Design heuristic for parallel many server systems under FCFS-ALIS

Please check the document version of this publication:

Design Heuristic for Parallel Many Server Systems under FCFS-ALIS

Ivo Adan[*]   Marko Boon[†]   Gideon Weiss[‡]

March 7, 2016

Abstract

We study a parallel service queueing system with servers of types $s_1, \ldots, s_J$, customers of types $c_1, \ldots, c_I$, bipartite compatibility graph $\mathcal{G}$, where arc (c_i, s_j) indicates that server type s_j can serve customer type c_i, and service policy of first come first served FCFS, assign longest idle server ALIS. For a general renewal stream of arriving customers and general service time distributions, the behavior of such systems is very complicated, in particular the calculation of matching rates r_{c_i, s_j}, the fraction of services of customers of type c_i by servers of type s_j, is intractable. We suggest through a heuristic argument that if the number of servers becomes large, the matching rates are well approximated by matching rates calculated from the tractable FCFS bipartite infinite matching model. We present simulation evidence to support this heuristic argument, and show how this can be used to design systems for given performance requirements.

1   Introduction

Parallel service systems have servers of types $\mathcal{S} = \{s_1, \ldots, s_J\}$, customers of types $\mathcal{C} = \{c_1, \ldots, c_I\}$, and bipartite compatibility graph $\mathcal{G} \subseteq \mathcal{C} \times \mathcal{S}$, where $(c_i, s_j) \in \mathcal{G}$ if servers of type s_j can serve customers of type c_i. They model situations in which a large volume of service requests of various types are channelled to a central facility, where they are attended by a large number of agents differentiated by skill. Such situations commonly occur in manufacturing, transportation, service contact centers, health systems, communications, internet data exchange, computing and various other areas of applications. The queueing model has a general renewal stream of arriving customers with rate λ, where successive arrivals are of i.i.d types, c_i with probability α_{c_i}, and there is a total of n servers, n_{s_j} of which are of type s_j. Service times are independent, distributed according to general distributions G_{c_i, s_j}, with mean m_{c_i, s_j} and service rate $\mu_{s_j, c_i} = 1/m_{c_i, s_j}$. Customers have finite patience, with independent patience time distributions F_{c_i}, and a customer abandons if he does not start service by the time his patience is exhausted.

Parallel server systems are widely discussed in the literature. An incomplete list would include an early study [13]; applications to manufacturing and supply chain management [26, 21], applications to call centers and internet service systems [11, 16, 22, 27], attempts to find optimal

[*]Department of Industrial Engineering, Eindhoven University of Technology, P.O. Box 513, 5600 MB Eindhoven, the Netherlands; email iadan@tue.nl

[†]Department of Mathematics and Computer Science, Eindhoven University of Technology, P.O. Box 513, 5600 MB Eindhoven, the Netherlands; email marko@win.tue.nl

[‡]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 Grants 711/09 and 286/13.

policies, mainly for small graph systems [29, 8, 6, 7, 12, 24], heavy traffic and fluid approximations [17, 18], and many server scaling [14, 15]. Most relevant to our current paper are [10, 5, 23, 20].

In assessing such systems there are various objectives that may be of importance, on the customer side they include waiting times and abandonment rates as well as consideration of fairness to customers of various types or priorities for some types. In conflict with those, on the server side there is the objective of maximum utilization of the servers, minimizing their number, and reaching a balanced work division between the various types. Each of these may carry a different weight in different application contexts. Often λ, α_{c_i} and F_{c_i} are given, together with some form of quality of service requirements. All the other parameters of the system can be adjusted to achieve the requirements in an optimal way: One can redesign the bipartite compatibility graph, change the service rates, change the workforce mix, decide on n, and decide on the service policy. It should perhaps be pointed out that changing the service policy may be as hard and costly as adjusting any of the other service parameters. At this level of generality such systems do not allow a complete analytic analysis, and performance is often evaluated in practice by simulation. However, any methods for calculating approximate performance measures or supporting design without the need to use simulation should be quite valuable. It is the aim of this paper to deliver such methods.

In the current paper we focus on the policy of first come first served (FCFS), where whenever a server is available he will take the longest waiting compatible customer, and assign longest idle server (ALIS), where whenever a customer arrives he will be assigned to the longest idling compatible server. We provide a heuristic to calculate performance measures under this policy, when λ and n are large.

FCFS-ALIS in a parallel service system has several advantages: It attempts to achieve resource pooling [25], i.e. all the servers are busy for about the same fraction of time, and it attempts to give all customers the same service level, i.e. global FCFS, equally for all types of customers [23]. It is also fair to the servers. One notable property of FCFS is the following: Assume that arriving customers can choose the server they wish to go to, and servers then serve the queueing customers FCFS. If each arrival has complete information on the schedule of all the servers at his moment of arrival, then to minimize his waiting time he will join the compatible server that has the shortest workload (JSW). Thus JSW is the Nash equilibrium of fully informed customers minimizing waiting times. But this policy of JSW is automatically achieved when customers queue up in a single queue and the servers are using FCFS. FCFS can then serve as a benchmark, and comparison of the costs under FCFS with other policies will provide an estimate of the price of anarchy. Apart form that, FCFS is easy to implement, as it does not require any calculations or knowledge of system parameters. It is also sometimes required by law. Finally it is indeed a policy very commonly used in practice. On the minus side, FCFS may waste resources by letting servers serve customers for which they are not efficient, and it may cause long delays to customer types that have a limited number of compatible servers. However, some of these shortcomings can be avoided by redesigning the compatibility graph.

Unfortunately, analysis of parallel service systems under FCFS is very hard. Foss and Chernova [10] provide an example of a symmetric system with 3 types of customer and 3 servers, and just 2 service distributions with fixed fast and slow service rates, where stability of the system depends on the entire shape of the service time distributions. The difficulty is in calculating the matching rates r_{c_i,s_j}, defined as the long term average fraction of customers of type c_i which are processed by servers of types s_j. Given the matching rates, one can calculate the total service capacity of the system, as

$$\mu = \sum_{(c_i,s_j)\in\mathcal{G}} r_{c_i,s_j}\mu_{c_i,s_j}.$$

It is the calculation of the matching rates, how many customers of type c_i are served by servers

of type s_j under FCFS-ALIS, which is intractable, and may depend on the entire shape of the service time distributions. Matching rates can be calculated for some types of graphs [23, 20], and they can also be calculated for general bipartite graphs when arrivals are Poisson, service rates depend only on the servers, and services are exponential [5], but not otherwise. However, matching rates can be calculated for the much simpler and very tractable FCFS infinite bipartite matching model [9, 4, 3].

It is our observation that when λ and n are large, the matching rates of the general parallel service system under FCFS-ALIS are approximately those of the FCFS infinite bipartite matching model, which is the basis for our current paper. We use our ability to calculate matching rates in order to design parallel service systems operating under FCFS-ALIS. We consider parallel service systems operating in Efficiency Driven mode (ED), Quality Driven mode (QD), and Quality and Efficiency Driven mode (QED) [19]. Our objective in each of these is to design the workforce required to achieve certain service requirements, specifically:

- In ED mode, we specify average waiting times for customers, and resulting abandonment rates.
- In QD mode, we specify average idle time for the servers,
- In QED mode we specify almost full utilization, no wait or short wait, and no abandonments.

Under FCFS-ALIS we achieve these pre-specified requirements with complete resource pooling of servers, and balance service levels for all types of customers.

We also present designs where under FCFS-ALIS the parallel servers are not pooled, and use this to achieve differentiated service levels for the various types of customers, based on pre-specified priority levels.

The rest of the paper is structured as follows. In Section 2 we describe the FCFS infinite bipartite matching model, and the formula for the calculation of matching rates. In Section 3 we present our conjecture on the behavior of FCFS-ALIS parallel service systems under many server scaling, which ties them up with the infinite bipartite matching model. In Section 4 we present our design algorithms, based on the calculation of matching rates. Finally, in Section 5 we present examples in which we calculate designs, and examine the performance under our designs. We present extensive simulation results, that confirm the validity of our approach for a range of λ, n scales.

This paper is an expansion, generalization, completion and extension of preliminary studies in [2, 1]. A recent discussion of many server scaling for the "N" system appears in [30].

2 FCFS infinite bipartite matching

We now consider a system with customer types $\mathcal{C}$ and server types $\mathcal{S}$, with a bipartite compatibility graph $\mathcal{G}$, and a much simplified stochastic model: We have infinite sequences of customers $c^1, c^2, \ldots, c^m, \ldots$ where $c^m \in \mathcal{C}$ and of servers $s^1, s^2, \ldots, s^n, \ldots$ where $s^n \in \mathcal{S}$. We assume that c^m are drawn according to probabilities $\alpha = (\alpha_{c_1}, \ldots, \alpha_{c_I})$ and s^n are drawn according to probabilities $\beta = (\beta_{s_1}, \ldots, \beta_{s_J})$, and they are all independent. For each realization of the sequences we match customers and servers according to a FCFS policy: s^n is matched to the earliest compatible c^m in $c^1, c^2, \ldots$, which has not yet been matched to $s^1, \ldots, s^{n-1}$. The matching process, for given graph $\mathcal{G}$, is illustrated in the figure below.
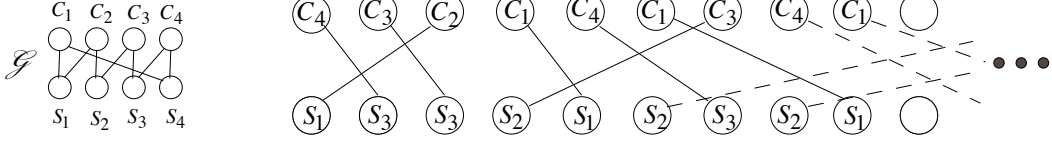
This model is much simpler than a queueing model, since it involves no arrival times, no service times, no busy or idle servers, and since it treats customers and servers in an entirely symmetric way. This system is studied in [9, 4, 3]. It is shown in [4] that the matching is uniquely determined for any two sequences and that all customers and servers are matched almost surely. Furthermore, the system demonstrates dynamic reversibility, and is associated with a Markov chain that has a product form stationary distribution. The stationary distribution is used to obtain explicit expressions for the matching rates. We describe the calculation of the matching rates now.

We use the following notations: we let $\mathcal{C}(s_j)$ be the set of customer types compatible with server type s_j, and $\mathcal{S}(c_i)$ be the set of server types compatible with customer type c_i. For a subset of customer types C we let $\mathcal{S}(C) = \bigcup_{c_i \in C} \mathcal{S}(c_i)$, and for a subset of server types S we let $\mathcal{C}(S) = \bigcup_{s_j \in S} \mathcal{C}(s_j)$. We also let $\mathcal{U}(S) = \overline{\mathcal{C}(\overline{S})}$ be the customer types that can only be served by servers of types in S. For subsets C, S we define $\alpha_C = \sum_{c_i \in C} \alpha_{c_i}$, and $\beta_S = \sum_{s_j \in S} \beta_{s_j}$.

Definition 2.1 *For given* $\alpha, \beta, \mathcal{G}$ *we say that there is complete resource pooling in the FCFS infinite bipartite matching system if the following three equivalent conditions hold:*

$$\alpha_C < \beta_{\mathcal{S}(C)}, \quad \beta_S < \alpha_{\mathcal{C}(S)}, \quad \beta_S > \alpha_{\mathcal{U}(S)}, \qquad S \subset \mathcal{S}, S \neq \emptyset, \mathcal{S}, \quad C \subset \mathcal{C}, C \neq \emptyset, \mathcal{C}. \qquad (1)$$

Theorem 2.2 (from [4]) *Let* $r_{c_i,s_j}(n)$ *be the (random) number of* c_i, s_j *matches between* $c^1, \ldots, c^n$ *and* $s^1, \ldots, s^n$, *in the FCFS infinite bipartite matching of the two sequences. If complete resource pooling holds, then almost surely* $\lim_{n \to \infty} r_{c_i,s_j}(n) = r_{c_i,s_j}$ *which is calculated by*

$$
\begin{aligned}
r_{c_i,s_j} &= \beta_{s_j} \sum_{\mathcal{P}_J} B \prod_{k=1}^{J-1} (\beta_{(k)} - \alpha_{(k)})^{-1} \\
&\quad \left(\sum_{k=1}^{J-1} \phi_k \frac{\alpha_{(k)}}{\beta_{(k)} - \alpha_{(k)}\chi_k} \prod_{l=1}^{k-1} \frac{\beta_{(l)} - \alpha_{(l)}}{\beta_{(l)} - \alpha_{(l)}\chi_l} + \frac{\phi_J}{\phi_J + \psi_J} \prod_{l=1}^{J-1} \frac{\beta_{(l)} - \alpha_{(l)}}{\beta_{(l)} - \alpha_{(l)}\chi_l} \right), \qquad (2)
\end{aligned}
$$

where the summation is over $\mathcal{P}_J$, *the set of all permutations of the server types* $\mathcal{S}$, *and for each permutation of the servers* $S_1, \ldots, S_J$, *the following notation is used:*

$$\alpha_{(k)} = \alpha_{U\{S_1,\ldots,S_k\}}, \qquad \beta_{(k)} = \beta_{\{S_1,\ldots,S_k\}}, \qquad k = 1, \ldots, J,$$

$$\phi_k = \frac{\alpha_{U\{S_1,\ldots,S_k\} \cap \{c_i\}}}{\alpha_{U\{S_1,\ldots,S_k\}}}, \qquad \psi_k = \frac{\alpha_{U\{S_1,\ldots,S_k\} \cap (C(s_j)\setminus\{c_i\})}}{\alpha_{U\{S_1,\ldots,S_k\}}}, \qquad \chi_k = 1 - \phi_k - \psi_k ,$$

and B *is the normalizing constant:*

$$B^{-1} = \sum_{\mathcal{P}_J} \left((\beta_{\{S_1\}} - \alpha_{\mathcal{U}\{S_1\}})(\beta_{\{S_1,S_2\}} - \alpha_{\mathcal{U}\{S_1,S_2\}}) \cdots (\beta_{\{S_1,\ldots,S_{J-1}\}} - \alpha_{\mathcal{U}\{S_1,\ldots,S_{J-1}\}}) \right)^{-1}.$$

An easy example of this formula is for the case that $I = J$ and $\mathcal{C}(s_j) = \mathcal{C}\setminus c_j$, i.e the bipartite compatibility graph is almost complete, each server can serve all but one of the customer types.

4

In that case, complete resource pooling holds if and only if $\alpha_{c_j} + \beta_{s_j} < 1$, and the matching rates are:

$$r_{c_i,s_j} \quad = \quad \alpha_{c_i}\beta_{s_j}\frac{(1-\alpha_{c_i})(1-\beta_{s_j}) - \alpha_{c_j}\beta_{s_i}}{(1-\alpha_{c_i}-\beta_{s_i})(1-\alpha_{c_j}-\beta_{s_j})} \bigg/ \left(1 + \sum_{i=1}^{I}\frac{\alpha_i\,\beta_i}{1-\alpha_i-\beta_i}\right). \qquad (3)$$

However, for any other bipartite compatibility graph, this formula does not seem to simplify, and we suspect that its calculation is $\sharp P$ hard. We have programmed it to be able to calculate it up to $I, J \leq 12$, but it will become hard to compute the matching rates for larger number of types. Further research to obtain perhaps easier approximations to r_{c_i,s_j} may be necessary.

When resource pooling does not hold, it is shown in [5] that there is a unique decomposition $(\mathcal{C}, \mathcal{S})$ into subsystems $(\mathcal{C}^{(1)}, \mathcal{S}^{(1)}), \ldots, (\mathcal{C}^{(L)}, \mathcal{S}^{(L)})$, such that

$$\frac{\beta_{\mathcal{S}^{(1)}}}{\alpha_{\mathcal{C}^{(1)}}} < \cdots < \frac{\beta_{\mathcal{S}^{(L)}}}{\alpha_{\mathcal{C}^{(L)}}}, \qquad (\mathcal{C}^{(l)}, \mathcal{S}^{(l)}) \text{ has complete resource pooling, } l = 1, \ldots, L. \qquad (4)$$

A Mathematica program to calculate the matching rates for given $\alpha, \beta, \mathcal{G}$ is available from the authors.

3   Matching under many server scaling

Consider a queueing system with a single customer type and a single server type, with arrival rate λ, and patience distribution F, and with n servers, each with service rate μ, so that the traffic intensity is $\rho = \lambda/n\mu$. Many server scaling occurs when we keep μ and ρ fixed and let both λ and n increase. Note that, to increase λ, we scale the inter-arrival time distribution, and thus we do not alter its shape. Because of abandonments the system will always be stable. There will be three behavior modes for this system: When $\rho < 1$ the system is in QD (*quality driven mode*). In QD mode, there is always a fraction $\approx (1-\rho)$ of idle servers and customers never wait and nobody abandons. When $\rho > 1$ the system is in ED (*efficiency driven mode*). In ED mode, servers are always busy, there is always a queue, and a fraction $\approx F(W) = (\rho-1)/\rho$ of customers abandons without service. Customers with patience $\leq W$ do not get served, and customers with patience $> W$ receive service after a wait of $\approx W$. When $\rho \approx 1$ the system is in QED (*quality and efficiency driven mode*). In QED mode, servers are busy most of the time and if they idle that is only for a short while, an appreciable fraction of customers do not need to wait, most customers wait a very short time, and very few customers abandon [28].

We now consider the system of parallel skill based servers of Section 1. We fix the fractions α_{c_i}, n_{s_j}/n, the service time distributions G_{c_i,s_j}, and the patience distributions F_{c_i}. We use FCFS-ALIS policy, and we let λ and n increase at the same rate, so that we get into many server scaling. We cannot directly calculate ρ for this system, as it depends on the service policy, and in particular we cannot calculate it directly under FCFS-ALIS, but we will try and approximate it. Under many server scaling we can expect that for favorable choices of parameters, the system will achieve resource pooling, so that customers of different types will have similar waiting times, and servers of different types will have similar workloads and similar idle times. Under such conditions, the system will again behave in one of the three modes, QD, ED or QED, according to the traffic intensity.

We now make the following conjectures regarding the behavior of the system under many server scaling, when resource pooling holds: We conjecture that the order in which customers will reach the head of the line (if they did not abandon previously) will be such that the types of customers will be i.i.d. with some probabilities α_{c_i}, approximately. Also we conjecture that the

order in which servers will become available at the head of line (if there is a queue of idle servers), will be such that the types of servers will be i.i.d with some probabilities β_{s_j}, approximately. Under this conjecture, the matching between customers and servers will be approximately the same as for the FCFS infinite bipartite matching model.

The following three figures illustrate the operation of our system under FCFS-ALIS in each of the above mentioned three modes:

In ED mode, all the servers are always busy, customers with enough patience wait a time W, and when they reach the head of the queue, they match with the next compatible server. Note that customers entering service are still of i.i.d. types, approximately, but with new probabilities α_{c_i}, since they are thinned independently by impatience.
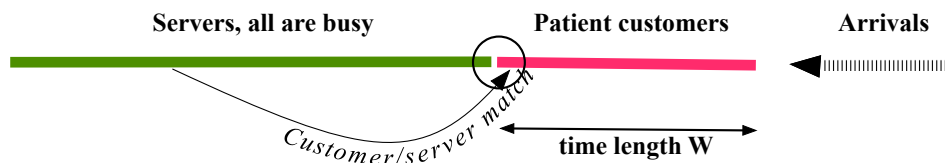
In QD mode, there is a queue of idle servers, each server, on completing a service, joins the end of this queue. A server reaches the head of the queue after an idle time T, and matches with the first compatible customer. Customers never wait and are of i.i.d. types with probabilities α_{c_i}.
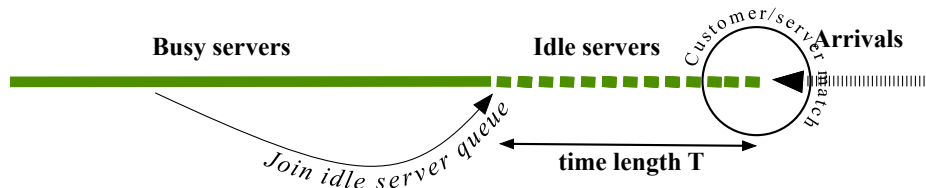
In QED mode, the system alternates infrequently between periods with a queue of customers and periods with a queue of servers. All servers are almost always busy, customers immediately enter service or wait a short time, and there are only a few abandonments.

The key assumption necessary for the matching rates to be according to the FCFS infinite bipartite matching model is that approximately the sequence of customers entering service has i.i.d. types and the sequence of servers that become available and that start service has i.i.d. types.

When there is no resource pooling, the system decomposes into subsystems as in (4). The following figure shows how such a system will behave under our conjecture. Here the system is decomposed into three sub systems, where $(\mathcal{C}^{(3)}, \mathcal{S}^{(3)})$ receives better service than $(\mathcal{C}^{(2)}, \mathcal{S}^{(2)})$, which receives better service than $(\mathcal{C}^{(1)}, \mathcal{S}^{(1)})$.
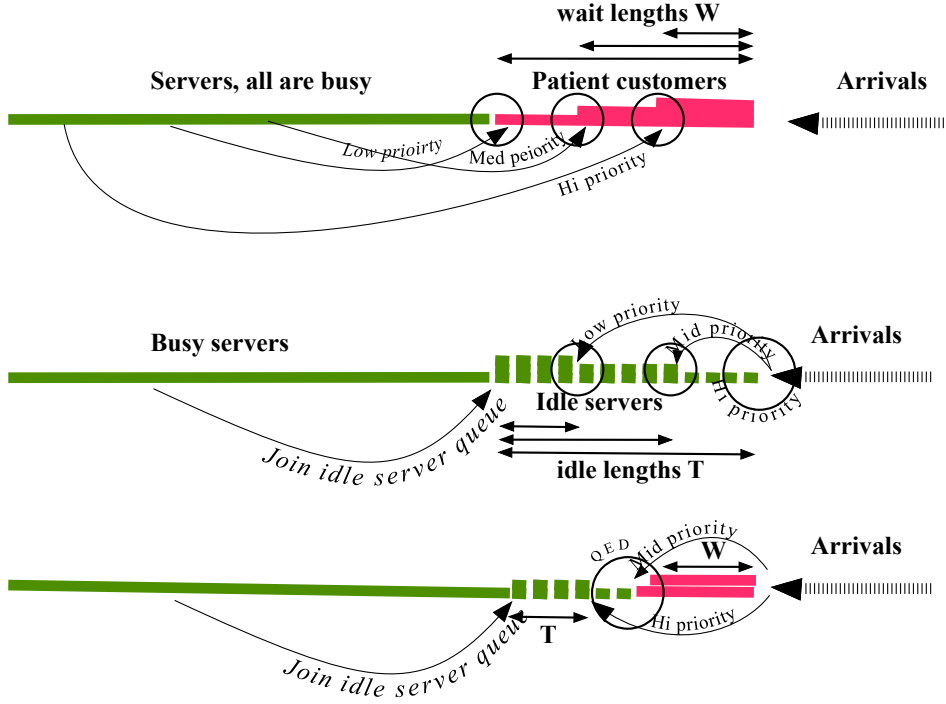
The three illustrations depict behavior under ED (top figure), QD (middle figure), and in the bottom one, $(\mathcal{C}^{(1)}, \mathcal{S}^{(1)})$ is in ED, $(\mathcal{C}^{(2)}, \mathcal{S}^{(2)})$ is in QED and $(\mathcal{C}^{(3)}, \mathcal{S}^{(3)})$ is in QD. In ED mode, customers of types $\mathcal{C}^{(1)}$ wait the longest time, before being served by servers of types $\mathcal{S}^{(1)}$. Servers of types $\mathcal{S}^{(2)}$ skip customers of type $\mathcal{C}^{(1)}$ and serve customers of type $\mathcal{C}^{(2)}$ that have a shorter wait, and servers of types $\mathcal{S}^{(3)}$ skip customers of type $\mathcal{C}^{(1)}, \mathcal{C}^{(2)}$ and serve customers of type $\mathcal{C}^{(3)}$ that have the shortest wait. In QD mode, servers of the high priority customers have longer idle periods, and are available for the high priority customers immediately, while servers of lower priority customers have shorter idle times, and are further in the queue, and their customers will skip the high priority servers in the queue, which are incompatible with them. In the mixed mode, low priority customers will wait, while high priority customers will have a queue of idle servers ready to serve them.

4 Design Algorithms

4.1 General Strategy

We specify the quality of service requirements: For QD this means specifying the utilization of the servers, for ED this means specifying the amount of waiting of patient customers, and for QED this means no abandonment and no idling. We next specify which fraction of the service should be done by each type of server. This is a design decision, which will not affect quality of service, but it is relevant for the operational costs. These determine the α and β for the matching of customers and servers. We then use the bipartite infinite matching model, formula (2), to obtain the matching rates r_{c_i, s_j}. Once we have the matching rates, we can calculate the amount of work required from each type of server, and this determines, by Little's law, the number of servers that are needed of each type in order to meet the requested service of quality. In the

7

case of differentiated service, we decompose the server and customer types into ED, QED and QD subsets.

In the following sections we show how to perform these steps for each of the three regimes, and for the decomposition. We illustrate the calculations for the examples of Section 5, and demonstrate the effectiveness of the heuristics through simulation.

Note that the inter-arrival time distribution and the service time distributions are not required as input for the algorithm, only the arrival rate and the average service times are required. The distribution of the abandonment time distribution is needed for design in ED mode.

4.2 Design for Quality Driven Service

Here the traffic intensity is < 1, and customers almost never wait, and therefore even more rarely abandon. There are almost always some idle servers waiting for customers, and because of ALIS, servers of different types all have the same idle time distribution. The quality parameter in this case is the value T of the average idle time. It is a measure of the utilization of the servers. Because there are virtually no abandonments, the patience time distribution is not required as input.

Algorithm for QD
Input:
• Compatibility graph $\mathcal{G}$
• Arrival rate λ
• Fractions of customer types α_{c_i}
• Mean service times m_{c_i,s_j}
Requested quality of service parameter:
• Mean server idle time after each service T
Design parameters:
• Fraction of services performed by each server type β_{s_j}
Algorithm:
Check α, β for complete resource pooling
Compute matching rates $\qquad\qquad\qquad\qquad r_{c_i,s_j} :=$ use Equation (2)
Compute staffing levels $\qquad\qquad\qquad\qquad n_{s_j} := \displaystyle\sum_{c_i \in \mathcal{C}(s_j)} \lambda r_{c_i,s_j}\big(m_{c_i,s_j} + T\big)$
Output:
• Required workforce n_{s_j}

4.3 Design for Efficiency Driven Service

Here the traffic intensity is > 1, servers are always busy and customers always need to wait, and a certain fraction will abandon. By FCFS, customers of different types all have the same waiting time distribution, and the system demonstrates global FCFS (this term was coined by Talreja and Whitt [23]). The system is stabilized by abandonments, with average waiting time

8

W. This means that approximately a fraction $1 - F_{c_i}(W)$ of customers of type c_i will abandon. The value of W is the quality of service parameter here, so that customers with patience less than W do not get served, while customers with patience that exceeds W get served after a wait of W. Since customers are thinned independently by impatience, we need to calculate the total effective arrival rate (of patient customers) and we need to adjust the fractions α_{c_i} of customer types entering service.

<table>
<tr><td colspan="2" align="center">Algorithm for ED</td></tr>
<tr><td colspan="2">Input:
    • Compatibility graph $\mathcal{G}$
    • Arrival rate λ
    • Fractions of customer types α_{c_i}
    • Patience distributions $F_{c_i}(\cdot)$
    • Mean service times m_{c_i,s_j}</td></tr>
<tr><td colspan="2">Requested quality of service parameter:
    • Mean waiting time W</td></tr>
<tr><td colspan="2">Design parameters:
    • Fraction of services performed by each server type β_{s_j}</td></tr>
<tr><td colspan="2">Algorithm:</td></tr>
<tr><td>Compute expected fraction of abandonments</td><td>$p_{c_i} := F_{c_i}(W)$</td></tr>
<tr><td>Compute the total effective arrival rate</td><td>$\tilde{\lambda} := \sum_{i=1}^{I} \alpha_{c_i}\lambda(1-p_{c_i})$</td></tr>
<tr><td>Adjust fraction of each customer type</td><td>$\alpha_{c_i} := \dfrac{\alpha_{c_i}\lambda(1-p_{c_i})}{\tilde{\lambda}}$</td></tr>
<tr><td>Check α,β for complete resource pooling</td><td></td></tr>
<tr><td>Compute matching rates</td><td>$r_{c_i,s_j} :=$ use Equation (2)</td></tr>
<tr><td>Compute staffing levels</td><td>$n_{s_j} := \sum_{c_i \in \mathcal{C}(s_j)} \tilde{\lambda} r_{c_i,s_j} m_{c_i,s_j}$</td></tr>
<tr><td colspan="2">Output:

    • Required workforce n_{s_j}</td></tr>
</table>

4.4 Design for Quality and Efficiency Driven Service

Given the arrival rates, there is a unique FCFS system that will supply QED service, with servers almost always busy, most customers either don't wait or wait a short time, and few abandonments. The calculation of QED design follows the same steps as for QD with $T = 0$ and for ED with $W = 0$.

4.5 Design for Differentiated Service

We now consider the case where we would like to give customers graded service levels, from high priority to standard priority to low priority customer types. We have a partition of customer types $\mathcal{C}$ into $\mathcal{C}^{(1)}, \ldots, \mathcal{C}^{(L)}$, where customers of types $c_i \in \mathcal{C}^{(l)}$ have higher priority than customer of type in $\mathcal{C}^{(l-1)}$, $l = 2, \ldots, L$. Customers in class $\mathcal{C}^{(l)}$ will then have a set of servers $\mathcal{S}^{(l)}$, so that each customer type $c_i \in \mathcal{C}^{(l)}$ will have at least one compatible server type $s_j \in \mathcal{S}^{(l)}$. In this decomposition to subsystems $(\mathcal{C}^{(l)}, \mathcal{S}^{(l)})$, we will allow $s_j \in \mathcal{S}^{(l)}$ to serve $c_i \in \mathcal{C}^{(k)}$, $k \geq l$, but will not allow $s_j \in \mathcal{S}^{(l)}$ to serve $c_i \in \mathcal{C}^{(k)}$, $k < l$. In other words, we redesign the compatibility graph $\mathcal{G}$ by eliminating all links from $\mathcal{S}^{(l)}$ to $\mathcal{C}^{(k)}$ for $k < l$, but we preserve the links to higher priority customers in $\mathcal{C}^{(k)}$ for $k > l$, since when we use FCFS, these links will hardly ever be used, because servers in $\mathcal{S}^{(l)}$ will be behind all servers in $\mathcal{S}^{(k)}$ for $k > l$ almost all the time.

The priorities will be translated into quality of service parameters: classes $\mathcal{C}^{(1)}, \ldots, \mathcal{C}^{(l)}$ will be served in ED mode with $W_1 > \cdots > W_l \geq 0$, classes $l+1, \ldots, L$ will be served in QD mode, with $0 < T_{l+1} < \cdots < T_L$. Class l may be in QED mode.

Algorithm for Differentiated Service

Input:
- Compatibility graph $\mathcal{G}$
- Arrival rate λ
- Fractions of customer types α_{c_i}
- Patience distributions $F_{c_i}(\cdot)$
- Mean service times m_{c_i, s_j}

Requested quality of service parameters:
- Partition of customer types by priority into $\mathcal{C}^{(1)}, \ldots, \mathcal{C}^{(L)}$
- Quality of service parameters: $W_1 > \cdots > W_l = 0 = T_l < T_{l+1} < \cdots < T_L$

Design parameters:
- Choose partition of server types $\mathcal{S}^{(1)}, \ldots, \mathcal{S}^{(L)}$
- Eliminate links from $\mathcal{S}^{(l)}$ to $\mathcal{C}^{(k)}$, for $k < l$
- Assign fraction of services performed by each server type β_{s_j}, within $\mathcal{S}^{(l)}$

Algorithm:

- For subsystem $(\mathcal{C}^{(l)}, \mathcal{S}^{(l)})$, $l = 1, \ldots, L$:
- Apply appropriate design algorithm for QD or ED to the subsystem

Output:

- Redesigned compatibility graph $\mathcal{G}$
- Required workforce n_{s_j}

---

5 Examples of Designs and Simulation Results

In this section we describe three examples, for each of which we have prepared several designs, under several modes of operation, and assuming Poisson arrivals using a range of values for λ. We have then performed extensive simulation runs on each of these designs. Our purpose in this section is threefold:

(i) Illustrate the implementation of the algorithms;

(ii) Examine the validity of the matching rates conjecture;

(iii) Evaluate the efficacy of our designs.

The first example system has 3 customer types and 3 server types with an almost complete bipartite compatibility graph. We have designed operation of this system with complete resource pooling, in ED, QD and QED mode. The purpose of this example is to assess pooled service designs. The second example has 5 customer types and 5 server types with a Hamiltonian bipartite compatibility graph. This example is used to assess differentiated service, with the customer types divided into high, standard and low priority customers. The third example has 6 customer types and 6 server types with a symmetric compatibility graph that has degree 3 for all nodes. The purpose of considering this example is to examine validity of the matching conjecture in a complex graph.

The designs depend on the service rates for each link, but not on the actual distributions of service times. To examine the validity of the matching conjecture and to assess the efficacy of the designs, we have chosen to simulate service time distributions which are very different, including uniform in a finite range, exponential and Pareto.

Our main conclusions from the simulations of these examples are:

(i) The matching rates conjecture seems to be valid under ED, QD and QED mode, for the whole range of λ values, and under all the different distributions of service times. For small values of λ the deviations are slightly larger, but this can be partly explained by the fact that the algorithm yields real numbers for n_{s_j}, but in the final design rounded integer values are used.

(ii) In ED mode, for large values of λ we get convergence to the exact values of W with very small variability in waiting times, and exact abandonment rates. Similarly under QD, for large values of λ we get convergence to the exact values of T with very small variability in idle times, and almost all customers are not waiting for service.

(iii) Most important, it seems that for small values of λ, while waiting times in ED mode and idle times in QD mode are quite variable, the average waiting time in ED and the average idle time in QD are almost exactly as designed. This indicates that our design heuristic may be effective already for a moderate number of servers.

(iv) Convergence in the QED mode is not appreciably worse than in the ED or QD modes.

(v) The system with differentiated service performs as designed.

(vi) The results do not seem to depend on the service time distributions.

We now present the three examples with detailed simulation results. The reported simulation results for each design have been obtained as the average of 1,000 runs, where each run consists of 1,250,000 customers. However, the first 250,000 customers have been removed from the results to account for a possible startup effect.

5.1 Example 1 − 3 × 3 Almost Complete Graph with Pooled Service

In this example we investigate pooled service designs. The system is specified below, where $\mathrm{Exp}(a)$ denotes the exponential distribution with rate a, $\mathrm{U}(a, b)$ is the uniform distribution on the interval (a, b) and $\mathrm{Pareto}(k, a)$ is the Pareto distribution $F(t) = 1 - (k/t)^a$ for $t > k$.

<div style="border:1px solid black; padding:10px;">

Example 1 – System and Data

There are 3 types of customers and 3 types of servers. The total arrival rate is parameterized by λ, The graph and the values of $\alpha_{c_i}\lambda$ are described in the following figure:
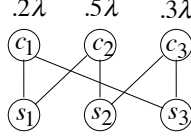
The patience times and service time distribution are given in the tables below.

Patience time distributions	
	F_{c_i}
c_1	Exp(0.1)
c_2	U(0,10)
c_3	Exp(0.2)

Service time distributions			
G_{c_i,s_j}	c_1	c_2	c_3
s_1	Pareto(2, 3)	Exp(0.125)	
s_2		Exp(0.2)	U(2, 6)
s_3	Pareto(3, 3)		U(1, 5)

Only the *mean* service times are used by the design algorithms. The full distributions are used in the simulations.

</div>

In the designs for Example 1 we take as service fractions: $\beta_{s_1} = 0.3$, $\beta_{s_2} = 0.3$, $\beta_{s_3} = 0.4$.

ED design: We specify the average waiting time $W = 1$, corresponding to approximately 25% of the average service times. For the given patience distributions this entails abandonment rates of approximately 10% for customers of types 1 and 2, and of 18% for customers of type 3. We calculate the effective arrival rates of customers that do get served after a wait of $W = 1$:

$$1 - F_{c_1}(W) = e^{-0.1W} = 0.905, \quad 1 - F_{c_2}(W) = (10 - W)/10 = 0.9, \quad 1 - F_{c_3}(W) = e^{-0.2W} = 0.819,$$

so

$$\alpha_{c_1}\lambda(1 - F_{c_1}(W)) = 0.2\lambda \times 0.905 = 0.181\lambda, \quad \lambda_{c_2}(1 - F_{c_2}(W)) = 0.450\lambda, \quad \lambda_{c_2}(1 - F_{c_2}(W)) = 0.246\lambda,$$

and thus the effective arrival rate equals

$$\tilde{\lambda} = (0.181 + 0.450 + 0.246)\lambda = 0.877\lambda.$$

Hence, the adjusted values of α_{c_j} are:

$$\alpha_{c_1} = \frac{0.18}{0.88} = 0.206, \quad \alpha_{c_2} = \frac{0.45}{0.88} = 0.513, \quad \alpha_{c_3} = \frac{0.25}{0.88} = 0.281.$$

QD design: We take an average idle time of $T = 0.5$. This corresponds to an utilization of approximately 0.9.
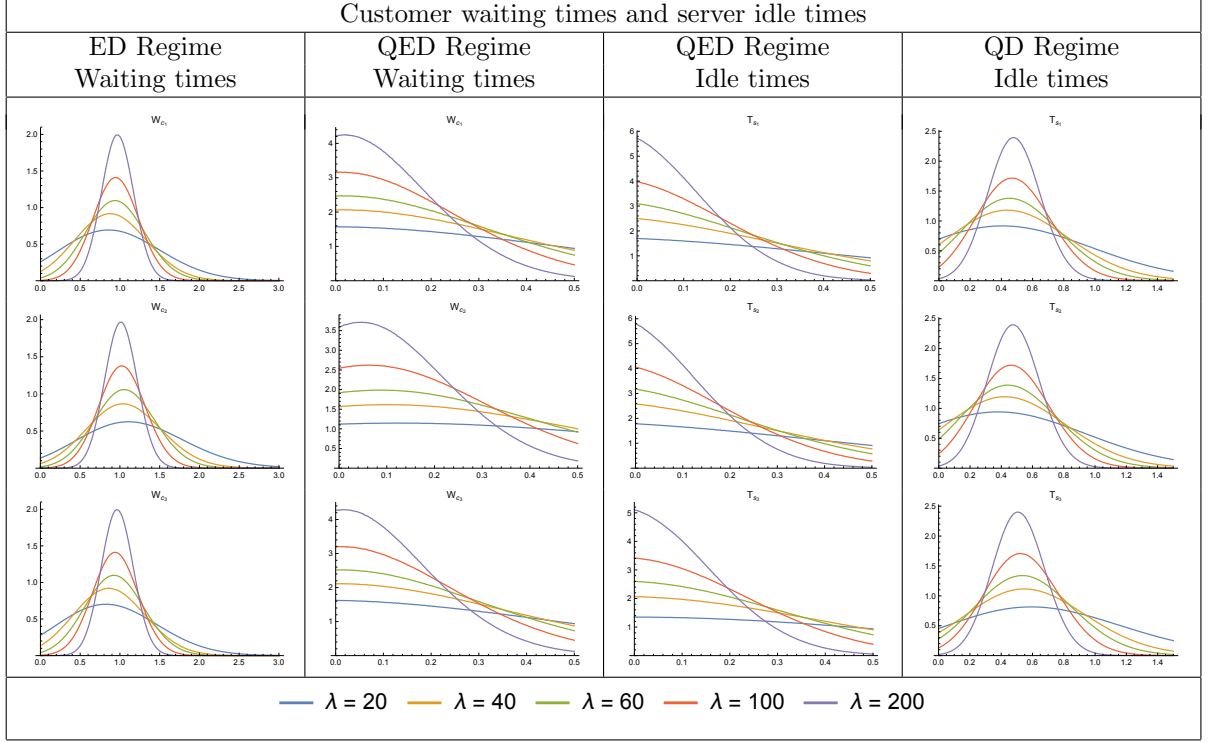
QED design: The unadjusted values of $\lambda, \alpha_{c_i}, m_{c_i,s_j}$ are used.

It is readily verified that in all three regimes (ED, QD and QED), Conditions (1) are satisfied, so complete resource polling holds. From the algorithms we obtain the *calculated required workforce* for the three designs:

	Required workforce								
	ED regime			QED regime			QD regime		
λ	n_{s_1}	n_{s_2}	n_{s_3}	n_{s_1}	n_{s_2}	n_{s_3}	n_{s_1}	n_{s_2}	n_{s_3}
20	39	25	25	44	29	29	47	32	33
40	77	51	51	88	58	57	94	64	65
60	116	76	76	131	87	86	140	96	98
100	194	127	127	219	144	144	234	159	164
200	387	254	255	438	288	287	468	318	327

The simulation results for Example 1 are listed in the tables below. We note that the histograms below only depict the waiting times and idle times greater than zero (so probability mass at zero is not shown).

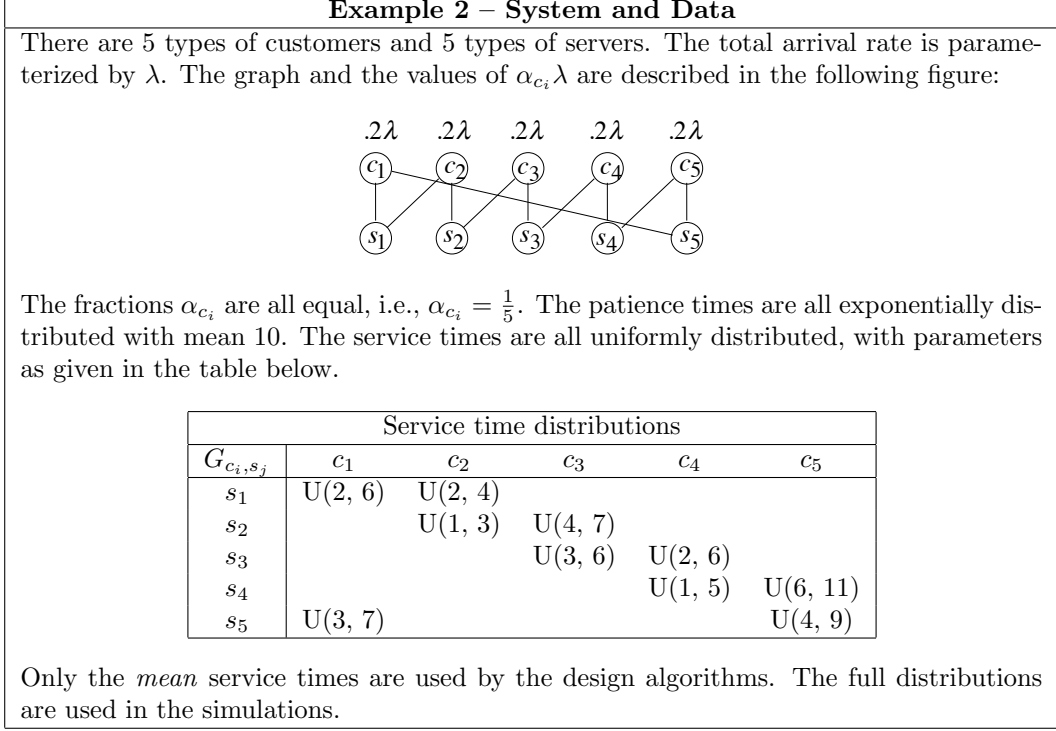|	Matching rates								
	ED regime			QED regime			QD regime		
	Theoretical								
r_{c_i,s_j}	c_1	c_2	c_3	c_1	c_2	c_3	c_1	c_2	c_3
s_1	0.038	0.262		0.042	0.258		0.042	0.258	
s_2		0.251	0.049		0.242	0.058		0.242	0.058
s_3	0.168		0.232	0.158		0.242	0.158		0.242
	$\lambda = 20$								
r_{c_i,s_j}	c_1	c_2	c_3	c_1	c_2	c_3	c_1	c_2	c_3
s_1	0.046	0.262		0.048	0.260		0.047	0.258	
s_2		0.241	0.056		0.239	0.064		0.241	0.065
s_3	0.164		0.230	0.155		0.234	0.153		0.236
	$\lambda = 60$								
r_{c_i,s_j}	c_1	c_2	c_3	c_1	c_2	c_3	c_1	c_2	c_3
s_1	0.041	0.261		0.045	0.258		0.045	0.257	
s_2		0.248	0.051		0.242	0.061		0.243	0.062
s_3	0.167		0.232	0.156		0.237	0.155		0.238
	$\lambda = 200$								
r_{c_i,s_j}	c_1	c_2	c_3	c_1	c_2	c_3	c_1	c_2	c_3
s_1	0.039	0.261		0.043	0.259		0.043	0.258	
s_2		0.250	0.049		0.242	0.059		0.242	0.059
s_3	0.168		0.232	0.157		0.240	0.157		0.241

Customer waiting times and server idle times

ED Regime Waiting times	QED Regime Waiting times	QED Regime Idle times	QD Regime Idle times

W_{c_1} · W_{c_2} · W_{c_3} W_{c_1} · W_{c_2} · W_{c_3} T_{s_1} · T_{s_2} · T_{s_3} T_{s_1} · T_{s_2} · T_{s_3}

— $\lambda = 20$ — $\lambda = 40$ — $\lambda = 60$ — $\lambda = 100$ — $\lambda = 200$

	Fraction of no wait and of no idling					
	ED regime		QED regime		QD regime	
λ	No waiting	No idling	No waiting	No idling	No waiting	No idling
20	0.047	0.946	0.444	0.540	0.814	0.181
60	0.003	0.997	0.420	0.571	0.947	0.053
200	0.000	1.000	0.410	0.585	0.999	0.001

	Abandonment rates								
	ED regime			QED regime			QD regime		
λ	c_1	c_2	c_3	c_1	c_2	c_3	c_1	c_2	c_3
20	0.089	0.123	0.168	0.020	0.035	0.039	0.003	0.008	0.006
60	0.091	0.109	0.173	0.014	0.020	0.027	0.000	0.001	0.001
200	0.093	0.102	0.177	0.008	0.010	0.016	0.000	0.000	0.000
Design	0.095	0.100	0.181	0.000	0.000	0.000	0.000	0.000	0.000

The table for the matching rates shows that the theoretical matching rates calculated by the algorithm are quite close to the simulated (actual) matching rates, already for moderate values of λ. The results for the waiting times and idle times confirm our intuition that they should converge to the targeted quality of service requirements: for large values of λ, the probability mass of the waiting times in ED concentrates near W and the probability mass of the idle times in QD concentrates near T. In the QED regime, waiting times, idle times and abandonment rates are small.

5.2 Example 2 – 5×5 Hamiltonian Graph with Differentiated Service

This example illustrates differentiated service, with the customer types divided into three classes: high, standard and low priority customers.

Example 2 – System and Data

There are 5 types of customers and 5 types of servers. The total arrival rate is parameterized by λ. The graph and the values of $\alpha_{c_i} \lambda$ are described in the following figure:

The fractions α_{c_i} are all equal, i.e., $\alpha_{c_i} = \frac{1}{5}$. The patience times are all exponentially distributed with mean 10. The service times are all uniformly distributed, with parameters as given in the table below.

Service time distributions					
G_{c_i, s_j}	c_1	c_2	c_3	c_4	c_5
s_1	U(2, 6)	U(2, 4)			
s_2		U(1, 3)	U(4, 7)		
s_3			U(3, 6)	U(2, 6)	
s_4				U(1, 5)	U(6, 11)
s_5	U(3, 7)				U(4, 9)

Only the *mean* service times are used by the design algorithms. The full distributions are used in the simulations.

We consider the following decomposition of the system of Example 2:

$$\mathcal{C}^{(1)} = \{c_1, c_2\}, \ \ \mathcal{S}^{(1)} = \{s_1\}, \ \ \ \mathcal{C}^{(2)} = \{c_3, c_4\}, \ \ \mathcal{S}^{(2)} = \{s_2, s_3\}, \ \ \ \mathcal{C}^{(3)} = \{c_5\}, \ \ \mathcal{S}^{(3)} = \{s_4, s_5\}.$$

The decomposed system is described in the following figure:
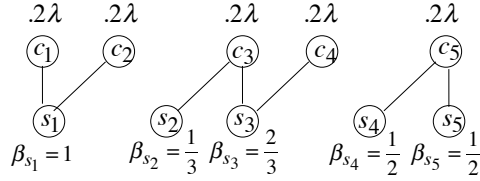
Note that this decomposition results from eliminating three links in the compatibity graph: the link from s_2 to c_2, s_4 to c_4, and from s_5 to c_1. We then have that:
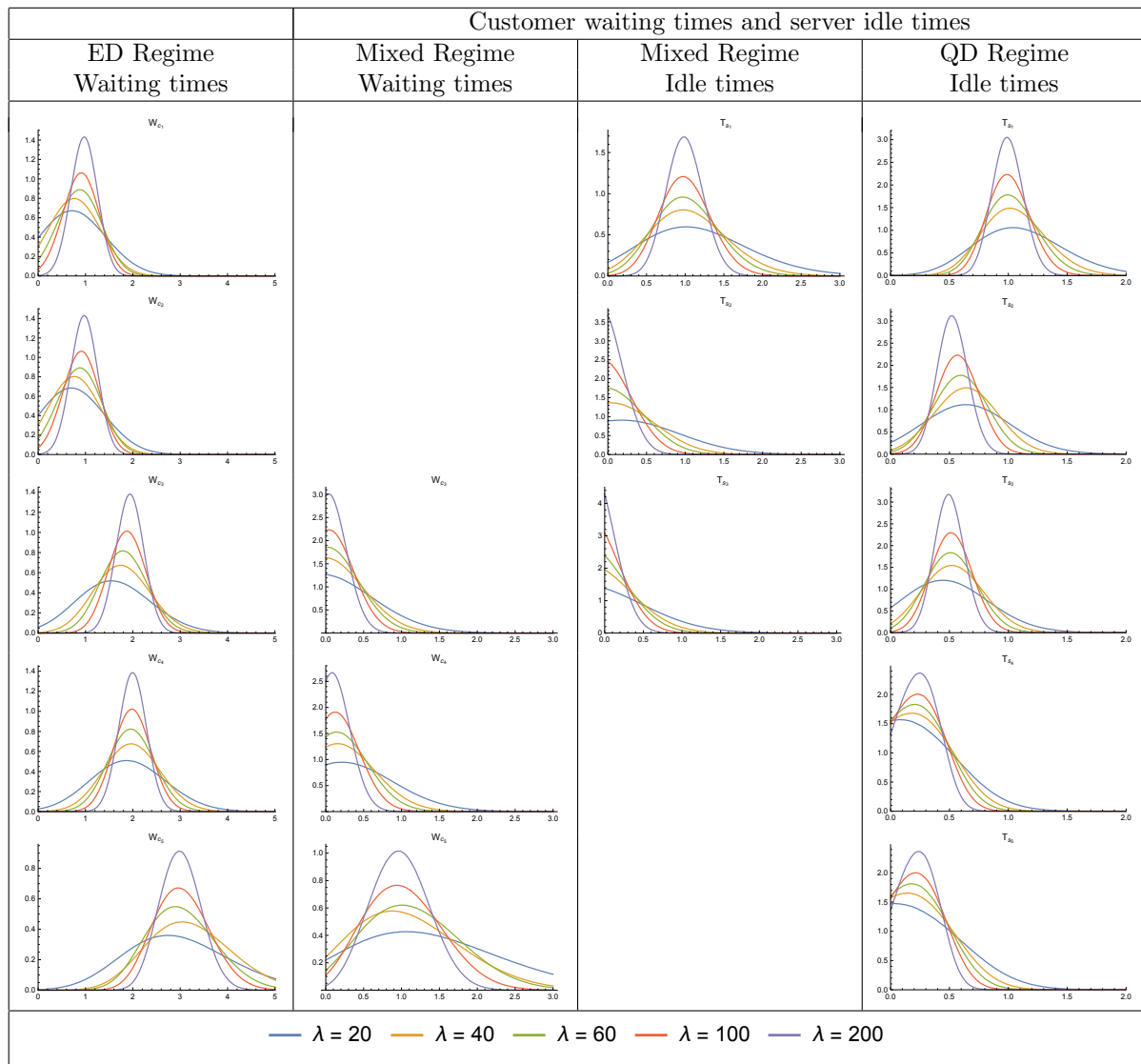- $\mathcal{C}^{(1)}, \mathcal{S}^{(1)}$ in isolation is a "V" system, with arrival rate 0.4λ, adjusted fractions $\alpha_{c_1} = \alpha_{c_2} = \frac{1}{2}$ and $\beta_{s_1} = 1$.
- $\mathcal{C}^{(2)}, \mathcal{S}^{(2)}$ in isolation is an "N" system, with arrival rates 0.4λ, adjusted fractions $\alpha_{c_3} = \alpha_{c_4} = \frac{1}{2}$ and we take $\beta_{s_2} = \frac{1}{3}, \beta_{s_3} = \frac{2}{3}$.
- $\mathcal{C}^{(3)}, \mathcal{S}^{(3)}$ in isolation is a "Λ" system, with arrival rate $\lambda = 0.2\lambda$, $\alpha_{c_5} = 1$ and we take $\beta_{s_4} = \beta_{s_5} = \frac{1}{2}$.

We make three designs for this network, in which customers c_1, c_2 have high priority, c_3, c_4 have standard priority, and c_5 have low priority. The first design is for a system in ED regime, the second is for a mixed design with the top priority sub-system in QD, the middle priority sub-system in QED and low priority sub-system in ED, and the third design is for the three systems in QD regime. It is readily verified, by checking Conditions (1), that in each design complete resource pooling holds for each subsystem. The following table shows the calculated workforce required for each type of server for the three designs, as a function of λ.

	Required workforce														
	ED regime					Mixed regime					QD regime				
	$W=1$	$W=2$		$W=3$		$T=1$	QED		$W=1$		$T=2$	$T=1$		$T=0.5$	
λ	n_{s_1}	n_{s_2}	n_{s_3}	n_{s_4}	n_{s_5}	n_{s_1}	n_{s_2}	n_{s_3}	n_{s_4}	n_{s_5}	n_{s_1}	n_{s_2}	n_{s_3}	n_{s_4}	n_{s_5}
20	25	12	18	13	10	36	15	22	15	12	44	17	27	18	14
40	51	24	36	25	19	72	29	44	31	24	88	35	55	36	28
60	76	36	54	38	29	108	44	66	46	35	132	52	82	54	42
100	127	60	90	63	48	180	73	110	77	59	220	87	137	90	70
200	253	120	180	126	96	360	147	220	154	118	440	173	273	180	140

The simulation results for Example 2 are listed in the tables below, and illustrate that the system with differentiated service performs as designed.

|	Matching rates														
	ED regime					Mixed regime					QD regime				
	Theoretical														
r_{c_i,s_j}	c_1	c_2	c_3	c_4	c_5	c_1	c_2	c_3	c_4	c_5	c_1	c_2	c_3	c_4	c_5
s_1	0.216	0.216				0.204	0.204				0.200	0.200			
s_2			0.130					0.136					0.133		
s_3			0.065	0.195				0.068	0.204				0.067	0.200	
s_4					0.088					0.092					0.100
s_5					0.088					0.092					0.100
	$\lambda = 20$														
r_{c_i,s_j}	c_1	c_2	c_3	c_4	c_5	c_1	c_2	c_3	c_4	c_5	c_1	c_2	c_3	c_4	c_5
s_1	0.215	0.196				0.207	0.194				0.200	0.186			
s_2		0.021	0.121				0.014	0.126				0.015	0.118		
s_3			0.079	0.176				0.077	0.186				0.083	0.181	
s_4				0.017	0.084				0.014	0.086				0.020	0.094
s_5	0.002				0.089	0.001				0.095	0.002				0.100
	$\lambda = 200$														
r_{c_i,s_j}	c_1	c_2	c_3	c_4	c_5	c_1	c_2	c_3	c_4	c_5	c_1	c_2	c_3	c_4	c_5
s_1	0.216	0.216				0.205	0.205				0.200	0.200			
s_2			0.130					0.135					0.132		
s_3			0.066	0.194				0.068	0.201				0.069	0.197	
s_4				0.001	0.088				0.001	0.093				0.003	0.099
s_5					0.088					0.093					0.101

	Customer waiting times and server idle times		
ED Regime Waiting times	Mixed Regime Waiting times	Mixed Regime Idle times	QD Regime Idle times

$\lambda = 20$ $\lambda = 40$ $\lambda = 60$ $\lambda = 100$ $\lambda = 200$

	Fraction of no wait and of no idling					
	ED regime		Mixed regime		QD regime	
λ	No waiting	No idling	No waiting	No idling	No waiting	No idling
20	0.056	0.934	0.578	0.398	0.869	0.124
60	0.010	0.988	0.570	0.412	0.935	0.063
200	0.000	1.000	0.556	0.430	0.974	0.025

5.3   Example 3 − 6 × 6 Symmetric Degree 3 Graph with Pooled Service

We now consider a more complex graph to examine the validity of the matching rates conjecture.

<div style="border:1px solid">

Example 3 – System and Data

There are 6 types of customers and 6 types of servers. The total arrival rate is parameterized by λ, The graph and the values of λ_{c_i}/λ are described in the following figure:
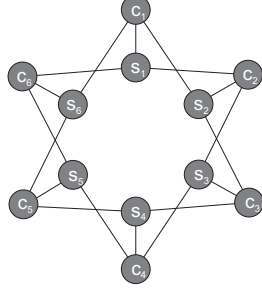
	λ_{c_i}/λ
c_1	1/9
c_2	2/9
c_3	1/9
c_4	2/9
c_5	1/9
c_6	2/9

The patience times are all exponentially distributed with mean 10. The service times distributions are given in the table below.

Service time distributions						
	c_1	c_2	c_3	c_4	c_5	c_6
s_1	U	E				P
s_2	P	U	E			
s_3		P	U	E		
s_4			P	U	E	
s_5				P	U	E
s_6	E				P	U

With:
$E \sim \mathrm{Exp}(1/4)$
$P \sim \mathrm{Pareto}(3,3)$
$U \sim \mathrm{U}(1,3)$

Only the *mean* service times are used by the design algorithms. The full distributions are used in the simulations.

</div>

In the designs for Example 3 we take as service fractions: $\beta_{s_j} = 1/6, \quad j = 1, \ldots, 6$. It then follows, by checking Conditions (1), that in each regime (ED, QED and QD) service is pooled.

	Required workforce																	
	ED regime						QED regime						QD regime					
λ	n_{s_1}	n_{s_2}	n_{s_3}	n_{s_4}	n_{s_5}	n_{s_6}	n_{s_1}	n_{s_2}	n_{s_3}	n_{s_4}	n_{s_5}	n_{s_6}	n_{s_1}	n_{s_2}	n_{s_3}	n_{s_4}	n_{s_5}	n_{s_6}
20	12	9	12	9	12	9	13	10	13	10	13	10	15	12	15	12	15	12
40	23	19	23	19	23	19	26	21	26	21	26	21	29	24	29	24	29	24
60	35	28	35	28	35	28	39	31	39	31	39	31	44	36	44	36	44	36
100	58	47	58	47	58	47	64	52	64	52	64	52	73	60	73	60	73	60
200	117	94	117	94	117	94	129	104	129	104	129	104	146	121	146	121	146	121

The theoretical matching rates are given in the table below. Note that these matching rates are the same for all three regimes (ED, QED and QD). This is caused by the fact that all customers have the same patience distribution and the same target waiting times, resulting in equal α_{c_i} and β_{s_j}. The simulated matching rates are also practically identical for all three regimes. In the tables we depict the averages over the three regimes, but the actual differences between the three simulated values and their averaged values are less than 0.001.

Theoretical matching rates (ED, QED, QD)						
r_{c_i,s_j}	c_1	c_2	c_3	c_4	c_5	c_6
s_1	0.028	0.069				0.069
s_2	0.041	0.084	0.041			
s_3		0.069	0.028	0.069		
s_4			0.041	0.084	0.041	
s_5				0.069	0.028	0.069
s_6	0.041				0.041	0.084
Simulated matching rates $\lambda = 20$ (Average ED, QED, QD)						
r_{c_i,s_j}	c_1	c_2	c_3	c_4	c_5	c_6
s_1	0.030	0.071				0.070
s_2	0.041	0.080	0.041			
s_3		0.070	0.030	0.071		
s_4			0.041	0.080	0.041	
s_5				0.070	0.030	0.071
s_6	0.041				0.041	0.080
Simulated matching rates $\lambda = 200$ (Average ED, QED, QD)						
r_{c_i,s_j}	c_1	c_2	c_3	c_4	c_5	c_6
s_1	0.028	0.069				0.069
s_2	0.041	0.084	0.041			
s_3		0.069	0.028	0.069		
s_4			0.041	0.084	0.041	
s_5				0.069	0.028	0.069
s_6	0.041				0.041	0.084

	Fraction of no wait and of no idling					
	ED regime		QED regime		QD regime	
λ	No waiting	No idling	No waiting	No idling	No waiting	No idling
20	0.044	0.950	0.278	0.709	0.862	0.135
40	0.014	0.985	0.398	0.594	0.929	0.071
60	0.003	0.997	0.351	0.642	0.974	0.026
100	0.000	1.000	0.314	0.681	0.994	0.006
200	0.000	1.000	0.352	0.644	1.000	0.000

6   Conclusion

In this paper we considered a parallel queueing system with multiple server types and multiple customer types, their compatibility described by a bipartite graph. This system has a general renewal arrival process, general customer-server dependent service times and operates under FCFS-ALIS service policy. As explained in the introduction, this system is relevant in many areas of applications, though at this level of generality, it is analytically intractable. Based on our intuitive understanding of how this system behaves under many server scaling and by exploiting exact results for the matching rates in the related FCFS infinite matching model, we proposed heuristic algorithms to calculate service work force levels required to meet quality of service parameters in three modes of operation: ED, QD and QED.

Extensive simulation confirmed that the algorithms are accurate and effective: they produce work force levels and, in case of differentiated service, a redesigned compatibility graph that meet targeted quality of service requirements. Moreover, the heuristic algorithms also appeared to work well when the required work force levels are not so large. As such, these algorithms

provide a valuable tool to support decisions on the design of multi-type service systems. The results suggest that our intuition, and in particular, the matching rates conjecture are correct. However, a rigorous justification is lacking at this point, and left as a major challenge for future research. A first step towards a rigorous analysis can be found in [30], treating many server scaling for the exponential "N" system.

References

[1] ADAN, I. J. B. F., BOON, M.A.A., WEISS, G. (2013) Design and evaluation of overloaded service systems with skill based routing, under FCFS policies, *Performance Evaluation*, 70(10):873-888.

[2] ADAN, I. J. B. F., BOON, M.A.A., BUSIC, A., MAIRESSE, J., WEISS, G. (2013) Queues with skill based parallel servers and a FCFS infinite matching model. *Performance Evaluation Review*, 41(3):22–24, 2013.

[3] ADAN, I. J. B. F., BUSIC, A., MAIRESSE, J., WEISS, G. (2015) Reversibility and further properties of FCFS infinite bipartite matching. arXiv:1507.05939 (2015).

[4] ADAN, I. J. B. F., WEISS, G. (2011) Exact FCFS matching rates for two infinite multi-type sequences. *Operations Research* **60** 475–489.

[5] ADAN, I. J. B. F., WEISS, G. (2014) A skill based parallel service system under FCFS-ALIS — steady state, overloads and abandonments, *Stochastc Systems* 4(1):250-299.

[6] ARMONY, M., WARD, A.R., (2010) Fair Dynamic Routing in Large-Scale Heterogeneous-Server Systems *Operations Research* **58**(3): 624–637.

[7] Armony, M., Ward, A. R. (2013). Blind fair routing in large-scale service systems with heterogeneous customers and servers. *Operations Research*, 61(1), 228-243.

[8] Bell, S. L., Williams, R. J. (2001). Dynamic scheduling of a system with two parallel servers in heavy traffic with resource pooling: asymptotic optimality of a threshold policy. *The Annals of Applied Probability*, 11(3), 608-649.

[9] CALDENTEY, R., KAPLAN, E. H., WEISS, G. (2009) FCFS infinite bipartite matching of servers and customers. *Advances in Applied Probability* **41** 695–730.

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

[11] Gans, N., Koole, G. and A. Mandelbaum (2003). Telephone call centers: tutorial, review, and research prospects. *Manufacturing Service Operations Management*, **5**(2) 79-141.

[12] Ghamami, S., Ward, A. R. (2013). Dynamic scheduling of a two-server parallel server system with complete resource pooling and reneging in heavy traffic: Asymptotic optimality of a two-threshold policy. *Mathematics of Operations Research*, 38(4):761-824.

[13] Green, L. (1985) A queueing system with genera-use and limited-use servers, *Operations Research* **33**:162–182.

[14] GURVICH, I., WHITT, W. (2009). Queue-and-idleness-ratio controls in many-server service systems. *Mathematics of Operations Research*, 34(2), 363-396.

20

[15] GURVICH, I., WHITT, W. (2010). Service-Level Differentiation in Many-Server Service System Via Queue-Ratio Routing. *Operations Research*, 58(2), 316-328.

[16] Harchol-Balter, M., Crovella, M. E., Murta, C. D. (1999). On choosing a task assignment policy for a distributed server system. *Journal of Parallel and Distributed Computing*, 59(2), 204-228.

[17] Harrison, J.M. and M.J. Lopez (1999). Heavy traffic resource pooling in parallel?server systems. *Queueing systems*, **33**(4), 339-368.

[18] Harrison, J. M., Zeevi, A. (2005). A method for staffing large call centers based on stochastic fluid models. *Manufacturing and Service Operations Management*, 7(1):20-36.

[19] Zeltyn, S., Mandelbaum, A. (2005). Call centers with impatient customers: many-server asymptotics of the M/M/n+ G queue. *Queueing Systems*, 51(3-4), 361-402.

[20] Nov, Y., Weiss, G., Zhang, H. (2016) Fluid Models of Parallel Service Systems under FCFS, Preprint.

[21] Rubino, M., Ata, B. (2009). Dynamic control of a make-to-order, parallel-server system with cancellations. *Operations Research*, 57(1), 94-108.

[22] Squillante, M. S., Xia, C. H., Yao, D. D., Zhang, L. (2001). Threshold-based priority policies for parallel-server systems with affinity scheduling. In American Control Conference, 2001. Proceedings of the 2001 4:2992-2999, IEEE.

[23] TALREJA, R., WHITT, W. (2007) Fluid models for overloaded multi-class many-service queueing systems with FCFS routing. *Management Science* **54** 1513–1527.

[24] Tezcan, T., Dai, J. G. (2010). Dynamic control of N-systems with many servers: Asymptotic optimality of a static priority policy in heavy traffic. *Operations Research*, 58(1):94-110.

[25] Tsitsiklis, J. N., Xu, K. (2012) On the power of (even a little) resource pooling, *Stochastic Systems*, 2:1-66

[26] Veeger, C.P.L., Etman, F.P. and Rooda (2008) Generating cycl time-throughput-product mix surfaces using effective process time based aggregate modeling. *Proc. 13th ASIM Conf., Berlin*, 519-529.

[27] Wallace R.B., Whitt, W. (2005) A staffing algorithm for call centers with skill-based routing. *Manufacturing and Service Operations Management* **7** 276–294.

[28] Whitt, W. (2006) Fluid models for multiserver queues with abandonments. *Operations research* **54** 37–54.

[29] Williams, R. J. (2000). On dynamic scheduling of a parallel server system with complete resource pooling. *Fields Institute Communications*, 28(49-71), 5-1.

[30] Zhan, Dongyuan and Weiss, G. (2016) Many server scaling of the N-system under FCFS-ALIS. In preparation