1} after which server M_{i+1} becomes idle:

$$\lambda_{C(\{M_{i+1},\dots,M_J\})}\pi(\mathfrak{s}) = \mu_{M_{i+1}}\mathcal{P}_{M_{i+1}}(\mathfrak{s}). \tag{4}$$

(ii) The total probability flux out of state $\mathfrak{s}$, due to an arrival of a customer that cannot be served by any of the idle servers (with arrival intensity $\lambda_{\mathcal{U}(\{M_1,\dots,M_i\})}$), equals the total probability flux into state $\mathfrak{s}$, due to the departure of a customer from a server which then finds another customer to serve, so that the set of busy servers $\{M_1,\dots,M_i\}$ remains the same in both transitions:

$$\lambda_{\mathcal{U}(\{M_1,\dots,M_i\})}\pi(\mathfrak{s}) = \sum_{j=1}^{i}\mu_{M_j}\mathcal{Q}_{M_j}(\mathfrak{s}). \tag{5}$$

Note that, since $C(\{M_{i+1},\dots,M_J\})$ and $\mathcal{U}(\{M_1,\dots,M_i\})$ are a partition of $\mathcal{C}$, the left hand sides of (4)-(5) add up to $\lambda_C\pi(\mathfrak{s})$, which is the total probability flux out of state $\mathfrak{s}$ due to an arrival.

(iii) The total probability flux out of state $\mathfrak{s}$ in which $n_i = 0$ due to a departure of a customer, equals the total probability flux into state $\mathfrak{s}$, due to an arrival of a customer which activates server M_i:

$$\begin{aligned}\mu_{\{M_1,\dots,M_i\}}\pi(\mathfrak{s}) = \; & \sum_{k=i+1}^{J}\lambda_{\mathcal{C}(M_i)\setminus\mathcal{C}(\{M_k,\dots,M_J\})}\pi(\mathit{activate}_k(\mathfrak{s})) \\ & +\lambda_{\mathcal{C}(M_i)}\pi(\mathit{activate}_{J+1}(\mathfrak{s})), \qquad n_i = 0.\end{aligned} \tag{6}$$

(iv) The total probability flux out of state $\mathfrak{s}$ in which $n_i > 0$ due to a departure of a job, equal the total probability flux into state $\mathfrak{s}$, due to an arrival of a customer which does not find an idle server to serve it, and joins the end of the queue M_i:

$$\mu_{\{M_1,\dots,M_i\}}\pi(\mathfrak{s}) \quad = \quad \lambda_{\mathcal{U}(\{M_1,\dots,M_i\})}\pi(wait(\mathfrak{s})), \qquad n_i > 0. \tag{7}$$

2.5 The stationary distribution

The following theorem states that the stationary distribution has a product form.

Theorem 1 *The solution to the equilibrium equations (2)-(3) is given by*

$$\pi(\mathfrak{s}) = B \prod_{j=1}^{i} \frac{\lambda_{\mathcal{U}(\{M_1,\dots,M_j\})}{}^{n_j}}{\mu_{\{M_1,\dots,M_j\}}{}^{n_j+1}} \prod_{j=i+1}^{J} \lambda_{\mathcal{C}(\{M_j\dots,M_J\})}{}^{-1}. \tag{8}$$

The Markov chain is ergodic if the two equivalent conditions hold:

$$\begin{array}{lll} \lambda_C < \mu_{\mathcal{S}(C)}, & \textit{for } C \subset \mathcal{C}, & C \neq \emptyset, \mathcal{C}, \\ \lambda_{\mathcal{U}(S)} < \mu_S, & \textit{for } S \subset \mathcal{S}, & S \neq \emptyset, \mathcal{S}, \end{array} \tag{9}$$

After normalisation this solution becomes the stationary distribution, with normalization constant:

$$B = \left(\sum_{\mathcal{P}_J} \sum_{i=0}^{J} \prod_{j=1}^{i} \left(\mu_{\{M_1,\dots,M_j\}} - \lambda_{\mathcal{U}(\{M_1,\dots,M_j\})}\right)^{-1} \prod_{j=i+1}^{J} \left(\lambda_{\mathcal{C}(\{M_j,\dots,M_J\})}\right)^{-1} \right)^{-1}, \tag{10}$$

where $\mathcal{P}_J$ is the set of all the permutations of $\{m_1, \dots, m_J\}$.

Proof. The proof consists of verifying the partial balance equations (i)-(iv) formulated in the previous section. For a permutation $M_1, \dots, M_J$ of the servers we use the notation:

$$\theta_j = \frac{\lambda_{\mathcal{U}(\{M_1,\dots,M_j\})}}{\mu_{\{M_1,\dots,M_j\}}}, \quad j = 1, \dots, J.$$

To verify (iv):
Substitute (8) into the l.h.s. of (7) to get:

$$\mu_{\{M_1,\dots,M_i\}} B \left(\prod_{j=1}^{i} \mu_{\{M_1,\dots,M_j\}} \prod_{j=i+1}^{J} \lambda_{\mathcal{C}(\{M_j\dots,M_J\})} \right)^{-1} \theta_1^{n_1} \cdots \theta_{i-1}^{n_{i-1}} \theta_i^{n_i}$$

and into the r.h.s of of (7) to get:

$$\lambda_{\mathcal{U}(\{M_1,\dots,M_i\})} B \left(\prod_{j=1}^{i} \mu_{\{M_1,\dots,M_j\}} \prod_{j=i+1}^{J} \lambda_{\mathcal{C}(\{M_j\dots,M_J\})} \right)^{-1} \theta_1^{n_1} \cdots \theta_{i-1}^{n_{i-1}} \theta_i^{n_i-1}.$$

These are equal since:

$$\theta_i = \frac{\lambda_{\mathcal{U}(\{M_1,\dots,M_i\})}}{\mu_{\{M_1,\dots,M_i\}}}.$$

To verify (iii):
Substitute (8) into the l.h.s. of (6) to get:

$$\mu_{\{M_1,\ldots,M_i\}} B \left(\prod_{j=1}^{i} \mu_{\{M_1,\ldots,M_j\}} \prod_{j=i+1}^{J} \lambda_{\mathcal{C}(\{M_j \ldots,M_J\})} \right)^{-1} \theta_1^{n_1} \cdots \theta_{i-1}^{n_{i-1}},$$

since $n_i = 0$. Substitute (8) into the r.h.s of (7) to get:

$$\sum_{k=i+1}^{J} \lambda_{\mathcal{C}(M_i)\backslash\mathcal{C}(\{M_k,\ldots,M_J\})} B \left[\prod_{j=1}^{i-1} \mu_{\{M_1,\ldots,M_j\}} \left(\prod_{j=i+1}^{k-1} \lambda_{\mathcal{C}(\{M_j \ldots,M_i,M_k,\ldots,M_J\})} \right) \lambda_{\mathcal{C}(\{M_i,M_k,\ldots,M_J\})} \prod_{j=k}^{J} \lambda_{\mathcal{C}(\{M_j,\ldots,M_J\})} \right]^{-1} \theta_1^{n_1} \cdots \theta_{i-1}^{n_{i-1}}$$
$$+\lambda_{\mathcal{C}(M_i)} B \left[\prod_{j=1}^{i-1} \mu_{\{M_1,\ldots,M_j\}} \left(\prod_{j=i+1}^{J} \lambda_{\mathcal{C}(\{M_j,\ldots,M_J,M_i\})} \right) \lambda_{\mathcal{C}(M_i)} \right]^{-1} \theta_1^{n_1} \cdots \theta_{i-1}^{n_{i-1}}.$$

We need to verify that:

$$\left(\prod_{j=i+1}^{J} \lambda_{\mathcal{C}(\{M_j \ldots,M_J\})} \right)^{-1}$$
$$= \sum_{k=i+1}^{J} \lambda_{\mathcal{C}(M_i)\backslash\mathcal{C}(\{M_k,\ldots,M_J\})} \left[\left(\prod_{j=i+1}^{k-1} \lambda_{\mathcal{C}(\{M_j \ldots,M_i,M_k,\ldots,M_J\})} \right) \lambda_{\mathcal{C}(\{M_i,M_k,\ldots,M_J\})} \prod_{j=k}^{J} \lambda_{\mathcal{C}(\{M_j,\ldots,M_J\})} \right]^{-1}$$
$$+\lambda_{\mathcal{C}(M_i)} \left[\left(\prod_{j=i+1}^{J} \lambda_{\mathcal{C}(\{M_j,\ldots,M_J,M_i\})} \right) \lambda_{\mathcal{C}(M_i)} \right]^{-1},$$

or:

$$\sum_{k=i+1}^{J} \frac{\lambda_{\mathcal{C}(M_i)\backslash\mathcal{C}(\{M_k,\ldots,M_J\})} \prod_{j=i+1}^{k-1} \lambda_{\mathcal{C}(\{M_j \ldots,M_J\})}}{\lambda_{\mathcal{C}(\{M_i,M_k,\ldots,M_J\})} \prod_{j=i+1}^{k-1} \lambda_{\mathcal{C}(\{M_j \ldots,M_i,M_k,\ldots,M_J\})}} + \frac{\prod_{j=i+1}^{J} \lambda_{\mathcal{C}(\{M_j \ldots,M_J\})}}{\prod_{j=i+1}^{J} \lambda_{\mathcal{C}(\{M_j \ldots,M_J,M_i\})}} = 1. \quad (11)$$

This holds trivially for $i = J$, as the sum and products are empty (by convention, an empty sum is 0, an empty product is 1).

For $i = J - 1$ we obtain:

$$\frac{\lambda_{\mathcal{C}(M_{J-1})\backslash\mathcal{C}(M_J)}}{\lambda_{\mathcal{C}(\{M_{J-1},M_J\})}} + \frac{\lambda_{\mathcal{C}(M_J)}}{\lambda_{\mathcal{C}(\{M_{J-1},M_J\})}} = 1.$$

Generally, for $i < J$ we proceed by induction. We rewrite the l.h.s. of (11) as:

$$\begin{aligned}
\text{l.h.s} &= \frac{\lambda_{\mathcal{C}(M_i)\backslash\mathcal{C}(\{M_{i+1},\ldots,M_J\})}}{\lambda_{\mathcal{C}(\{M_i,M_{i+1},\ldots,M_J\})}} + \frac{\lambda_{\mathcal{C}(\{M_{i+1} \ldots,M_J\})}}{\lambda_{\mathcal{C}(\{M_i,M_{i+1},\ldots,M_J\})}} \\
&\quad \times \left[\sum_{k=i+2}^{J} \frac{\lambda_{\mathcal{C}(M_i)\backslash\mathcal{C}(\{M_k,\ldots,M_J\})} \prod_{j=i+2}^{k-1} \lambda_{\mathcal{C}(\{M_j \ldots,M_J\})}}{\lambda_{\mathcal{C}(\{M_i,M_k,\ldots,M_J\})} \prod_{j=i+2}^{k-1} \lambda_{\mathcal{C}(\{M_j \ldots,M_i,M_k,\ldots,M_J\})}} + \frac{\prod_{j=i+2}^{J} \lambda_{\mathcal{C}(\{M_j \ldots,M_J\})}}{\prod_{j=i+2}^{J} \lambda_{\mathcal{C}(\{M_j \ldots,M_J,M_i\})}} \right] \\
&= \frac{\lambda_{\mathcal{C}(M_i)\backslash\mathcal{C}(\{M_{i+1},\ldots,M_J\})}}{\lambda_{\mathcal{C}(\{M_i,M_{i+1},\ldots,M_J\})}} + \frac{\lambda_{\mathcal{C}(\{M_{i+1} \ldots,M_J\})}}{\lambda_{\mathcal{C}(\{M_i,M_{i+1},\ldots,M_J\})}} = 1,
\end{aligned}$$

where we used the induction hypothesis to say that the quantity in brackets equals 1.

To verify (i):

The terms into the r.h.s of (4) are:

$$\mu_{M_{i+1}} \left(\sum_{k=1}^{i} \sum_{l=0}^{n_k} p_{k,l}(\mathfrak{s}) \pi(idle_{k,l}(\mathfrak{s})) + p_{1,n_1}(\mathfrak{s}) \pi(idle_{0,0}(\mathfrak{s})) \right).$$

Substituting (8) into an individual term in the summation we get:

$$\begin{aligned}
&\mu_{M_{i+1}} p_{k,l}(\mathfrak{s}) \pi(idle_{k,l}(\mathfrak{s})) \\
&= \mu_{M_{i+1}} \delta_k(M_{i+1})^l \delta_{k+1}(M_{i+1})^{n_{k+1}} \dots \delta_i(M_{i+1})^{n_i} \pi(idle_{k,l}(\mathfrak{s})) \\
&= \mu_{M_{i+1}} B \left(\prod_{j=1}^{k} \mu_{\{M_1,\dots,M_j\}} \prod_{j=k}^{i} \mu_{\{M_1,\dots,M_j,M_{i+1}\}} \prod_{j=i+2}^{J} \lambda_{\mathcal{C}(\{M_j\dots,M_J\})} \right)^{-1} \\
&\theta_1^{n_1} \cdots \theta_k^{n_k - l} \left(\delta_k(M_{i+1}) \frac{\lambda_{\mathcal{U}(\{M_1,\dots,M_k,M_{i+1}\})}}{\mu_{\{M_1,\dots,M_k,M_{i+1}\}}} \right)^l \\
&\left(\delta_{k+1}(M_{i+1}) \frac{\lambda_{\mathcal{U}(\{M_1,\dots,M_{k+1},M_{i+1}\})}}{\mu_{\{M_1,\dots,M_{k+1},M_{i+1}\}}} \right)^{n_{k+1}} \cdots \left(\delta_i(M_{i+1}) \frac{\lambda_{\mathcal{U}(\{M_1,\dots,M_i,M_{i+1}\})}}{\mu_{\{M_1,\dots,M_i,M_{i+1}\}}} \right)^{n_i} \\
&= \mu_{M_{i+1}} B \left(\prod_{j=1}^{k} \mu_{\{M_1,\dots,M_j\}} \prod_{j=k}^{i} \mu_{\{M_1,\dots,M_j,M_{i+1}\}} \prod_{j=i+2}^{J} \lambda_{\mathcal{C}(\{M_j\dots,M_J\})} \right)^{-1} \\
&\theta_1^{n_1} \cdots \theta_k^{n_k - l} \left(\frac{\lambda_{\mathcal{U}(\{M_1,\dots,M_k\})}}{\mu_{\{M_1,\dots,M_k,M_{i+1}\}}} \right)^l \\
&\left(\frac{\lambda_{\mathcal{U}(\{M_1,\dots,M_{k+1}\})}}{\mu_{\{M_1,\dots,M_{k+1},M_{i+1}\}}} \right)^{n_{k+1}} \cdots \left(\frac{\lambda_{\mathcal{U}(\{M_1,\dots,M_i\})}}{\mu_{\{M_1,\dots,M_i,M_{i+1}\}}} \right)^{n_i} \\
&= \mu_{M_{i+1}} B \left(\prod_{j=1}^{k} \mu_{\{M_1,\dots,M_j\}} \prod_{j=k}^{i} \mu_{\{M_1,\dots,M_j,M_{i+1}\}} \prod_{j=i+2}^{J} \lambda_{\mathcal{C}(\{M_j\dots,M_J\})} \right)^{-1} \\
&\theta_1^{n_1} \cdots \theta_i^{n_i} \left(\frac{\mu_{\{M_1,\dots,M_k\}}}{\mu_{\{M_1,\dots,M_k,M_{i+1}\}}} \right)^l \left(\frac{\mu_{\{M_1,\dots,M_{k+1}\}}}{\mu_{\{M_1,\dots,M_{k+1},M_{i+1}\}}} \right)^{n_{k+1}} \cdots \left(\frac{\mu_{\{M_1,\dots,M_i\}}}{\mu_{\{M_1,\dots,M_i,M_{i+1}\}}} \right)^{n_i} \\
&= \lambda_{\mathcal{C}(\{M_{i+1},\dots,M_J\})} B \left(\prod_{j=1}^{i} \mu_{\{M_1,\dots,M_j\}} \prod_{j=i+1}^{J} \lambda_{\mathcal{C}(\{M_j\dots,M_J\})} \right)^{-1} \theta_1^{n_1} \cdots \theta_i^{n_i} \\
&\frac{\mu_{M_{i+1}}}{\mu_{\{M_1,\dots,M_k,M_{i+1}\}}} \left(\frac{\mu_{\{M_1,\dots,M_k\}}}{\mu_{\{M_1,\dots,M_k,M_{i+1}\}}} \right)^l \left(\frac{\mu_{\{M_1,\dots,M_{k+1}\}}}{\mu_{\{M_1,\dots,M_{k+1},M_{i+1}\}}} \right)^{n_{k+1}+1} \cdots \left(\frac{\mu_{\{M_1,\dots,M_i\}}}{\mu_{\{M_1,\dots,M_i,M_{i+1}\}}} \right)^{n_i+1} \\
&= \lambda_{\mathcal{C}(\{M_{i+1},\dots,M_J\})} \, \tilde{\pi}(\mathfrak{s}) \, (1 - \gamma_k) \, \gamma_k^l \, \gamma_{k+1}^{n_{k+1}+1} \, \cdots \, \gamma_i^{n_i+1},
\end{aligned}$$

where we put $\tilde{\pi}(\mathfrak{s})$ for the presumed form of $\pi(\mathfrak{s})$ according to (8), and define

$$\gamma_j = \frac{\mu_{\{M_1,\dots,M_j\}}}{\mu_{\{M_1,\dots,M_j,M_{i+1}\}}}.$$

For the $0,0$ term on the r.h.s. of (4) we get similarly:

$$\mu_{M_{i+1}} p_{1,n_1}(\mathfrak{s}) \pi(idle_{0,0}(\mathfrak{s}))$$

$$
\begin{aligned}
&= \mu_{M_{i+1}} \delta_1(M_{i+1})^{n_1} \cdots \delta_i(M_{i+1})^{n_i} \pi(idle_{0,0}(\mathfrak{s})) \\
&= \mu_{M_{i+1}} B \left(\mu_{M_{i+1}} \prod_{j=1}^{i} \mu_{\{M_1,\dots,M_j,M_{i+1}\}} \prod_{j=i+2}^{J} \lambda_{\mathcal{C}(\{M_j \dots, M_J\})} \right)^{-1} \\
&\left(\delta_1(M_{i+1}) \frac{\lambda_{\mathcal{U}(\{M_1,M_{i+1}\})}}{\mu_{\{M_1,M_{i+1}\}}} \right)^{n_1} \cdots \left(\delta_i(M_{i+1}) \frac{\lambda_{\mathcal{U}(\{M_1,\dots,M_i,M_{i+1}\})}}{\mu_{\{M_1,\dots,M_i,M_{i+1}\}}} \right)^{n_i} \\
&= \lambda_{\mathcal{C}(\{M_{i+1},\dots,M_J\})} \, \tilde{\pi}(\mathfrak{s}) \, \gamma_1^{n_1+1} \, \cdots \, \gamma_i^{n_i+1}.
\end{aligned}
$$

We can therefore rewrite the r.h.s. of (4) as

$$
\lambda_{\mathcal{C}(\{M_{i+1},\dots,M_J\})} \, \tilde{\pi}(\mathfrak{s}) \, \left[\left(\sum_{k=1}^{i} \sum_{l=0}^{n_k} (1-\gamma_k) \, \gamma_k^l \, \gamma_{k+1}^{n_{k+1}+1} \, \cdots \, \gamma_i^{n_i+1} \right) + \gamma_1^{n_1+1} \, \cdots \, \gamma_i^{n_i+1} \right].
$$

We now argue that the sum of all the γ terms on the left hand side is 1. We note that the γ_k represent probabilities for Bernoulli trials, of which there are altogether $\sum_{k=1}^{i}(n_k+1)$ trials, starting with n_i+1 trials with success probability of $(1-\gamma_i)$, followed by n_k+1 trials with success probability $(1-\gamma_k)$, for $k=i-1,\dots,2,1$. The summation of terms $\sum_{k=1}^{i}\sum_{l=0}^{n_k}$ sums up the probabilities that the first success will be on the first, the second, ... or the last of the trials, while the last summand of the right hand side is the probability of no success at all. These obviously add up to 1.

We have therefore shown that the r.h.s. of (4) equals the presumed form of the l.h.s. according to (8), which verifies (i).

The argument to verify (ii) is similar; see also [24].

Calculation of the normalization constant is straightforward from summation of the geometric sequences; note that it is the inverse of a sum of $J!$ terms. $\square$

Note that, if $\mathcal{U}(\{M_1,\dots,M_j\}) = \emptyset$, then $\lambda_{\mathcal{U}(\{M_1,\dots,M_j\})} = 0$ and hence, $\pi(\mathfrak{s}) = 0$ for all $\mathfrak{s} = (M_1, n_1, \dots, M_j, n_j, \dots, M_J)$ with $n_j > 0$, as it should.

2.6 Comparison with random assignment model

In a recent paper [24] the same queueing system under a different policy has been analysed. Surprisingly we show in this section that the steady state distribution under that policy is the same as for the system under FCFS-ALIS which we derived here.

The policy used in [24] is as follows: When a server becomes available he still will take the longest waiting customer that he can serve, FCFS. However, when a customer arrives and finds several idle servers which can serve him, he will not go to the longest idle server, but will instead choose one of the compatible idle servers randomly, using a random assignment probability. Thus, if a customer of type c arrives to find servers $M_1,\dots,M_i$ busy, he will go to server M_j which is idle and for which $c \in \mathcal{C}(M_j)$ with probability $P(c, M_j | \{M_1,\dots,M_i\})$. It is shown in [24] that these probabilities can always be chosen in such a way that the queueing system (if it is stable) will have a product form solution. While these special assignment probabilities may not be unique, they will determine unique values of $\lambda_{M_j}(\{M_1,\dots,M_i\})$, the rate at which idle server M_j will be activated when servers $\{M_1,\dots,M_i\}$ are busy. These unique activation rates can be calculated recursively from:

$$
\lambda_{\mathcal{C}(\mathcal{S}\setminus\{M_1,\dots,M_i\})} = \sum_{M \notin \{M_1,\dots,M_i\}} \lambda_M(\{M_1,\dots,M_i\}), \tag{12}
$$

$$\frac{\lambda_{M_j}(\{M_1,\ldots,M_i\})}{\lambda_{\mathcal{C}(\mathcal{S}\backslash\{M_1,\ldots,M_i\})}} = \Big(1 + \sum_{M_k\notin\{M_1,\ldots,M_i,M_j\}} \frac{\lambda_{M_k}(\{M_1,\ldots,M_i,M_j\})}{\lambda_{M_j}(\{M_1,\ldots,M_i,M_k\})}\Big)^{-1},$$
$$i \le J-2,\ M_j \notin \{M_1,\ldots,M_i\}.$$

These activation rates have the special property that:

$$\prod_{j=1}^{i} \lambda_{M_j}(\{M_1,\ldots,M_{j-1}\}) = \prod_{j=1}^{i} \lambda_{\overline{M}_j}(\{\overline{M}_1,\ldots,\overline{M}_{j-1}\}) \tag{13}$$

for every permutation $\overline{M}_1,\ldots,\overline{M}_i$ of $M_1,\ldots,M_i$.

The dynamics of the system under the random assignment policy are described by a Markov chain $Y(t)$, whose states list the busy servers in order, and the number of customers queueing between them, given by $M_1,n_1,\ldots,n_{i-1},M_i,n_i$, where the idle servers are $\mathcal{S}\backslash\{M_1,\ldots,M_i\}$. The steady state distribution, when the assignment probabilities and the activation rates are as described above, are given by (see [24], Theorem 2)

$$\pi_Y(M_1,n_1,\ldots,n_{i-1},M_i,n_i) = \pi_Y(0)\prod_{j=1}^{i}\frac{\lambda_{\mathcal{U}(\{M_1,\ldots,M_j\})}{}^{n_j}}{\mu_{\{M_1,\ldots,M_j\}}{}^{n_j+1}}\prod_{j=1}^{i}\lambda_{M_j}(\{M_1,\ldots,M_{j-1}\}). \tag{14}$$

We will now denote the steady state probabilities of the system under FCFS-ALIS as derived in Theorem 1 by $\pi_X(\cdot)$ to distinguish them from $\pi_Y(\cdot)$.

Theorem 2 *The system under random assignment, satisfying the assignment condition of [24], and the system under FCFS-ALIS share the same stationary behavior in the sense that:*

$$\pi_Y(M_1,n_1,\ldots,n_{i-1},M_i,n_i) = \sum_{\overline{M}_{i+1},\ldots,\overline{M}_J\in\mathcal{P}(\{M_{i+1},\ldots,M_J\})} \pi_X(M_1,n_1,\ldots,n_{i-1},M_i,n_i,\overline{M}_{i+1},\ldots,\overline{M}_J)$$

where $\mathcal{P}(\{M_{i+1},\ldots,M_J\})$ denotes the set of all the permutations of $M_{i+1},\ldots,M_J$.

Proof. Comparing (8) and (14), and recalling that π_X and π_Y both sum to 1, what we have to show is that for some constant D (which is the same for any $M_1,\ldots,M_i$):

$$\prod_{j=1}^{i}\lambda_{M_j}(\{M_1,\ldots,M_{j-1}\}) = D\sum_{\overline{M}_{i+1},\ldots,\overline{M}_J\in\mathcal{P}(\{M_{i+1},\ldots,M_J\})}\prod_{j=i+1}^{J}\big(\lambda_{\mathcal{C}(\{\overline{M}_j,\ldots,\overline{M}_J\})}\big)^{-1}.$$

We take

$$D = \prod_{j=1}^{J}\lambda_{M_j}(\{M_1,\ldots,M_{j-1}\}).$$

By the assignment condition, D is the same for all permutations of $M_1,\ldots,M_J$, and so it is the same for all choices of $M_1,\ldots,M_i$. We note that

$$\prod_{j=1}^{i}\lambda_{M_j}(\{M_1,\ldots,M_{j-1}\}) = D\Big(\prod_{j=i+1}^{J}\lambda_{M_j}(\{M_1,\ldots,M_{j-1}\})\Big)^{-1}.$$

Hence we need to show that:

$$\Big(\prod_{j=i+1}^{J}\lambda_{M_j}(\{M_1,\ldots,M_{j-1}\})\Big)^{-1} = \sum_{\overline{M}_{i+1},\ldots,\overline{M}_J\in\mathcal{P}(\{M_{i+1},\ldots,M_J\})}\prod_{j=i+1}^{J}\big(\lambda_{\mathcal{C}(\{\overline{M}_j,\ldots,\overline{M}_J\})}\big)^{-1}. \tag{15}$$

We prove (15) by induction starting from $i = J$ and going down to $i = 0$. For $i = J$ the products are empty, and there is nothing to prove. For $i = J - 1$, equation (15) says that $\lambda_{\mathcal{C}(\{M_J\})} = \lambda_{M_J}(\{M_1, \dots, M_{J-1}\})$, i.e. the only idle server M_J is activated at the rate of arrivals from $\mathcal{C}(\{M_J\})$, which is obviously true. We assume that (15) holds for $i+1$ and prove it for i.

$$
\begin{aligned}
&\sum_{\overline{M}_{i+1},\dots,\overline{M}_J \in \mathcal{P}(\{M_{i+1},\dots,M_J\})} \prod_{j=i+1}^{J} \left(\lambda_{\mathcal{C}(\{\overline{M}_j,\dots,\overline{M}_J\})}\right)^{-1} \\
&= \left(\lambda_{\mathcal{C}(\{M_{i+1}\dots,M_J\})}\right)^{-1} \sum_{k=i+1}^{J} \sum_{\overline{M}_{i+2},\dots,\overline{M}_J \in \mathcal{P}(\{M_{i+1},\dots,M_J\}\backslash M_k)} \prod_{j=i+2}^{J} \left(\lambda_{\mathcal{C}(\{\overline{M}_j,\dots,\overline{M}_J\})}\right)^{-1} \\
&= \left(\lambda_{\mathcal{C}(\{M_{i+1}\dots,M_J\})}\right)^{-1} \sum_{k=i+1}^{J} \Big(\prod_{j=i+2}^{J} \lambda_{\overline{M}_j}(\{M_1,\dots,M_i,M_k,\dots,\overline{M}_{j-1}\}) \Big)^{-1} \\
&= \left(\lambda_{\mathcal{C}(\{M_{i+1}\dots,M_J\})}\right)^{-1} \sum_{k=i+1}^{J} \lambda_{M_k}(\{M_1,\dots,M_i\}) \times \\
&\qquad\qquad \Big(\lambda_{M_k}(\{M_1,\dots,M_i\}) \prod_{j=i+2}^{J} \lambda_{\overline{M}_j}(\{M_1,\dots,M_i,M_k,\dots,\overline{M}_{j-1}\})\Big)^{-1} \\
&= \left(\lambda_{\mathcal{C}(\{M_{i+1}\dots,M_J\})}\right)^{-1} \sum_{k=i+1}^{J} \lambda_{M_k}(\{M_1,\dots,M_i\}) \Big(\prod_{j=i+1}^{J} \lambda_{M_j}(\{M_1,\dots,M_{j-1}\})\Big)^{-1} \\
&= \Big(\prod_{j=i+1}^{J} \lambda_{M_j}(\{M_1,\dots,M_{j-1}\})\Big)^{-1}.
\end{aligned}
$$

In the first equality we write the permutations of $\{M_{i+1}, \dots, M_J\}$ as starting with M_k, followed by a permutation of $\{M_{i+1}, \dots, M_J\}\backslash M_k$, and add them up over $k = i+1, \dots, J$ and we take out the common term of $\left(\lambda_{\mathcal{C}(\{M_{i+1}\dots,M_J\})}\right)^{-1}$. The second equality uses the induction hypothesis, for $M_1, \dots, M_i, M_k$. Note that here $M_{i+1} = M_k$ and for $j = i+2$ one needs to read $\overline{M}_{j-1} = M_{i+1} = M_k$. In the third equality we multiply and divide each summand by $\lambda_{M_k}(\{M_1, \dots, M_i\})$. In the fourth equality we then use the assignment condition (13), to rewrite all the products in the order $M_{i+1}, \dots, M_J$. In the final equality we take out the common term, and we then use equation (12) to see that the remaining terms equal 1. This completes the induction, and completes the proof of the theorem. □

We note that the results of this section are similar to those obtained for a skill based service Erlang loss system [2, 4]. It is shown in [4] that these loss systems under a random assignment policy which satisfies the same assignment condition (13), or under ALIS policy, share the same stationary behavior. This is proved in [4], Theorem 2, and the proof of that theorem is the same as the proof given here. We repeat the proof here, because we are using completely different notation.

2.7 Waiting time distribution

The waiting time distribution for a customer of type c that arrives at the system is derived in [24], for the random assignment policy. By Theorem 2 in the previous section, a customer of type c that arrives at the system when the system is governed by the FCFS-ALIS policy, will, under steady state, see exactly the same distribution of states as under the random assignment policy of [24]. As a consequence, the waiting time distribution for a customer of type c will be exactly the one derived in [24], Theorem 3. We thus quote this theorem here as for our FCFS-ALIS system.

Let $\mathcal{S}^i$ denote all the ordered subsets of servers of length i. Let $\pi(M_1, \cdot, \dots, M_i, \cdot)$ denote the steady state probability that servers $M_1, \dots, M_i$ are busy in that order, so that:

$$
\begin{aligned}
&\pi(M_1, \bullet, \dots, M_i, \bullet) = \\
&= \sum_{n_1,\dots,n_i=0}^{\infty} \sum_{(M_{i+1},\dots,M_i)\in\mathcal{P}(\mathcal{S}\setminus\{M_1,\dots,M_i\})} \pi(M_1, n_1, \dots, M_i, n_i, M_{i+1}, \dots, M_J) \\
&= B \prod_{j=1}^{i} \left(\mu_{\{M_1,\dots,M_j\}} - \lambda_{\mathcal{U}(\{M_1,\dots,M_j\})}\right)^{-1} \sum_{(M_{i+1},\dots,M_i)\in\mathcal{P}(\mathcal{S}\setminus\{M_1,\dots,M_i\})} \prod_{j=i+1}^{J} \lambda_{\mathcal{C}(\{M_j\dots,M_J\})}{}^{-1} \\
&= B \prod_{j=1}^{i} \left(\mu_{\{M_1,\dots,M_j\}} - \lambda_{\mathcal{U}(\{M_1,\dots,M_j\})}\right)^{-1} \Big(\prod_{j=i+1}^{J} \lambda_{M_j}(\{M_1, \dots, M_{j-1}\}) \Big)^{-1}. \qquad (16)
\end{aligned}
$$

The following theorem specifies the steady-state waiting time distribution of type c customer.

Theorem 3 *The LST of the steady-state waiting time W_c of a job of type c, under FCFS-ALIS policy is given by*

$$
\begin{aligned}
E(e^{-sW_{c\cdot}}) \;=\; & \sum_{i=0}^{J} \sum_{(M_1,\dots,M_i)\in\mathcal{S}^i} \pi(M_1, \bullet, \dots, M_i, \bullet) \times \\
& \prod_{\substack{j\,=\,1 \\ c\,\in\,\mathcal{U}(\{M_1,\dots,M_j\})}}^{i} \frac{\mu_{\{M_1,\dots,M_j\}} - \lambda_{\mathcal{U}(\{M_1,\dots,M_j\})}}{\mu_{\{M_1,\dots,M_j\}} - \lambda_{\mathcal{U}(\{M_1,\dots,M_j\})} + s}, \qquad (17)
\end{aligned}
$$

where $\pi(M_1, \bullet, \dots, M_i, \bullet)$ is given by (16).

This has the following interpretation: Consider the system in steady state. Jobs of type c arrive as a Poisson stream, and hence they see the queue in steady state, and find machines $(M_1, \dots, M_i)$ busy with probability $\pi(M_1, \cdot, \dots, M_i, \cdot)$. If some of the idle machines can process a job of type c, the arriving job will go into service immediately, and the waiting time will be 0. This is expressed in (17) by noting that in that case $c \notin \mathcal{U}(\{M_1, \dots, M_i\})$, and so the product is empty (and thus equal to 1). Otherwise the job will have to wait a sum of exponential waiting times as follows: let k be such that $c \notin \mathcal{C}(M_j)$ for $j = k+1, \dots, i$, but $c \in \mathcal{C}(M_k)$. Then there will be an exponential waiting time for each of the servers $k, \dots, i$. The waiting time for server M_j will be like an $M/M/1$ waiting time, i.e. it will be exponential with parameter $\mu_{\{M_1,\dots,M_j\}} - \lambda_{\mathcal{U}(\{M_1,\dots,M_j\})}$.

Hence, when a job of type c arrives, his waiting time can be interpreted as going through a tandem sequence of $M/M/1$ queues, until he can be served by the last of them. Note that this interpretation is not really what happens: while it is true that the job of type c has to wait until server k or one of the earlier servers $1, \dots, k-1$ will reach him, it is not the case that all the customers ahead of him will be served before him, and that he will be skipped by all the servers $k+1, \dots, i$. It is possible that server k will skip over all the other jobs and servers, and take the job of type c immediately, and it is also possible that one of the earlier servers $1, \dots, k-1$ will skip over all the intervening jobs and servers and take the job of type c. This is again one of those mysteries that one encounters from time to time in queueing theory.

3 The overloaded system

In this section we consider the system as before, with total arrival rate $\lambda = \sum_{c \in C} \lambda_c$ and total service rate $\mu = \sum_{j=1}^{J} \mu_{m_j}$. We introduce the following notations: the total traffic intensity $\rho = \lambda/\mu$, the fraction arrivals for the customer types $\alpha_c = \lambda_c/\lambda$ and the fraction service capacity for the servers $\beta_{m_j} = \mu_{m_j}/\mu$. Also, for subsets, $\alpha_{\mathcal{X}} = \lambda_{\mathcal{X}}/\lambda$, $\mathcal{X} \subseteq \mathcal{C}$, and $\beta_{\mathcal{Y}} = \mu_{\mathcal{Y}}/\mu$, $\mathcal{Y} \subseteq \mathcal{S}$.

It is convenient in this section to rewrite the stationary probabilities as:

$$\pi(\mathfrak{s}) = \tilde{B}(\rho) \prod_{j=1}^{i} \frac{\left(\rho \alpha_{\mathcal{U}(\{M_1,\dots,M_j\})}\right)^{n_j}}{\beta_{\{M_1,\dots,M_j\}}{}^{n_j+1}} \prod_{j=i+1}^{J} \left(\rho \alpha_{\mathcal{C}(\{M_j \dots, M_J\})}\right)^{-1} \tag{18}$$

with

$$\tilde{B}(\rho) = B/\mu^J = \left(\sum_{\mathcal{P}_J} \sum_{i=0}^{J} \Big(\prod_{j=1}^{i} \left(\beta_{\{M_1,\dots,M_j\}} - \rho \alpha_{\mathcal{U}(\{M_1,\dots,M_j\})}\right)^{-1} \prod_{j=i+1}^{J} \left(\rho \alpha_{\mathcal{C}(\{M_j,\dots,M_J\})}\right)^{-1} \Big) \right)^{-1}.$$

We will study what happens as the total traffic intensity increases. We will assume that α, β are fixed, μ is fixed, and the total arrival rate λ increases. Under these conditions, for $\rho > 1$ the system becomes unstable with some of the queues growing without bounds. We will discover that when some of the queues grow without bounds, the rest of the system stabilizes and has a stationary behavior, which is identical to that observed for FCFS matching of two infinite sequences (of servers and of customers) as discussed in [3]. We will distinguish a case of complete resource pooling and a case of incomplete resource pooling. For the latter we will find a unique decomposition of the system.

We state the infinite matching model of [3] here (see also [12]): There are two infinite sequences, a sequence of customers $c^1, c^2, \dots$ chosen i.i.d. from $\mathcal{C}$ with probabilities α, and a sequence of servers (or of services) $s^1, s^2, \dots$ chosen i.i.d. from $\mathcal{S}$ with probabilities β, and a bipartite graph of compatibilities. Servers and customers are matched on a FCFS basis. The process $X^I(N)$ (the subscript I stands for Infinite-matching) is a discrete time Markov chain that describes the state of this system after matching the first N servers, and it is defined as follows: After matching the first N servers to customers, the sequence of customers has an initial segment in which all are matched, followed by a middle segment in which some are matched and some are not, followed by a last infinite segment where none are matched. $X^I(N)$ describes this middle segment. The state is of the form $(M_1, n_1, \dots, M_{J-1}, n_{J-1}, M_J)$, where $M_1, \dots, M_J$ is a permutation of $\mathcal{S}$ that lists the last matched server of each type according to its order of appearance in the sequence of customers, and n_j is the number of unmatched customers between M_j and M_{j+1}. It is shown in [3] that $X^I(N)$ is an ergodic Markov chain and has a product form steady state distribution given by:

$$\pi^I(M_1, n_1, \dots, n_{J-1}, M_J) = B^I \prod_{j=1}^{J} \frac{\alpha_{\mathcal{U}(\{M_1,\dots,M_j\})}{}^{n_i}}{\beta_{\{M_1,\dots,M_j\}}{}^{n_i+1}}, \tag{19}$$

with

$$B^I = \left(\sum_{\mathcal{P}_J} \prod_{j=1}^{J-1} \left(\beta_{\{M_1,\dots,M_j\}} - \alpha_{\mathcal{U}(\{M_1,\dots,M_j\})}\right)^{-1} \right)^{-1}. \tag{20}$$

The analogy between X^I and our system is clear: The permutation of the last match of server of type j is the permutation of the servers m_j, and the unmatched customers correspond to the queues between the servers.

The rest of this section is structured as follows: In Section 3.1 we define complete resource pooling, under which the system is stable for all $\lambda < \mu$, and show that as $\lambda \nearrow \mu$ the steady state distribution of the system converges to that of $X^I(N)$. In Section 3.2 we study the fluid approximation to our queueing system. This contains information on the dynamics of the system which cannot be gleaned from its steady state distribution. The rest of the section deals with overloaded systems. In Section 3.3 we study the limiting behavior of the overloaded system under complete resource pooling. We show that while the last queue grows to infinity, the rest of the system converges in law to the steady state distribution of $X^I(N)$. In particular, while the last queue grow to infinity, the remaining queues remain well behaved, and the servers stay close together. In Section 3.4 we study a network maximal flow problem related to the stability of our system and derive a unique decomposition the system when there is incomplete resource pooling. In Section 3.5 we study the limiting behavior of the overloaded system under incomplete resource pooling.

3.1 Complete resource pooling

We say that the system satisfies *complete resource pooling* if the following three equivalent statements hold:

$$\begin{aligned} \beta_{\{M_1,\dots,M_i\}} &> \alpha_{\mathcal{U}(\{M_1,\dots,M_i\})}, & \{M_1,\dots,M_i\} &\neq \emptyset, \mathcal{S}, \\ \beta_{\{M_1,\dots,M_i\}} &< \alpha_{\mathcal{C}(\{M_1,\dots,M_i\})}, & \{M_1,\dots,M_i\} &\neq \emptyset, \mathcal{S}, \\ \alpha_{\{c_1,\dots,c_i\}} &< \beta_{\mathcal{S}(\{c_1,\dots,c_i\})}, & \{c_1,\dots,c_i\} &\neq \emptyset, \mathcal{C}. \end{aligned} \tag{21}$$

As we state in the next theorem, under complete resource pooling the system will be stable for all $\rho < 1$, and as $\rho \nearrow 1$ its steady state distribution will converge to the steady state distribution of $X^I(N)$, the process describing FCFS matching of two infinite sequences in [3].

We will use the notation (similar to (16)):

$$\begin{aligned} &\pi(M_1, n_1, \dots, M_j, n_j, M_{j+1}, \cdot, \dots, M_i, \cdot, M_{i+1}, \dots, M_J) \\ &\qquad\qquad = \sum_{n_{j+1},\dots,n_i=0}^{\infty} \pi(M_1, n_1, \dots, , M_i, n_i, M_{i+1}, \dots, M_J). \end{aligned}$$

Theorem 4 *Consider the system with fixed μ, α, β, and let λ vary. Assume that complete resource pooling holds.*

(i) The system is ergodic for all $\lambda < \mu$.

(ii) For states when not all servers are busy,

$$\lim_{\rho \nearrow 1} \pi(M_1, n_1, \dots, , M_i, n_i, M_{i+1}, \dots, M_J) = 0, \text{ for } i < J.$$

(iii) For states when all servers are busy

$$\lim_{\rho \nearrow 1} \pi(M_1, n_1, \dots, M_{J-1}, n_{J-1}, M_J, \cdot) = B^I \prod_{j=1}^{J} \frac{\alpha_{\mathcal{U}(\{M_1,\dots,M_j\})}{}^{n_i}}{\beta_{\{M_1,\dots,M_j\}}{}^{n_i+1}} = \pi^I(M_1, n_1, \dots, M_{J-1}, n_{J-1}, M_J),$$

where:

$$B^I = \tilde{B}(1) = \left(\sum_{\mathcal{P}_J} \prod_{j=1}^{J-1} \left(\beta_{\{M_1,\dots,M_j\}} - \alpha_{\mathcal{U}(\{M_1,\dots,M_j\})} \right)^{-1} \right)^{-1}.$$

(iv) The same results with the same limiting values hold also for the stationary distribuiton of the Markov chain of jumps.

Proof. We note that $\alpha_{\mathcal{U}(\{M_1,\dots,M_i\})} < \beta_{\{M_1,\dots,M_i\}}$ implies that $\lambda_{\mathcal{U}(\{M_1,\dots,M_i\})} < \mu_{\{M_1,\dots,M_i\}}$ whenever $\lambda < \mu$, and so the system is ergodic for all λ such that $\lambda < \mu$, by Theorem 1. This proves (i).

Next we observe that the value of $\tilde{B}(\rho)$ is:

$$\begin{aligned}\tilde{B}(\rho) \;=\; & \Bigg[\sum_{\mathcal{P}_J}\sum_{i=0}^{J-1}\Big(\prod_{j=1}^{i}\big(\beta_{\{M_1,\dots,M_j\}} - \rho\alpha_{\mathcal{U}(\{M_1,\dots,M_j\})}\big)^{-1}\prod_{j=i+1}^{J}\big(\rho\alpha_{\mathcal{C}(\{M_j,\dots,M_J\})}\big)^{-1}\Big) \\ & + \sum_{\mathcal{P}_J}\Big((1-\rho)^{-1}\prod_{j=1}^{J-1}\big(\beta_{\{M_1,\dots,M_j\}} - \rho\alpha_{\mathcal{U}(\{M_1,\dots,M_j\})}\big)^{-1}\Big)\Bigg]^{-1}.\end{aligned}$$

The term $(1-\rho)^{-1}$ is there because of $\beta_{\{M_1,\dots,M_J\}} = \alpha_{\mathcal{U}(\{M_1,\dots,M_J\})} = 1$. All the expressions inside the square brackets remain bounded as $\rho \nearrow 1$, except for $(1-\rho)^{-1}$ which tends to ∞. Hence:

$$\begin{aligned}&\lim_{\rho\nearrow 1}\tilde{B}(\rho) = 0,\\ &\lim_{\rho\nearrow 1}\tilde{B}(\rho)(1-\rho)^{-1} = \left(\sum_{\mathcal{P}_J}\prod_{j=1}^{J-1}\big(\beta_{\{M_1,\dots,M_j\}} - \rho\alpha_{\mathcal{U}(\{M_1,\dots,M_j\})}\big)^{-1}\right)^{-1} = B^I.\end{aligned}$$

For a single permutation $M_1,\dots,M_J$ with servers $M_1,\dots,M_i$ busy, $i < J$ we calculate:

$$\begin{aligned}\pi(M_1,\boldsymbol{\cdot},\dots,M_i,\boldsymbol{\cdot},M_{i+1},\dots,M_J) &= \sum_{n_1,\dots,n_i=0}^{\infty}\pi(M_1,n_1,\dots,M_i,n_i,M_{i+1},\dots,M_J)\\ &= \tilde{B}(\rho)\prod_{j=1}^{i}(\beta_{\{M_1,\dots,M_j\}} - \rho\alpha_{\mathcal{U}(\{M_1,\dots,M_j\})})^{-1}\prod_{j=i+1}^{J}(\rho\alpha_{\mathcal{C}(\{M_j\dots,M_J\})})^{-1}.\end{aligned}$$

All terms in this expression except $\tilde{B}(\rho)$ remain bounded, and hence the whole expression tends to 0 as $\rho \nearrow 1$. This proves (ii).

On the other hand, when all J servers are busy we calculate:

$$\begin{aligned}\pi(M_1,n_1,\dots,M_{J-1},n_{J-1},M_J,\boldsymbol{\cdot}) &= \sum_{n_J=0}^{\infty}\pi(M_1,n_1,\dots,M_J,n_J)\\ &= \tilde{B}(\rho)\prod_{j=1}^{J-1}\frac{\big(\rho\alpha_{\mathcal{U}(\{M_1,\dots,M_j\})}\big)^{n_j}}{\beta_{\{M_1,\dots,M_j\}}{}^{n_j+1}}(1-\rho)^{-1}\end{aligned}$$

where we again used $\beta_{\{M_1,\dots,M_J\}} = \alpha_{\mathcal{U}(\{M_1,\dots,M_J\})} = 1$. As $\rho \nearrow 1$, $\tilde{B}(\rho)(1-\rho)^{-1} \to B^I$, and we obtain (iii).

To prove (iv) we note that the transition rates of the process are bounded between λ and $\lambda+\mu$, and so the jump chain and the continuous process are both ergodic or both non-ergodic at the same time. Furthermore, all the stationary probabilities of states when not all the servers are busy will tend to zero for the jump chain as well as for the continuous process, as $\rho \nearrow 1$. Finally, the transition rate at which jumps occur when all the servers are busy is $\lambda+\mu$, independent of the state. Hence the stationary probabilities for the jump chain and the continuous process, for states when all servers are busy, will tend to the same limits as $\rho \nearrow 1$. This proves (iv). □

3.2 Fluid limits of the FCFS queueing system with multitype customers and servers

The steady state distribution of a Markovian system provides complete information about long term performance measures, and is therefore very useful. However, it does not provide any information about the dynamics of the system. The dynamics of a Markovian system are of interest when the system is stable, as they provide information about its short term behavior, and for unstable systems it provides information on the transient behavior of the system. Exact analysis of the dynamics is almost always intractable, however the dynamics can be approximated by studying fluid or diffusion approximations to the system. In this section we study the fluid approximation of our system. For an introduction to the study of these fluid limits see [11, 13, 14].

We introduce a different notation for our Markov process. We let $X(t) = (S(t), Q_1(t), \dots, Q_J(t))$. Here $S(t) = (M_1(t), \dots, M_J(t))$ is a permutation of $\mathcal{S}$. The total number of customers in the system is $|Q(t)| = \sum Q_j(t)$, ordered in order of arrival with Q_1 earlier than Q_2 earlier than Q_3, and so on. Of the queues at time t the first $k(t)$ queues are non-empty, the remaining are empty. Server $M_j(t)$ is serving the first of the customers of $Q_j(t)$ (the $1 + \sum_{i<j} Q_i(t)$ customer in the queue), where $j = 1, \dots, k(t)$, and servers $M_j(t)$, $j = k(t)+1, \dots, J$ are idle, ordered by increasing idle time. The customer in service at $M_j(t)$ is of course of type $c \in \mathcal{C}(M_j)$. The remaining waiting customers in $Q_j(t)$ are all of them of types in $\mathcal{U}(M_1(t), \dots, M_j(t))$.

We also introduce the following processes that give an alternative description of the system. Assume each customer upon arrival is given a number which counts its position in the arrival stream. We let $A(t)$ be the total number of arrivals by time t, where the last arrival was numbered $A(t)$. We also let $Y_1(t) < \cdots < Y_J(t)$ be the positions of the servers in the sequence of customers up to time t, so that $Y_i(t)$ is the number in the sequence of the customer currently served by the ith server, which is server $M_i(t)$. If $k(t) < J$ servers are busy we will let the positions of the idle servers $M_{k(t)+1}(t), \dots, M_J(t)$ be $(Y_{k(t)+1}, \dots, Y_J(t)) = (A(t)+1, \dots, A(t)+J-k(t))$. We denote $Y(t) = (Y_1(t), \dots, Y_J(t))$. Note that $A(t), Y_1(t), \dots, Y_J(t)$ are all monotone non-decreasing in t. We shall let $A(0), Y_1(0), \dots, Y_J(0) \geq 0$ be some initial state, not necessarily empty.

The fluid scaling of an arbitrary function $z(t)$ is denoted $\bar{z}^r(t) = z(rt)/r$. Let $z(t, \omega)$ be a stochastic process with paths in D^d (real vector functions in the d dimensional Euclidean space, which are right continuous with left limits). Let ω denote a fixed sample path, and let r be a divergent sequence of integers. If $\bar{z}^r(t, \omega)$ converges u.o.c (uniformly on compacts) to a deterministic function $\bar{z}(t)$ as $r \to \infty$ then we call $\bar{z}(t)$ a fluid limit.

We shall for simplicity take $\lambda + \mu = 1$ and think of our system as powered by a Poisson process with rate 1. Each event of the process is either an arrival or a service completion. An arrival of a customer of type c occurs with probability $\lambda\alpha_c$. A service completion by server m_j occurs with probability $\mu\beta_{m_j}$. A customer departs at a service completion only if server m_j is not idle. This process is the primitive building block of our system and it obeys the FSLLN (fucnctional strong law of large numbers). All our statements about fluid limits will hold, for every sample path of the Poisson processes of arrivals of the various customer types and service completions of the various servers, for which FSLLN convergence holds.

The dynamics of our system are described by the system process $Z(t) = (S(t), Q(t), A(t), Y(t), T(t))$ where $T(t) = (T_1(t), \dots, T_J(t))$ are the actual (non-idling) cumulative processing times on servers $m_1, \dots, m_J$. Fluid limits of components of $Z(t)$ exist, as shown, for example, by Dai and Lin [14]. Fluid limits of $T(t)$ exist, because of the equi-continuity of the fluid scalings of $T(t)$. This implies also that fluid limits of $Q(t), Y(t)$ exist. A small addition to the arguments of Dai and Lin is needed for our system, since in standard queueing networks, queues change at each completion of service by at most 1, while in our system the change in the corresponding components of Q and Y is a random integer. However these random integers are geometrically distributed random variables, with probability of success $\geq \min_c \alpha_c > 0$, so by the law of large numbers we retain continuity of $\bar{Q}$ and $\bar{Y}$ as a function of $\bar{T}$, and in fact both $\bar{Q}(t), \bar{Y}(t)$ are Lipschitz continuous. In particular they are differentiable almost everywhere and they are equal to the integrals of their derivatives. We say that t is a regular point of the fluid limit if all components of $\bar{Q}(t), \bar{Y}(t), \bar{T}(t)$ are differentiable at t.

Note that $S(t)$ is not included in the fluid limits, since it is a random permutation, and while fluid acceleration of time i.e. $S(rt)$ makes some sense, fluid scaling of S, i.e $S(rt)/r$ makes no sense. We can however trace the behavior of S under fluid scaling: for all t, $Y_1(t) < \cdots < Y_J(t)$, so $S(t)$ is uniquely defined. In the limit however we only have $\bar{Y}_1(t) \leq \bar{Y}_2(t) \leq \cdots \leq \bar{Y}_J(t)$. So we define $\bar{S}(t)$ as an ordered partition of the servers, where for example we will have $\bar{S}(t) = (M_1, \{M_2, M_3\}, M_4, \ldots)$ if $\bar{Y}_1(t) < \bar{Y}_2(t) = \bar{Y}_3(t) < \bar{Y}_4(t) < \cdots$. We will on occasion write $\bar{S}(t) = (S_1, S_2, \ldots, S_L)$ with $(S_1, S_2, \ldots, S_L)$ a partition of $\mathcal{S}$, even when we have more detailed information on the order of servers within some of the subsets S_i.

We will now study the dynamics of the fluid limits. Let $\bar{Z}(t)$ be a fluid limit, and assume that it starts from a fluid initial state $\bar{Z}(0)$. Of course, if we start with a finite initial state $Z(0)$, then $Z(r\,0)/r \to 0$ as $r \to \infty$, but one can assume a sequence of starting states such that $\bar{Z}(0) \neq 0$, and because our system is Markovian, this sequence of initial states is irrelevant. We do however assume that $\bar{Z}(0)$ is obtained from feasible states, i.e. given $S(0)$, the contents of $Q_i(t)$ include only customers from $\mathcal{U}(\{M_1(0), \ldots, M_i(0)\})$. In the following propositions we study the behavior of the components of $\bar{Z}(t)$. For example we know, by the FSLLN that $\bar{A}(t) = \lambda t$ almost surely.

In addition to $\bar{A}(t) = \lambda t$ we have the following immediate relation between $\bar{Y}(t)$ and $\bar{Q}(t)$:

Proposition 5 *The following relation holds almost surely and for all regular t:*

$$\begin{array}{rcll} \bar{Q}_i(t) & = & \big(\bar{Y}_{i+1}(t) - \bar{Y}_i(t)\big)\alpha_{\mathcal{U}(\{M_1,\ldots.M_i\})}, & \\ \frac{d}{dt}\bar{Q}_i(t) & = & \Big(\frac{d}{dt}\bar{Y}_{i+1}(t) - \frac{d}{dt}\bar{Y}_i(t)\Big)\alpha_{\mathcal{U}(\{M_1,\ldots.M_i\})}, & i = 1, \ldots, J-1. \\ \bar{Q}_J(t) & = & \bar{A}(t) - \bar{Y}_J(t) & \\ \frac{d}{dt}\bar{Q}_J(t) & = & \lambda - \frac{d}{dt}\bar{Y}_J(t) & \end{array} \tag{22}$$

Proof. We note that while $Y_{i+1} - Y_i$ counts *all* arrivals from the position of server M_i to the position of server M_{i+1} at time t, $Q_i(t)$ consists only of those customers which have not been processed by $M_{i+1}, \ldots, M_J$, so conditional on $Y_{i+1}(t) - Y_i(t)$, $Q_i(t)$ is a Binomial random variable, $Q_i(t) \sim \text{Binomial}\big(Y_{i+1}(t) - Y_i(t), \alpha_{\mathcal{U}(\{M_1,\ldots,M_i\})}\big)$. Under the fluid scaling we get

$$\bar{Q}^r_i(t) \to \big(\bar{Y}_{i+1}(t) - \bar{Y}_i(t)\big)\alpha_{\mathcal{U}(\{M_1,\ldots,M_i\})},$$

from which the first part of (22) follows, the second part is immediate at all regular points. The equations for the fluid queue after the last server follow similarly. $\square$

We now study the dynamics of $\bar{Y}(t)$. Our first result examines the dynamics of the fluid system when $\bar{Q}_i(t) > 0$ for all $i = 1, \ldots, J$, or equivalently, $\bar{Y}_1(t) < \bar{Y}_2(t) < \cdots < \bar{Y}_J(t)$.

Proposition 6 *Consider a fluid limit for which $\bar{Y}_1(t) < \bar{Y}_2(t) < \cdots < \bar{Y}_J(t)$, and let $\bar{S}(t) = (M_1, \ldots, M_J)$. Then almost surely at all regular points:*

$$\frac{d}{dt}\bar{Y}_i(t) = \mu \frac{\beta_{M_i}}{\alpha_{\mathcal{U}(\{M_1,\ldots,M_i\})} - \alpha_{\mathcal{U}(\{M_1,\ldots,M_{i-1}\})}}, \quad i = 1, \ldots, J. \tag{23}$$

Proof. Consider a sample path ω and sequence $r \to \infty$ such that $\bar{Z}^r(t, \omega)$ converges to $\bar{Z}$. Assume that the sample path ω obeys FSLLN and that t is a regular point. We will now show that for such a sample path, the fluid limit satisfies (23). Since every fluid limit is obtained from some ω and some $r \to \infty$, this implies that every fluid limit satisfies (23) with probability one for almost all t. This preliminary framework is standard, and will be used in all subsequent propositions on fluid limits.

Because $\bar{Y}_i$ are continuous, there exists $\Delta > 0$ such that for all $\tau \in (t-\Delta, t+\Delta)$ the order of $\bar{Y}_i$ is unchanged, $\bar{Y}_1(\tau) < \bar{Y}_2(\tau) < \cdots < \bar{Y}_J(\tau)$, and so for r large enough, for $\tau \in (t-\Delta/2, t+\Delta/2)$, $\bar{Y}_1^r(\tau) < \bar{Y}_2^r(\tau) < \cdots < \bar{Y}_J^r(\tau)$, and so we have, for large enough r that $Y_1(s) < Y_2(s) < \cdots < Y_J(s)$, $s \in I = (rt - r\Delta/2, rt + r\Delta/2)$. In particular this means that servers do not overtake each other, and the order of the servers is given by the permutation $S(s) = (M_1, \dots, M_J)$ and it is unchanged over $s \in I$. Also, by (22), $\bar{Q}_i(s) > 0$, $i = 1, \dots, J-1$, for $s \in I$.

Consider now the movement of server M_i, during $s \in I$. Since $\bar{Q}_i(t) > 0$, we have that $Q_i(s) > 0$ for $s \in I$, and so server M_i will be busy all the time. Hence during the time interval I it will complete a total of L services, where $L \sim \text{Poisson}(\mu\beta_{M_i} r\Delta)$. Server M_i will serve customers which are in $\mathcal{C}(M_i)$, and which have not been served by any of the servers $M_{i+1}, \dots, M_J$, i.e it will serve customers in $\mathcal{U}(\{M_1, \dots, M_i\}) \backslash \mathcal{U}(\{M_1, \dots, M_{i-1}\})$. Hence, it will skip all customers which are not in $\mathcal{U}(\{M_1, \dots, M_i\}) \backslash \mathcal{U}(\{M_1, \dots, M_{i-1}\})$, and so at each service completion it will move a random number of places G in the sequence of customers, where G is a geometric random variable with probability of success $\alpha_{\mathcal{U}(\{M_1,\dots,M_i\})} - \alpha_{\mathcal{U}(\{M_1,\dots,M_{i-1}\})}$. Hence, the total change in Y_i over the interval I will be:

$$Y_i(rt + r\Delta/2) - Y_i(rt - r\Delta/2) = \sum_{l=1}^{L} G_l, \text{ with } G_l \text{ i.i.d distributed like } G.$$

Hence:

$$\bar{Y}_i^r(t + \Delta/2) - \bar{Y}_i^r(t - \Delta/2) = \frac{\sum_{l=1}^{L} G_l}{r},$$

which by Wald's equation and the FSLLN converges as $r \to \infty$ to

$$\bar{Y}_i(t + \Delta/2) - \bar{Y}_i(t - \Delta/2) = \Delta\mu\beta_{M_i} \frac{1}{\alpha_{\mathcal{U}(\{M_1,\dots,M_i\})} - \alpha_{\mathcal{U}(\{M_1,\dots,M_{i-1}\})}}$$

from which (23) follows. □

Proposition 6 clarifies how single isolated servers move in the fluid limits. The next proposition studies movement of servers which stay together in the fluid limit.

Proposition 7 *Consider a fluid limit for which $\bar{Y}_{k-1}(\tau) < \bar{Y}_k(\tau) = \cdots = \bar{Y}_l(\tau) < \bar{Y}_{l+1}(\tau)$ for some $k < l$ and for all $\tau \in (t-\Delta, t+\Delta)$. Let $\bar{S}(\tau) = (S', \{M_k, \dots, M_l\}, S'')$ for the same range of τ, where S', S'' are the subsets of servers preceding and succeeding $M_k, \dots, M_l$ (their order may be known, but it is irrelevant here). Then:*

$$\frac{d}{dt}\bar{Y}_i(t) = \mu \frac{\beta_{\{M_k,\dots,M_l\}}}{\alpha_{\mathcal{U}(\{M_1,\dots,M_l\})} - \alpha_{\mathcal{U}(\{M_1,\dots,M_{k-1}\})}}, \quad i = k, \dots, l. \tag{24}$$

Proof. We consider the processes $Y_k(s), \dots, Y_l(s)$, $s \in I = (rt - r\Delta/2, rt + r\Delta/2)$, and the fluid scaling, $\bar{Y}_k^r(\tau), \dots, \bar{Y}_l^r(\tau)$, $\tau \in (t - \Delta/2, t + \Delta/2)$. As before, because $\bar{Q}_{k-1}(\tau) > 0$, $\bar{Q}_l(\tau) > 0$, for r large enough these processes move in isolation from the other Y_j during $s \in I$, and they consist of the movement of the fixed set of servers $S = \{M_k, \dots, M_l\}$. Note that these servers may change their order many times during the time interval I. For r large enough we have that $\bar{Y}_k^r(t - \Delta/2) \approx \cdots \approx \bar{Y}_l^r(t - \Delta/2)$, and also $\bar{Y}_k^r(t + \Delta/2) \approx \cdots \approx \bar{Y}_l^r(t + \Delta/2)$, where we can choose r large enough that $\approx$ will be a distance which is negligible relative to Δ.

The servers in S are working all the time, so they will process a total of $L \sim \text{Poisson}(\mu\beta_{\{M_k,\dots,M_l\}} r\Delta)$. They will all start approximately at the same place, and end up approximately at the same place, processing all the customers of types in $\mathcal{U}(\{M_1,\dots,M_l\})\backslash\mathcal{U}(\{M_1,\dots,M_{k-1}\}$, and skipping all the other customers, between their approximately common starting and ending positions. The total distance travelled by all the processors in S will therefore be approximately equal to $\sum_{l=1}^{L} G_l$ where G_l are again i.i.d. geometric random variables with probability of success $\alpha_{\mathcal{U}(\{M_1,\dots,M_l\})} - \alpha_{\mathcal{U}(\{M_1,\dots,M_{k-1}\})}$.

Thus, proceeding to the limit as in the proof of Proposition 6 we get that

$$\bar{Y}_i(t-\Delta/2) - \bar{Y}_i(t+\Delta/2) = \Delta\mu \frac{\beta_{\{M_k,\dots,M_l\}}}{\alpha_{\mathcal{U}(\{M_1,\dots,M_l\})} - \alpha_{\mathcal{U}(\{M_1,\dots,M_{k-1}\})}}$$

and (24) follows. $\square$

For the next proposition we will make use of the following elementary lemma, the proof of which may be found in [15]

Lemma 1 *Let $g(t)$ be an absolutely continuous nonnegative function on $t \geq 0$ and let $\dot{g}(t)$ denote its derivative whenever it exists.*

(i) If $g(t) = 0$ and $\dot{g}(t)$ exists, then $\dot{g}(t) = 0$.

(ii) Assume the condition that for some $\epsilon > 0$, whenever $g(t) > 0$ and $\dot{g}(t)$ exists, then $\dot{g}(t) < -\epsilon$. Then $g(t) = 0$ for all $t > \delta$ where $\delta = g(0)/\epsilon$. Furthermore $g(\cdot)$ is nonincreasing and hence, once it reaches zero, it stays there forever.

Proposition 8 *Assume that complete resource pooling holds, and that we start from $\bar{Q}_i(0) = 0$, $i = 1,\dots,J-1$, $\bar{Q}_J(0) > 0$. Then for some $\Delta > 0$ we will have $\bar{Q}_i(t) = 0$, $i = 1,\dots,J-1$, and $\frac{d}{dt}\bar{Y}_i(t) = \mu$, $i = 1,\dots,J$, for $0 < t < \Delta$.*

Proof. By continuity of $\bar{Q}$ we can find $\Delta > 0$ such that $\bar{Q}_J(t) > 0$, $0 < t < \Delta$, and so during $0 < t < \Delta$ all servers will be busy. We wish to show that all the servers move at the same rate. We will show that indeed, if $\bar{Q}_i(t) > 0$ at some $0 < t < \Delta$, then $\frac{d}{dt}\bar{Q}_i(t) < -\epsilon < 0$ which by Lemma 1 implies that $\bar{Q}_i(t) = 0$, $i = 1,\dots,J-1$ for $0 < t < \Delta$. Applying formula (24) to $\mathcal{S}$ we then get $\frac{d}{dt}\bar{Y}_i(t) = \mu$ for $0 < t < \Delta$. We proceed in three stages.

(i) We show that it is not possible for the servers to break into two groups, each of which is moving together. Assume we have for all $0 < t < \Delta$ that $\bar{Y}_1(t) = \dots = \bar{Y}_i(t)$ and $\bar{Y}_{i+1}(t) = \dots = \bar{Y}_J(t)$, and assume that for some time t in the interval, $\bar{Q}_i(t) > 0$. Let $S_1 = \{M_1(t),\dots,M_i(t)\}$ and $S_2 = \{M_{i+1}(t),\dots,M_J(t)\}$. Specializing Proposition 7 to the sets S_1 and S_2 we have:

$$\begin{aligned} \frac{d}{dt}\bar{Y}_j(t) &= \mu\frac{\beta_{S_1}}{\alpha_{\mathcal{U}(S_1)}}, \quad j = 1,\dots,i, \\ \frac{d}{dt}\bar{Y}_j(t) &= \mu\frac{\beta_{S_2}}{\alpha_{\mathcal{C}(S_2)}}, \quad j = i+1,\dots,J. \end{aligned}$$

By complete resource pooling, $\beta_{S_1} > \alpha_{\mathcal{U}(S_1)}$ while $\beta_{S_2} < \alpha_{\mathcal{C}(S_2)}$. In other words, the front part of the split into S_1, S_2 moves at rate $< \mu$ while the back part moves at rate $> \mu$. As a result we have that $\frac{d}{dt}\bar{Q}_i(t) < 0$. In fact, looking at all possible splits we can find $\epsilon > 0$ such that for any S_1, S_2 as above, $\frac{d}{dt}\bar{Q}_i(t) < -\epsilon < 0$. Hence, by Lemma 1, if the sets of servers S_1 move together and the set of servers S_2 move together, then starting with $\bar{Q}_i(0) = 0$ we will have that $\bar{Q}_i(t) = 0$ for all $0 < t < \Delta$.

(ii) Assume now that the servers split into more subsets which move together. Let S_1 be as before, but S_2 splits into $S_3,\dots,S_L$. While S_1 moves behind, the remaining servers will move together in subsets, with servers of S_3 moving at the slowest rate, behind S_4, and so on, and servers S_L moving at the fastest rate ahead of all the other subsets. Service by these subsets of servers will split into serving the following disjoint subsets of customer types:

$$\mathcal{C}(S_2) = \Big(\mathcal{U}(S_1 \cup S_3) \backslash \mathcal{U}(S_1)\Big) \cup \Big(\mathcal{U}(S_1 \cup S_3 \cup S_4) \backslash \mathcal{U}(S_1 \cup S_3)\Big) \cup \cdots \cup \Big(\mathcal{C}(S_2) \backslash \mathcal{U}(S_1 \cup \cdots \cup S_{L-1})\Big)$$

Note that if we regard $S_2, \mathcal{C}(S_2)$ as a system on its own, we do not know that it has complete resource pooling, hence we need indeed consider the possibility that the servers of S_2 will split up. If they do split up, then we will have that these subsets of processors will move at rates

$$\mu \frac{\beta_{S_1}}{\alpha_{\mathcal{U}(S_1 \cup S_3) \backslash \mathcal{U}(S_1)}} < \mu \frac{\beta_{S_2}}{\alpha_{\mathcal{U}(S_1 \cup S_3 \cup S_4) \backslash \mathcal{U}(S_1 \cup S_3)}} < \cdots < \mu \frac{\beta_{S_L}}{\alpha_{\mathcal{C}(S_2) \backslash \mathcal{U}(S_1 \cup \cdots \cup S_{L-1})}}.$$

By the elementary inequality:

$$\frac{a}{b} < \frac{c}{d} \Longleftrightarrow \frac{a}{b} < \frac{a+c}{b+d} \Longleftrightarrow \frac{a+c}{b+d} < \frac{c}{d},$$

we then have that

$$\mu \frac{\beta_{S_1}}{\alpha_{\mathcal{U}(S_1 \cup S_3) \backslash \mathcal{U}(S_1)}} < \mu \frac{\beta_{S_2}}{\alpha_{\mathcal{C}(S_2)}}.$$

It then follows by comparing with part (i) of the proof that the servers in subset S_1 move faster than those in S_3, and so again we get that $\frac{d}{dt}\bar{Q}_i(t) < -\epsilon < 0$.

(iii) What we have seen so far is that if the servers are split, then the last subset of servers will catch up with the one before last. We now proceed step by step: Because of what we showed, if $\bar{Q}_1(t) > 0$ then $\frac{d}{dt}\bar{Q}_1(t) < -\epsilon < 0$, and so we conclude that if $\bar{Q}_1(0) = 0$ then it will stay 0 for all $0 < t < \Delta$. We then move to $\bar{Q}_2$. Because $\bar{Q}_1(t) = 0$, servers M_1, M_2 move together. Hence, again by part (ii), $\frac{d}{dt}\bar{Q}_2(t) < -\epsilon < 0$, and we conclude that also $\bar{Q}_2(t) = 0$ for all $0 < t < \Delta$. Repeating this step for the 3rd queue, 4th queue and so on, we conclude that $\bar{Q}_i(t) = 0$, $i = 1, \ldots, J-1$ for all $0 < t < \Delta$ as required. □

Complete resource pooling is also necessary for all the servers to move together as the next Proposition shows:

Proposition 9 *Assume that there is no complete resource pooling. Assume we start from $\bar{Q}_i(0) = 0$, $i = 1, \ldots, J-1$, $\bar{Q}_J(0) > 0$. Then immediately the set of servers will split into more than one subset, which will move at different rates.*

Proof. Assume to the contrary that all servers move together. Then by Proposition 8, for all servers $\frac{d}{dt}\bar{Y}_i(t) = \mu$, $0 < t < \Delta$.

If there is no resource pooling then there is a subset of servers S such that $\beta_S < \alpha_{\mathcal{U}(S)}$. We will show that for any time t some of the servers in S will move at a rate which is $< \mu$. This will prove the proposition.

Assume first that the servers in S move together, and are behind all other servers. Then the rate at which they move will be, by Proposition 7, $\frac{d}{dt}\bar{Y}_i(t) = \mu \frac{\beta_S}{\alpha_{\mathcal{U}(S)}} < \mu$. Assume next that the servers in S split into $S = S_1 \cup \cdots \cup S_L$ subsets, each of which moves together, and that all these subsets are behind all the servers in $\overline{S}$. Then as argued in the proof of Proposition 8, the servers of the last subset S_1 will move at a rate slower than $\mu \frac{\beta_S}{\alpha_{\mathcal{U}(S)}} < \mu$.

Finally, if these subsets of S which move together are not behind all the servers in $\overline{S}$, then the servers of the last subset S_1 will have more customers to serve than if they were moving in the very back. Hence the rate of moving for $i \in S_1$ will be:

$$\frac{d}{dt}\bar{Y}_i(t) \le \frac{\beta_{S_1}}{\alpha_{\mathcal{U}(S_1)}} \le \frac{\beta_S}{\alpha_{\mathcal{U}(S)}} < \mu.$$

□

We now extend the results of Propositions 8, 9 to any subset of servers, to be able to say when $\bar{Y}_k, \dots, \bar{Y}_l$ might be moving together. To do so we need the following definition of relative complete resource pooling, as follows:

Definition 10 *Consider a partition of the servers into subsets S', S, S''. We say that S has complete resource pooling between S' and S'' (the order of S' before S'' is important here), if the subsystem which consists of servers $m_i \in S$, and the customer types $c \in \mathcal{U}(S' \cup S) \backslash \mathcal{U}(S')$, with $\tilde{\beta}_{m_i} = \beta_{m_i}/\beta_S$, $\tilde{\alpha}_c = \alpha_c/\alpha_{\mathcal{U}(S' \cup S)\backslash\mathcal{U}(S')}$ has complete resource pooling.*

Proposition 11 *Consider a fluid limit for which $\bar{Y}_{k-1}(t) < \bar{Y}_k(t) = \cdots = \bar{Y}_l(t) < \bar{Y}_{l+1}(t)$ for some $k < l$ and for some t. Let $\bar{S}(t) = (S', \{M_k, \dots, M_l\}, S'')$ be the corresponding partition of the servers. Then $\bar{Y}_k, \dots \bar{Y}_l$ will continue to move together, i.e. $\bar{Y}_k(\tau) = \cdots = \bar{Y}_l(\tau)$ for $t < \tau < t + \Delta$ for some $\Delta > 0$ if and only if $\{M_1, \dots, M_k\}$ have complete resource pooling between S' and S''.*

Proof. Because $\bar{Q}_{k-1}(t) > 0$ and $\bar{Q}_l(t) > 0$ the servers $S = \{M_k, \dots, M_l\}$ will move in isolation of all the other servers during a time interval $t < \tau < t + \Delta$, and will be between servers S' and S''. In their movement they will process all the customers in $\mathcal{U}(S \cup S')\backslash\mathcal{U}(S')$, and only those customers. The condition that S has complete resource pooling between S' and S'' means that in a system that would consist only of servers S and customer types $\mathcal{U}(S \cup S')\backslash\mathcal{U}(S')$, with $\tilde{\beta}, \tilde{\alpha}$ as defined above, there would be complete resource pooling. Hence, by Propositions 8, 9, it would be necessary and sufficient for the servers of these subsystems to move together. But the movement of the servers S when they are between S', S'' and $\bar{Q}_{k-1}(t) > 0$ and $\bar{Q}_l(t) > 0$, is exactly as if they were a separate system, with the only difference that they will actually also skip over all the customers types in $\mathcal{C}(S'')$. Hence, by Propositions 8, 9 they will stay together if and only if S has complete resource pooling between S' and S''. □

3.3 The overloaded system under complete resource pooling

We consider now the behavior of the system under complete resource pooling, when $\lambda > \mu$. Here clearly the Markov process is transient. However, what we will see is that while the queue behind the last server grows without bound, the servers and the queues between them tend to a limiting distribution which is again that of the FCFS infinite matching model. We will use the notation and the results on the fluid dynamics from Section 3.2. We will also need the following lemma, the proof of which is given in [1].

Lemma 2 *Let $X(n) = (X_1(n), X_2(n))$ be a Markov chain on countable state space with $X_i(n) \in \mathbb{Z}^+$. Assume the following:*

1. $\lim_{n\to\infty} X_2(n) = \infty$ *almost surely.*
2. $P(X_1(n+1) = j | X_1(n) = i,\ X_2(n) = l) = P_{i,j}$, *for all values of $l > 0$, where $P_{i,j}$ are transition probabilities of an ergodic Markov chain with stationary probabilities π_j.*

Then for all initial i_0, j_0:

$$\sup_j \Big| P\big(X_1(n) = j \,|\, X_1(0) = i_0, X_2(0) = j_0\big) - \pi_j \Big| \to 0, \ \text{as } n \to \infty$$

i.e. $X_1(n)$ converges in distribution to π in total variation norm.

We can now prove the following theorem:

Theorem 12 *Assume complete resource pooling and $\lambda > \mu$.*

(i) *From any fixed initial state, as $t \to \infty$:*

$$\begin{aligned} &k(t) \to J \quad a.s. \\ &Q_J(t)/t \to (\lambda - \mu) \quad a.s. \end{aligned} \tag{25}$$

(ii) *As $t \to \infty$, $P(S(t) = (m_1, m_2, \ldots, m_J), Q_1(t) = n_1{+}1, \ldots, Q_{J-1}(t) = n_{J-1}{+}1)$ converges to $\pi^I(m_1, n_1, \ldots, n_{J-1}, m_J)$ in total variation distance, where π^I is the stationary distribution of $X^I(N)$, the Markov chain describing the FCFS infinite matching model, given in (19), (20)*

Proof. We consider first the fluid limits for the overloaded system. By the results of the previous section, we have that $\frac{d}{dt}\bar{Y}_J(t) \le \mu$. This will hold, because by Proposition 7 the front subset of servers S will move at a rate $\mu\frac{\beta_S}{\alpha_{\mathcal{C}(S)}}$, and by complete resource pooling, $\beta_S < \alpha_{\mathcal{C}(S)}$. Since $\bar{Q}_J(t) = \bar{A}(t) - \bar{Y}_J(t)$ we have:

$$\frac{d}{dt}\bar{Q}_J(t) = \frac{d}{dt}\bar{A}(t) - \frac{d}{dt}\bar{Y}_J(t) = \lambda - \mu\frac{\beta_S}{\alpha_{\mathcal{C}(S)}} \ge \lambda - \mu > 0.$$

Hence $\bar{Q}_J(t) \to \infty$ as $t \to \infty$, and then of course $Q_J(t)$ will diverge almost surely. In particular this implies that $k(t) \to J$ almost surely as $t \to \infty$.

Consider now the behavior of our Markovian system when $Q_J(t) > 0$. When $Q_J(t) > 0$, all the servers are busy, and the queues $Q_j(t), j = 1, \ldots, J-1$ have transitions which occur irrespective of the current state at times which are a Poisson process of rate μ. The sequence of states following each transition form a discrete time process, with Markovian transition probabilities which are exactly those of the FCFS infinite matching model, and do not depend on the value of $Q_J(t)$. Hence, the conditions of Lemma 2 are fulfilled, and the discrete time jump process of states will converge in law to π^I. As a result the continuous time process will also converge in law to π^I.

Because $Q_1(t), \ldots, Q_{J-1}(t)$ converge to a steady state distribution, $\bar{Q}_1, \ldots, \bar{Q}_{J-1}$ converge almost surely to 0. Hence, in the fluid limit all the servers will move together at rate μ and for any fixed initial state, $\bar{Q}_J(t) = (\lambda - \mu)t$. Hence, $Q_J(t)/t \to \lambda - \mu$ almost surely. $\square$

3.4 Unique decomposition under incomplete resource pooling

We again consider the system with fixed α, β, μ and let λ increase, but we now consider the case that complete resource pooling does not hold. We show that there exists a unique decomposition of the system when it is overloaded. To do so we associate with our system the following network (see Fig. 7): The nodes are $c \in \mathcal{C}$, $m_j \in \mathcal{S}$, a source node o, and a sink node t. The arcs are (o, c) of capacity λ_c for all $c \in \mathcal{C}$, (m_j, t) of capacity μ_{m_j} for all $m_j \in \mathcal{S}$, and (c, m_j) of infinite capacity for all (c, m_j) in the bipartite compatibility graph.

In what follows we use the terms, notation, and results as formulated in Ford and Fulkerson's book [16], pages 1–14, see also [9]. In a directed network with origin and terminal o, t, a cut is given by a partition of the nodes into two sets, one of which includes o and the other includes t. For our network it is given by $\{o, \overline{C}, S\}$ and $\{t, C, \overline{S}\}$, for some subset of customer types C and some subset of servers S, where $\overline{C}$, $\overline{S}$ are the complements of C, S. The capacity of the cut is the sum of the capacities of arcs directed from the o part to the t part. For our network this will be the sum of the capacities of the arcs from o to the nodes in C, the arcs from the nodes of S to t, and the arcs from $\overline{C}$ to $\overline{S}$. For any cut with finite capacity, C and S must be such that

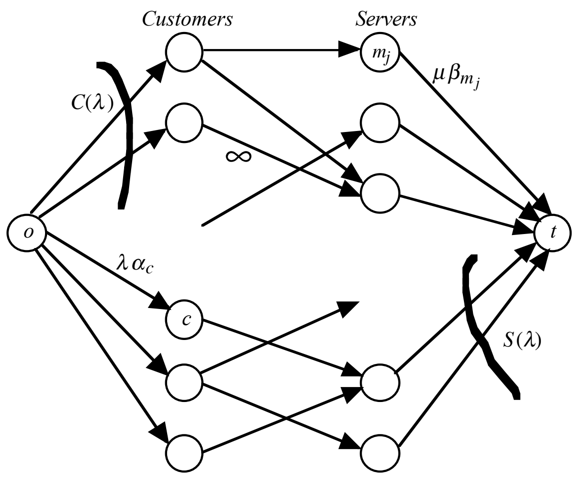

Figure 7: s–t maximal flow problem with minimum cut through $C(\lambda)$, $S(\lambda)$.

there are no arcs in the compatibility graph from $\overline{C}$ to $\overline{S}$. We will only consider cuts with finite capacity, for which the arcs from the o to the t part of the network are exactly the arcs from the o to C and the arcs from S to t. Rather than talk about the cut as a partition we will describe a cut by the sets C and S. Note that for cuts with finite capacity we will have that $S(\overline{C}) \subseteq S$, and that $C(\overline{S}) \subseteq C$. The celebrated max-flow min-cut theorem states that the maximal o to t flow through the network equals the capacity of the minimal cut.

The maximal flow in the network is related to the stability of our system through the following proposition.

Proposition 13 *A necessary condition for stability is that the maximal flow from o to t is λ. A sufficient condition is that there exists $\epsilon > 0$ such that if the capacities of the arcs (o, c) are increased to $(1+\epsilon)\lambda_c$ the maximal flow is $(1+\epsilon)\lambda$.*

Proof. The proof is similar to proofs that were given in [7, 12], we give it here for completeness. For sufficiency, assume that with o to c flow capacities $(1+\epsilon)\lambda_c$, the maximal flow in the network is $(1+\epsilon)\lambda$. Then for any subset of customer types C, the total flow from o to the nodes in C equals $(1+\epsilon)\lambda_C$, so the total flow from C to the server nodes $\mathcal{S}(C)$ equals at least $(1+\epsilon)\lambda_C$ (it may be more since $\mathcal{S}(C)$ may receive flow from additional customer nodes), and so the total flow from server nodes $\mathcal{S}(C)$ to t equals at least $(1+\epsilon)\lambda_C$. But the capacity of the arcs from server nodes $\mathcal{S}(C)$ to t equals $\mu_{\mathcal{S}(C)}$, so if the maximal flow is $(1+\epsilon)\lambda$ we have $\mu_{\mathcal{S}(C)} \geq (1+\epsilon)\lambda_C > \lambda_C$. Hence the condition (9) for stability of the queueing system with total arrival rate λ holds, and the system is stable.

For necessity, assume that the total arrival rate of the queueing system is λ. Then the maximal flow in the network must be $\leq \lambda$. Assume now that the maximal flow is $< \lambda$. By the max-flow min-cut theorem there exists a minimal cut of capacity $< \lambda$. Let such a minimal cut be defined by the arcs from o to C and the arcs from S to t. Then as observed above, $\mathcal{S}(\overline{C}) \subseteq S$. The capacity of the cut is $\lambda_C + \mu_S < \lambda$. We now see that $\mu_{\mathcal{S}(\overline{C})} \leq \mu_S < \lambda - \lambda_C = \lambda_{\overline{C}}$. But $\mu_{\mathcal{S}(\overline{C})} < \lambda_{\overline{C}}$ contradicts the condition (9) for the stability of the system. Thus if the queueing system is stable, then the maximal flow in the network must equal λ. $\square$

The following theorem describes the solution of the o to t maximum network flow problem, as a function of λ. Fig. 8 illustrates this theorem as well as the following Corollary 15.

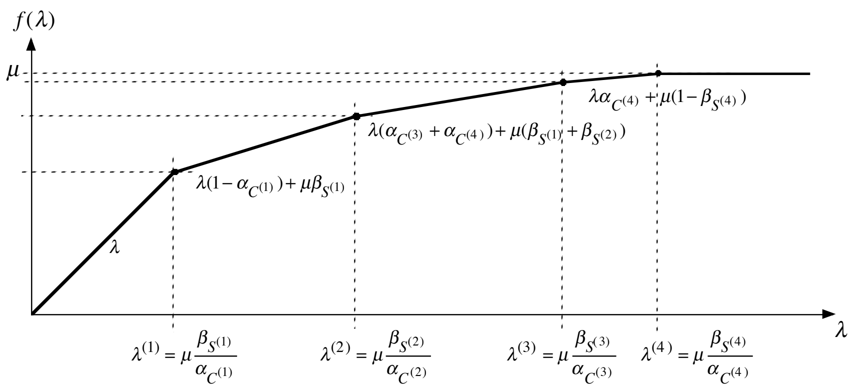

Figure 8: Maximal flow as a function of λ

Theorem 14 *Consider the o–t maximum network flow problem as λ increases. Then:*

(i) The maximal flow $f(\lambda)$ is a continuous piecewise linear non-decreasing concave function of λ, with breakpoints $0 = \lambda^{(0)} < \lambda^{(1)} < \cdots < \lambda^{(L)} < \lambda^{(L+1)} = \infty$, which has slope 1 for $0 < \lambda < \lambda^{(1)}$, and is constant and equal to μ for $\lambda > \lambda^{(L)}$.

(ii) For each interval $(\lambda^{(i-1)}, \lambda^{(i)})$ there exist a set of customer types $C(\lambda^{(i-1)}, \lambda^{(i)})$ and a set of servers $S(\lambda^{(i-1)}, \lambda^{(i)})$ such that they form a cut, which is the unique minimal cut for all λ in the interval.

(iii) The sets $C(\lambda^{(i-1)}, \lambda^{(i)})$ are decreasing in i and the sets $S(\lambda^{(i-1)}, \lambda^{(i)})$ are increasing in i, in the sense that: $C(\lambda^{(0)}, \lambda^{(1)}) \supset C(\lambda^{(1)}, \lambda^{(2)}) \supset \cdots \supset C(\lambda^{(L)}, \lambda^{(L+1)})$ and $S(\lambda^{(0)}, \lambda^{(1)}) \subset S(\lambda^{(1)}, \lambda^{(2)}) \subset \cdots \subset S(\lambda^{(L)}, \lambda^{(L+1)})$.

Proof. (i) The maximal flow problem for each fixed λ is a linear program with feasible and bounded solutions, and when it is considered with varying λ it is a parametric linear program. As such it will have intervals in which the same basis is optimal, and such intervals will cover the whole line of $\lambda > 0$. Within such an interval the flows of the optimal solution will be affine functions of λ. The optimal maximal flow objective $f(\lambda)$ is clearly a continuous non-decreasing function of λ. Consider now the optimal flows for λ' and λ'' with $\lambda' < \lambda''$ and look at $\lambda = (1-\theta)\lambda' + \theta\lambda''$. The convex combination of the optimal flows for λ' and for λ'' is a feasible flow for λ with objective value $(1-\theta)f(\lambda') + \theta f(\lambda'')$ which can only be suboptimal. This proves the concavity. Finally, if $0 < \lambda < \min\{\mu_{m_1}, \ldots, \mu_{m_J}\}$ the maximal flow is λ, so the slope of the initial interval of $f(\lambda)$ is 1, and for λ such that $\min_{c\in C} \lambda_c > \mu$ the maximal flow is μ, so the slope of $f(\lambda)$ in the last half infinite interval is 0.

(ii) Consider an interval $\lambda^{(i-1)} < \lambda < \lambda^i$ in which the maximal flow is $f(\lambda) = a + b\lambda$. For fixed λ_0 in the interval, consider a minimum cut, so its capacity is $a+b\lambda_0$. For any other λ in the interval the capacity of this cut will an affine function of λ, and it will be at least $f(\lambda)$ because any cut capacity is an upper bound on the flow. This implies that the capacity of the cut is equal to $a+b\lambda$ for all $\lambda^{(i-1)} < \lambda < \lambda^i$, and hence the cut is a minimal cut for all $\lambda^{(i-1)} < \lambda < \lambda^i$. Hence we have shown that any minimal cut for λ_0 is in fact a minimal cut for the whole range of values $\lambda^{(i-1)} < \lambda < \lambda^i$.

Assume now that there are two different minimal cuts, given by the partitions $\{o, \overline{C}_1, S_1\}$, $\{t, C_1, \overline{S}_1\}$ and $\{o, \overline{C}_2, S_2\}$, $\{t, C_2, \overline{S}_2\}$. The capacity of these minimal cuts will be $\lambda\alpha_{C_1} + \mu\beta_{S_1}$ and $\lambda\alpha_{C_2} + \mu\beta_{S_2}$ respectively, where both expressions are equal to $a+b\lambda$. By Corollary 5.4 in [16], the cut formed by $C_1 \cap C_2$ and $S_1 \cup S_2$ will also be a minimal cut, with capacity $\lambda\alpha_{C_1\cap C_2} + \mu\beta_{S_1\cup S_2}$. Recall that $\alpha_c > 0$ for all customer types $c \in \mathcal{C}$. Hence we cannot have equal capacities for the three cuts for a range of values of λ unless $C_1 = C_2$. Once we have that $C_1 = C_2$, we see that we cannot have equality of the capacities of the three cuts unless also $S_1 = S_2$. This proves the uniqueness in each interval.

(iii) We consider $\lambda' < \lambda''$. Let the minimal cut for λ' be given by subsets C, S, where the arcs across the cut are the arcs from o to C, and the arcs from S to t. We partition the network into two subnetworks, the first consists of $\{o, t, \mathcal{C}_1, \mathcal{S}_1\}$ where $\mathcal{C}_1 = \overline{C}$, $\mathcal{S}_1 = S$, and the second consists of $\{o, t, \mathcal{C}_2, \mathcal{S}_2\}$, where $\mathcal{C}_2 = C$, $\mathcal{S}_2 = \overline{S}$. We look at the solution of the o–t maximal network flow problem for each of the subnetworks, where the inflows to the two networks are increased by a factor of λ''/λ'. For the first subnetwork the maximal flow remains constant for all $\lambda'' > \lambda'$ and equal to $f_1 = \mu\beta_S$, and the minimal cut consists of the arcs from S_1 to t. By part (ii) we have a unique minimal cut also for the second subnetwork problem, given by unique sets $C_3 \subseteq C_2$ and $S_3 \subseteq S_2$ and the maximal flow equals the capacity of this cut $f_2 = \lambda''\alpha_{C_3} + \mu\beta_{S_3}$. We now claim that using the combined maximal flows of these two networks will give us the maximal flow for the original full network, with total arrival rate λ''. To see this we note the following: (1) The combined flow is a feasible flow for total arrival rate λ'', and it equals the sum of the flows $f_1(\lambda_1) + f_2(\lambda_2)$, which equals the capacities of the two minimal cuts. (2) The sets of nodes $\{o, C_1 \cup (C_2 \backslash C_3), S_1 \cup S_3\}$ and $\{t, C_3, S_2 \backslash S_3\}$ provide a partition and are a cut. (3) There are no links from C_1 or from $C_2 \backslash C_3$ to $S_2 \backslash S_3$, and so the capacity of this cut is finite, and equal to the capacities of the arcs o to C_3 and $S_1 \cup S_2$ to t, which is $f_1(\lambda_1) + f_2(\lambda_2)$. Thus we have a feasible flow equal to a cut capacity, which implies optimality.

We have shown that the minimal cut for λ'' is given by C_3 and $S_1 \cup S_3$, while the minimal cut for λ' is given by C_2 and S_1, so that $C_3 \subseteq C_2$, and $S_1 \cup S_3 \supseteq S_1$, which is the monotonicity we needed to show. Strict monotonicity holds if λ', λ'' belong to different intervals. $\square$

Theorem 14 induces a decomposition of the servers and of the customer types as detailed in the next Corollary. The decomposition is illustrated in Fig. 9. In this figure we have minimal cuts which belong to a case where there are 4 breakpoints in $f(\lambda)$, which correspond to 5 intervals, and 5 minimal cuts as numbered.

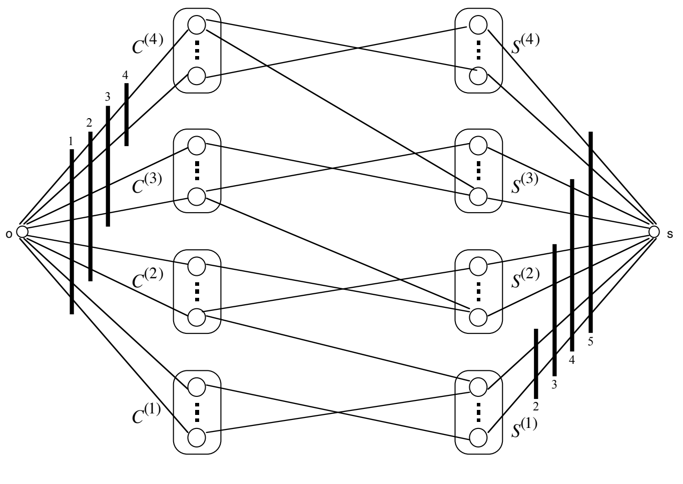

Figure 9: Decomposition of servers and of customer types

Corollary 15 *If the maximal flow $f(\lambda)$ has breakpoints $0 = \lambda^{(0)} < \lambda^{(1)} < \cdots < \lambda^{(L)} < \lambda^{(L+1)} =$*

∞, then there is a unique partition of the set of servers into non-empty subsets $\mathcal{S}^{(1)}, \dots, \mathcal{S}^{(L)}$ and of the set of customer types into non-empty subsets $\mathcal{C}^{(1)}, \dots, \mathcal{C}^{(L)}$, such that:

(i) The minimal cut for $\lambda^{(i-1)} < \lambda < \lambda^{(i)}$ consists of

$$\mathcal{C}(\lambda^{(i-1)}, \lambda^{(i)}) = \bigcup_{k \geq i} \mathcal{C}^{(k)}, \qquad \mathcal{S}(\lambda^{(i-1)}, \lambda^{(i)}) = \bigcup_{k < i} \mathcal{S}^{(k)},$$

for $i = 1, 2, \dots, L+1$.

(ii) The maximal flow $f(\lambda)$ where $\lambda^{(i-1)} < \lambda < \lambda^{(i)}$, is:

$$f(\lambda) = \lambda \sum_{k \geq i} \alpha_{\mathcal{C}^{(k)}} + \mu \sum_{k < i} \beta_{\mathcal{S}^{(k)}},$$

for $i = 1, \dots, L+1$.

(iii) The values of the breakpoints are:

$$\lambda^{(1)} = \mu \frac{\beta_{\mathcal{S}^{(1)}}}{\alpha_{\mathcal{C}^{(1)}}} < \lambda^{(2)} = \mu \frac{\beta_{\mathcal{S}^{(2)}}}{\alpha_{\mathcal{C}^{(2)}}} < \cdots < \lambda^{(L)} = \mu \frac{\beta_{\mathcal{S}^{(L)}}}{\alpha_{\mathcal{C}^{(L)}}}.$$

(iv) Consider the subsystem composed of servers $\mathcal{S}^{(i)}$ and customer types $\mathcal{C}^{(i)}$, with arrival rates $\lambda\alpha_c$ for customers of type c. Then this system is stable for $\lambda < \lambda^{(i)}$ and unstable for $\lambda \geq \lambda^{(i)}$.

(v) Consider the subsystem composed of servers $\bigcup_{k=i}^{L} \mathcal{S}^{(k)}$, and customer types $\bigcup_{k=i}^{L} \mathcal{C}^{(k)}$, with arrival rates $\lambda\alpha_c$ for customers of type c. Then this system is stable for $\lambda < \lambda^{(i)}$ and unstable for $\lambda \geq \lambda^{(i)}$.

(vi) For $\lambda^{(i-1)} < \lambda < \lambda^{(i)}$ the maximal flow solution of the whole system will have zero flow on arcs from customers $c \in \mathcal{C}^{(k)}$ to servers $m_j \in \mathcal{S}^{(l)}$ for all $k \geq i > l$.

Proof. Parts (i) and (ii) and (iii) follow directly from the construction of minimal cuts in Theorem 14. Parts (iv) and (v) then follow from Theorem 13 and the expression in (iii). Finally, for part (vi) we note that when $\lambda^{(i-1)} < \lambda < \lambda^{(i)}$ then the servers $\mathcal{S}^{(j)}$ receive flow $\mu\beta_{\mathcal{S}^{(j)}}$ from customer types $\mathcal{C}^{(j)}$ for $j = 1, \dots, i-1$, so there can be no additional flow to any of them from nodes of customer types in $\mathcal{C}^{(i)}, \dots, \mathcal{C}^{(L)}$. □

The following corollary gives another way of solving the max plow problem and decomposing the sets of servers and customer types, and pinpoints the nature of this decomposition:

Corollary 16 *The sets $\mathcal{C}^{(i)}, \mathcal{S}^{(i)}$ have the following characterization:*

$$\mathcal{C}^{(i)} = \arg\min_{C \subseteq \mathcal{C} \setminus \bigcup_{k<i} \mathcal{C}^{(k)}} \frac{\beta_{\mathcal{S}(C)}}{\alpha_C}, \qquad \mathcal{S}^{(i)} = \mathcal{S}(\mathcal{C}^{(i)}).$$

Proof. One way to see this is that, by Corollary 15 (iv), for each $C \subset \mathcal{C}^{(i)}$ we have $\frac{\beta_{\mathcal{S}(C)}}{\alpha_C} < \frac{\beta_{\mathcal{S}^{(i)}}}{\alpha_{\mathcal{C}^{(i)}}}$, and by (iv) $\frac{\beta_{\mathcal{S}^{(i)}}}{\alpha_{\mathcal{C}^{(i)}}}$ are monotone increasing in i. □

Intuitively the picture is as follows: Under complete resource pooling $\mathcal{C}^{(1)} = \mathcal{C}$, $\mathcal{S}^{(1)} = \mathcal{S}$. When there is no resource pooling, for some subsets of customers the requirement that $\alpha_C < \beta_{\mathcal{S}(C)}$ is violated. As λ increases the subset of customers which have the least value of $\beta_{\mathcal{S}(C)}/\alpha_C$ becomes overloaded when λ reaches $\mu\frac{\beta_{\mathcal{S}(C)}}{\alpha_C}$, and this defines $\mathcal{C}^{(1)}$ and $\mathcal{S}^{(1)}$. This leaves servers

$\mathcal{S}\backslash\mathcal{S}^{(1)}$ to serve the remaining customer types $\mathcal{C}\backslash\mathcal{C}^{(1)}$. Note that even if $c \in \mathcal{C}\backslash\mathcal{C}^{(1)}$ can be served by a server in $\mathcal{S}^{(1)}$, this will not happen in the max flow solution, since all the servers in $\mathcal{S}^{(1)}$ are fully occupied by $\mathcal{C}^{(1)}$. The remaining servers and customer types now behave like a subsystem with $\tilde{\alpha}_c = \frac{\alpha_c}{1-\alpha_{\mathcal{C}^{(1)}}}$, $\tilde{\beta}_{m_j} = \frac{\beta_{m_j}}{1-\beta_{\mathcal{S}^{(1)}}}$. If in the remaining subsystem $\frac{\tilde{\beta}_{\mathcal{S}(C)}}{\tilde{\alpha}_C} \geq 1$ for all subsets, then $L = 2$ and the minimum of these ratios will be reached by $\mathcal{S}\backslash\mathcal{S}^{(1)}$, $\mathcal{C}\backslash\mathcal{C}^{(1)}$. Else, if the minimum is again < 1, the subsets $C, \mathcal{S}(C)$ with minimal ratio of $\beta_{\mathcal{S}(C)}/\alpha_C$ will be $\mathcal{S}^{(2)}, C^{(2)}$, and the servers in $\mathcal{S}^{(2)}$ will become overloaded when $\lambda = \mu\frac{\beta_{\mathcal{S}^{(2)}}}{\alpha_{C^{(2)}}}$, and so on (see Figs. 8 and 9).

3.5 Limiting behavior of overloaded system under incomplete resource pooling

Following our study of the fluid limits of our system, and the decomposition of the set of servers and of customer types when there is no resource pooling, we can now describe the limiting behavior of our system as $t \to \infty$ in the case that there is no resource pooling, as a function of the total arrival rate λ. In the following theorem we use the notation developed in Section 3.4.

Theorem 17 *Assume the system has incomplete resource pooling, as in Section 3.4 and that the total arrival rate is $\lambda^{(l)} < \lambda < \lambda^{(l+1)}$. Then as $t \to \infty$ the following convergences will occur for the state of the system, $\mathfrak{s} = (S(t), Q_1(t), \dots, Q_J(t))$:*

(i) The permutation $S(t)$ as $t \to \infty$ will consist of a permutation of $S^{(1)}$ followed by a permutation of $S^{(2)}$ and so on up to a permutation of $S^{(l)}$, followed by a permutation of the remaining servers.

(ii) As $t \to \infty$ the queue between the last server of $S^{(k)}$ and the first server of $\mathcal{S}^{(k+1)}$, for $k = 1, \dots, \min\{l, L-1\}$ will diverge, growing at a rate $\mu\left(\frac{\beta_{\mathcal{S}^{(k+1)}}}{\alpha_{\mathcal{C}^{(k+1)}}} - \frac{\beta_{\mathcal{S}^{(k)}}}{\alpha_{\mathcal{C}^{(k)}}}\right)\alpha_{\bigcup_{j\leq k}\mathcal{C}^{(j)}}$.

If $l = L$, i.e. $\lambda > \lambda_L$, the queue after the last server will grow at rate: $\mu\left(\lambda - \frac{\beta_{\mathcal{S}^{(L)}}}{\alpha_{\mathcal{C}^{(L)}}}\right)$.

(iii) For $k = 1, \dots, l$, the probability distribution of the permutation of $S^{(k)}$ and the queue length between the servers $\mathcal{S}^{(k)}$ will converge to the stationary distribution of the FCFS infinite matching model for the subsystem of $\mathcal{C}^{(k)}, \mathcal{S}^{(k)}$.

(iv) If $l < L$, then the probability distribution of the permutation of the remaining servers and the queue lengths between them and behind the last of them, and the ordered set of idle servers, will converge to the steady state distribution of the stable system consisting of $\bigcup_{k>l}\mathcal{C}^{(k)}, \bigcup_{k>l}\mathcal{S}^{(k)}$, as given by Theorem 1.

The diverging queues will diverge almost surely. The convergence of the probabilities to stationary probabilities will be in total variation distance. The overloaded subsystems and the remaining stable system will converge in distribution to independent processes.

Proof. It follows from the results of Sections 3.2 and 3.4 that the fluid limits of the processes Y_k for the various servers will eventually coalesce into subsets which will move together, with the servers of the subsets $S^{(k)}$ moving together at rates increasing with k for $k = 1, \dots, l$, and the remaining servers will move together with $A(t)$ at the rate λ. This implies (i). The rates in (ii) are obtained directly from (22) and (24).

It is then seen that for each subsystem the transition rates, conditional on the diverging queues being > 0 are exactly those of the FCFS matching model for the subsystems $k = 1, \dots, l$, and for the remaining subsystem they are equal to those of a stable system consisting of $\bigcup_{k>l}\mathcal{C}^{(k)}, \bigcup_{k>l}\mathcal{S}^{(k)}$. Parts (iii) and (iv) then follow by Lemma 2.

The independence follows since the various subsystems have independent transitions given that the diverging queues are > 0. $\square$

4 The overloaded system with abandonments

We now consider our system with the added feature of abandonments. We assume that customers of type c have patience distribution F_c which we take for simplicity to be absolutely continuous, and a customer abandons the system when his waiting time before service exceeds his patience. These abandonments assure the stability of our system. We study the behavior of the system under uniform acceleration, where we assume that the arrival rate λ and the service rate μ increase, while the patience distribution remains constant.

In a seminal paper Talreja and Whitt [23] have studied a system with multi-type customers and multi-type servers under FCFS, in the presence of abandonments and under uniform acceleration. Our results in this paper, with the added assumption of memoryless arrivals and services, allow us to give a more complete description with more details on its behavior.

While exact analysis of systems with abandonments is difficult, their asymptotic description under uniform acceleration is much simpler. For a single stream of customers with abandonments, under uniform acceleration, if the offered load is < 1 the speed with which customers move means that there are very few abandonments and in the limit they disappear. When the system is overloaded, with offered load > 1, there are many customers with sufficient patience in the system at all times, and successive customers will have approximately the same waiting time, so that there will be an effective cutoff time w, such that customers with patience less than w will not be served, while customers with patience $\geq w$ will in the limit be served after waiting exactly w. These results are obtained by studying the fluid and diffusion approximations of the process. Similar results were obtained also for multi-type customers, by Jennings and Reed [22]. The case of offered load close to 1 is much harder, and leads to the Halfin-Whitt limiting regime.

It is to be expected that similar behavior under uniform acceleration also occurs in our multi-type server multi-type customer FCFS system with abandonments. In what follows we do not prove the asymptotic behavior under uniform acceleration, and we do not consider the case of offered load close to 1. We assume that the asymptotic results hold, and under this assumption we derive the behavior of our system.

We now discuss four cases in order of increasing complexity, in the limit under uniform acceleration:

Uniform patience, complete resource pooling

Assume all customers have the same patience distribution, $F_c = F$ and the system has complete resource pooling. If the system is overloaded then a value w will be uniquely determined by $\mu = \lambda(1 - F(w))$. This will be the effective arrival rate of customers that do not abandon, the system will be stable, and include at any time μw customers which have enough patience to be served, while losing customers that abandon at a rate $\lambda F(w)$. Note that all the different types of customers will have the same fraction of abandonments, and those customers that will be served will have the same waiting time before service. Talreja and Whitt [23] refer to this as *global first come first served.*

The system will then evolve on two time scales. On the fluid scale it will have a large $\sim \mu w$ number of customers, all of them behind the last server, and all the servers will move together at rate μ. On the detailed scale the servers will behave like servers in the FCFS infinite matching model, with the servers ordered according to a random permutation, with stable queues between them, and the permutation of servers and the queues between the servers will have the steady state distribution π^I.

Uniform patience, incomplete resource pooling

Assume common patience distribution F and no resource pooling, so that the max flow as function of λ and the resulting system decomposition are as described in Section 3.4. If the system is overloaded so that $\lambda > \lambda^{(L)}$ and we go to uniform acceleration, by increasing both λ and μ, then all the servers will be overloaded, and the system will split into L subsystems, with $\mathcal{S}^{(i)}$ serving only customers of $\mathcal{C}^{(i)}$. There will be individual w_i for $i = 1, \dots, L$, where w_i is determined by $\lambda\alpha_{\mathcal{C}^{(i)}}(1 - F(w_i)) = \mu\beta_{\mathcal{S}^{(i)}}$. Here the fraction of abandonment and the waiting times are the same within each subsystem, but they are different for each of the subsystems, with $w_1 > w_2 > \cdots > w_L$.

Now each subsystem evolves independent of the others, similar to the behavior of the whole system when resource pooling holds.

If $\lambda^{(l)} < \lambda < \lambda^{(l+1)}$, then only subsystems $1, \dots, l$ will be overloaded, and under uniform acceleration subsystems $l+1, \dots, L$ will form one subsystem which is stable, with no abandonments in the limit. The steady state distribution of this subsystem will be that given by Theorem 1.

Individual patience distribution, complete resource pooling

Assume now that the patience distribution of customers of type c is F_c, and that complete resource pooling holds. If the system is overloaded, because of complete resource pooling there will be global first come first served, and the waiting time cutoff w will be determined as the unique value that satisfies:

$$\lambda \sum_{c\in\mathcal{C}} \alpha_c(1 - F_c(w)) = \mu$$

That w is unique follows from the monotonicity of all the F_c. Now under uniform acceleration all the customers that have patience greater or equal to w will be served after waiting exactly w. However, different customer types will have different abandonment fractions, given by $F_c(w)$ for customers of type c.

Again the system will evolve on two time scales, with a queue of length $\sim \mu w$, and all the servers moving together on the fluid scale, and on the detailed scale the servers and customers between them will have steady state distribution given by π^I, which however needs to be modified as follows: because different types have different fractions that abandon, the new i.i.d. distribution of the types of customers that get served is given by $\tilde{\alpha}_c = \alpha_c(1 - F_c(w))\lambda/\mu$.

It is however possible that with the new values of $\tilde{\alpha}_c$ the system will not have complete resource pooling. This leads to the behavior of the next, most general case.

Individual patience distribution, incomplete resource pooling

In that case the system will decompose into several subsystems, each of which will behave independently of the others. One of the subsystems may not be overloaded, and under uniform acceleration will behave like our stable queueing network of Section 2. All the other subsystems will evolve as overloaded systems with individual patience distribution, and complete resource pooling.

Within the overloaded subsystems the following will be true: Each subsystem $\mathcal{S}^{(i)}$, $\mathcal{C}^{(i)}$ will have its own w_i, satisfying:

$$\lambda \sum_{c \in \mathcal{C}^{(i)}} \alpha_c (1 - F_c(w_i)) = \mu \beta_{\mathcal{S}^{(i)}},$$

and the subsystem of $\mathcal{S}^{(i)}$, $\mathcal{C}^{(i)}$ will have complete resource pooling with the modified

$$\tilde{\alpha}_c = \alpha_c (1 - F_c(w_i)) \frac{\lambda}{\mu \beta_{\mathcal{S}^{(i)}}}, \quad c \in \mathcal{C}^{(i)}.$$

It remains to show that this decomposition is unique, and how to obtain it. We proceed as follows: We start from $w = \infty$, only customers with infinite patience will be served, so everyone abandons, and the effective arrival rate is 0. We then decrease w. As a result, $\lambda(w) = \sum_{c \in C} \lambda \alpha_c (1 - F_c(w))$ which is the effective arrival rate, of customers that will be served if we serve only those with patience $\geq w$, will increase. For each such value of w we will also have the fractions of customers of each type, which will be proportional to $\tilde{\alpha}_c = \alpha_c(1 - F_c(w_i))$. As w decreases and the arrival rate increases we will get a first set of servers which is overloaded, $\mathcal{S}^{(1)}$ and a value w_1. With it we have $\mathcal{C}^{(1)} = \mathcal{U}(\mathcal{S}^{(1)})$. We now exclude $\mathcal{S}^{(1)}$,$\mathcal{C}^{(1)}$, and continue in the same way for the remaining servers and customer types, where we now continue to decrease w further beyond w_1. This procedure will yield a decomposition, and a sequence of waiting times $w_1 > w_2 > \cdots$. It will end either when all the servers and customer types have been put into subsets, with a last, smallest w_L, in which case all the servers are overloaded, or when some servers and customer types are still left and w has reached the value 0, in which case these remaining servers are not overloaded, and can serve all their customer types even without abandonments.

The subsystems will then evolve independently as for the previous three cases. Within each subsystem there will be global FCFS but the fraction that abandon will be different for different customer types, given by the individual $F_c(w_i)$. On the detailed scale the overloaded subsystems will have steady state distribution π^I with the appropriate parameters. The last subsystem which is not overloaded will have in the uniform acceleration limit no abandonments, and have steady state distribution as in Theorem 1.

References

[1] Adan, I.J.B.F., Foss, S., Weiss, G. (2012) Local stability of a transient Markov chain, working paper.

[2] Adan, I.J.B.F., Hurkens, C., Weiss, G. (2010) A reversible Erlang loss system with multitype customers and multitype servers. *Probability in Engineering and Informational Sciences* 24:535–548.

[3] Adan, I.J.B.F., Weiss, G. (2011) Exact FCFS matching rates for two infinite multi-type sequences. *Operations Research* to apear.

[4] Adan, I.J.B.F., Weiss, G. (2011) A loss system with skill based servers under assign to longest idle server policy *Probability in Engineering and Informational Sciences* to apear.

[5] Adan, I.J.B.F., Wessels, J., Zijm, W.H.M. (1989) Queuing analysis in a flexible assembly system with a job-dependent parallel structure. *Operations Research Proceedings 1988* Springer-Verlag, Berlin, pp. 551–558.

[6] Adan, I.J.B.F., Wessels, J., Zijm, W.H.M. (1991) Flexible assembly and shortest queue problems. *Modern production concepts, theory and applications*, G. Fandel, G. Zaepfel (eds.), Springer-Verlag, Berlin, pp. 644–659.

[7] Adan, I. , Foley, R. and McDonald, D. (2009) Exact Asymptotics of the Stationary Distribution of a Markov Chain: a Production Model. *Queueing Systems* 62: 311–344.

[8] Aerts, J., Korst, J., Verhaegh, W., (2007) Load balancing for redundant storage strategies: Multiprocessor scheduling with machine eligibility. *Journal of Scheduling* 4: 245–257.

[9] Ravindra K. Ahuja, Thomas L. Magnanti and James B. Orlin (1993) *Network Flows: Theory, Algorithms, and Applications* Prentice Hall, NJ.

[10] Aksin, Z., Armony, M., Mehrotra, V., (2007) The modern call-center, a multi-disciplinary perspective on operations management research. *Production and Operations Management* 16(6): 665–688.

[11] M. Bramson (2008) *Stability of Queueing Networks.* Springer.

[12] Caldentey, R., Kaplan, E.H., Weiss, G., (2009) FCFS infinite bipartite matching of servers and customers. *Advances in Applied Probability* 41:695–730.

[13] J. G. Dai (1995) On positive Harris recurrence of multiclass queueing networks: a unified approach via fluid limit models. *Annals of Applied Probability*, 5(1):49–77.

[14] J. G. Dai and W. Lin (2005) Maximum pressure policies in stochastic processing networks. *Operations Research*, 53(2):197–218.

[15] Dai, J. and Weiss, G. (1996) Stability and Instability of Fluid Models for certain Re-Entrant Lines. *Mathematics of Operations Research* 21:115–134

[16] Ford, L. R.,Jr., Fulkerson, D. R. (1962) *Flows in Networks*, Princeton University Press, Princeton.

[17] Foss, S., and Chernova, N. (1998) On the stability of a partially accessible multi-station queue with state-dependent routing. *Queueing Systems* 29(1):55–73.

[18] Garnett, O. and Mandelbaum, A. (2000) An Introduction to Skill-Based Routing and its Operational Complexities. http://iew3.technion.ac.il/serveng/Lectures/SBR.pdf

[19] Green, L. (1985) A queueing system with general-use and limited-use servers. *Operations Research* 33:168–182.

[20] Kaplan, E.H. (1988) A public housing queue with reneging and task-specific servers. *Decision Sciences* 19:383–391.

[21] McManus, M.L., Long, M.C., Copper, A., Litvak, E. (2004) Queueing theory accurately models the need for critical care resources. *Anesthesiology* 100: 1271–1276.

[22] Jennings O. B., Reed, J. E (2011) An Overloaded Multiclass FIFO Queue with Abandonments. *Operations Research* to appear.

[23] Talreja, R. and Whitt, W. (2007) Fluid Models for Overloaded Multi-class many-service queueing systems with FCFS routing. *Management Science* 54:1513–1527.

[24] Visschers, J., Adan, I.J.B.F., Weiss, G. (2012) A product form solution to a system with multi-type customers and multi-type servers. *Queueing Systems* 70:269–298.

[25] Zijm, W.H.M., Laarhoven, P.J.M. (1993) Production Preparation and Numerical Control in PCB assembly. *International Journal of Flexible Manufacturing Systems* 5(3):187–207.