Queues with Skill Based Parallel Servers and a FCFS Infinite Matching Model

Ivo Adan, Marko Boon, Ana Bušić, Jean Mairesse, Gideon Weiss

Queues with Skill Based Parallel Servers and a FCFS Infinite Matching Model

Ivo Adan [*] Marko Boon[*] Ana Busic[†] Jean Mairesse[‡] Gideon Weiss[§]

1. A QUEUEING SYSTEM

We consider the following skill based parallel service queueing system: Customers are of types $\mathcal{C} = \{c_1, \dots, c_I\}$, servers are of types $\mathcal{S} = \{s_1, \dots, s_J\}$, and there is a bipartite graph G of compatiblities between $\mathcal{C}, \mathcal{S}$. The graph has arc $(i, j) \in G$ if server type s_j has the skill to serve customer type c_i. Customers arrive in independent Poisson streams of rates λ_i, and have absolutely continuous patience distributions F_i. There are n_j servers of type s_j, and the service times are *customer-server-type dependent*, the service of a customer of type c_i by a server of type s_j has a random duration distributed as G_{ij}, with average m_{ij}. We use the terminology of queueing theory throughout, but this type of system, with minor modifications, is useful in modeling call centers, manufacturing systems, organ transplants, multimedia servers, and cloud computing [8].

Performance of such systems is highly dependent on the operating policy. We focus here on first come first served (FCFS), where a server is assigned to the longest waiting compatible customer, coupled with assign longest idle server (ALIS), where a customer is assigned to the compatible server that has been idle for the longest time. FCFS-ALIS is widely used, because it is fair to both customers and servers, it is simple to implement, it requires little information about the parameters and the current state of the system, and it is robust under time varying conditions.

Our goal here is to develop a structured method to support the design and efficient operation of skill based parallel service systems under FCFS. At this level of generality such systems are highly intractable, no analytic results are expected, and asymptotics, e.g. using many server scaling are called for [12, 10, 7, 13]. We suggest an approximation based on a simplified look at the process — if we discard all arrival times, identities of servers, and processing times, and consider just the sequence of customers in order of arrivals, and the sequence of services in the order in which they become available, we have two infinite sequences of customer types and of server types. Making the simplifying assumption that these are i.i.d., and using FCFS, we obtain the FCFS infinite matching model. This model, which is of interest in its own right, is tractable, using symmetry and reversibility, as we describe in Section 2. From it we calculate matching rates at which customers of type c_i are served by servers of type s_j under FCFS. In Section 3 we use these matching rates in the design of our queueing system, with specified performance measures. We demonstrate the effectiveness of this approach by an illustrative example.

This paper answers some questions raised in [10, 6]. It summarizes recent results from in [1, 2] and current research in progress [3, 4].

2. FCFS INFINITE MATCHING

The model of first come first served (FCFS) infinite matching was introduced in [6], to answer some questions raised in [9, 10]. At that point the model looked highly intractable. It was followed by [1], where surprisingly it was found that the model is highly tractable, with a product form solution. A more general FCFS infinite matching model was described in [5]. The infinite matching model was found to be closely related to several queueing models with skill based parallel service [11, 2]. Here we present some novel results.

We consider two independent infinite random sequences: c^m, s^n, $m, n \in T$, c^m chosen i.i.d. from $\mathcal{C}, \mathcal{S}$, with respective probabilities α_i and β_j, and the bipartite compatibility graph G. T is either $\mathbb{N} = 1, 2, 3, \dots$ or $\mathbb{Z} = \dots, -2, -1, 0, 1, 2, 3 \dots$.

Notation: We shall refer to s^n as servers, to c^m as customers. Let $\mathcal{S}(c_i)$ be the server types compatible with c_i. For subsets of types C, S let $\mathcal{S}(C) = \bigcup_{c_i \in C} \mathcal{S}(c_i)$, and let $\alpha_C = \sum_{c_i \in C} \alpha_i$, $\beta_S = \sum_{s_j \in S} \beta_j$.

FCFS matching between the two sequences, for $T = \mathbb{N}$ is constructed as follows: s^1 is matched to the first compatible c^m. Thereafter s^n is matched to the first compatible c^m which was not matched to any of $s^1, \dots, s^{n-1}$. We refer to the sequences with the matchings as *the FCFS matching of* s^n, c^m, $m, n \in \mathbb{N}$.

Definition 1. *Let c^m be matched with s^n in the FCFS matching of the two sequences. The exchange transformation replaces c^m by $\tilde{s}^m = s^n$, and s^n by $\tilde{c}^n = c^m$ (see Figure 1).*

Assume all servers up to N were matched and exchanged.

[*] Department of Mathematics and Computer Science, Eindhoven University of Technology, P.O. Box 513, 5600 MB Eindhoven, the Netherlands, {*i.j.b.f.adan, m.a.a.boon*}*@tue.nl* Research supported in part by the Netherlands Organization for Scientific Research (NWO).

[†] INRIA/CNS, 23 avenue d'italie, CS 81321, 75214 Paris Cedex 13, France, *ana.busic@inria.fr.*

[‡] LIAFA, CNRS et Universite Paris 7, case 7014, 75205 Paris Cedex 13, France, *mairesse@liafa.univ-paris-diderot.fr.*

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

Let $\underline{M}$ be the location of the first unmatched customer, and $\overline{M}$ be the the location of the last matched customer, which was exchanged by a server. Consider the ordered sequence of servers and customers, $\mathfrak{z} = c^{\underline{M}}, \dots, \tilde{s}^{\overline{M}}$ with $\mathfrak{z} = 0$ if $N = \overline{M} = \underline{M} - 1$ (see Figure 1 where $N = 4, \underline{M} = 3, \overline{M} = 6$ and $\mathfrak{z} = (c^3, \tilde{s}^4, c^5, \tilde{s}^6)$). We define a Markov chain Z_N with state $\mathfrak{z}$, where Z_{N+1} will be obtained when s^{N+1} is matched and exchanged.

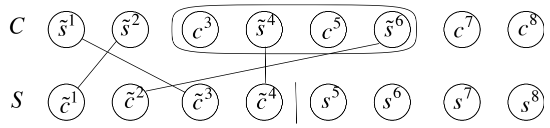


Figure 1: FCFS matching with exchange transformation, $Z_4 = \mathfrak{z} = (c^3, \tilde{s}^4, c^5, \tilde{s}^6)$

THEOREM 1. *The stationary distribution of Z_N is given by*

$$\pi(\mathfrak{z}) = B \prod_{i=1}^{I} {\alpha_i}^{\#c_i} {\beta_j}^{\#s_j} \tag{1}$$

where $\#c_i$ is the number of customers of type c_i and $\#s_j$ is the number of servers of type s_j in $\mathfrak{z}$, and B is the normalizing constant.

A necessary and sufficient condition for finite B and ergodicity is *complete resource pooling* defined by:

$$\alpha_C < \beta_{\mathcal{S}(C)}, \quad \text{for all non trivial subsets } C \subset \mathcal{C}. \tag{2}$$

We note that the clean and beautiful simplicity of this multi-Bernoulli product form of π masks the true complexity of the process, which is revealed if we examine what states are allowed: we note that each c^m in the sequence $\mathfrak{z}$ must be incompatible with all later $\tilde{s}^n$. The proof of Theorem 1 uses time reversal, and the symmetry in the process. It leads us to make the following conjecture:

CONJECTURE 1 (REVERSIBILITY). *Let s^n, c^m, $m, n \in \mathbb{Z}$, be two independent i.i.d. sequences, with FCFS matching between them. Let $\tilde{s}^m, \tilde{c}^n$ be obtained by the exchange transformation, with the corresponding matching. Then $\tilde{s}^m, \tilde{c}^n$ are independent i.i.d. sequences, and the matching between them is FCFS in the reversed time direction.*

We can in fact show that $\tilde{s}^m, \tilde{c}^n$ are FCFS in the reversed time direction, and that each of these sequences is i.i.d, but we are still missing the proof that the sequence $\tilde{s}^m$ is independent of $\tilde{c}^n$.

From π one can obtain various marginal distributions of interest, and various performance measures. Most relevant for our queueing system are the matching rates r_{c_i,s_j} at which customers of type c_i are served by servers of type s_j. The formula for that is:

$$r_{c_i,s_j} = \beta_j \sum_{\mathcal{P}_J} B \prod_{k=1}^{J-1} (\beta_{(k)} - \alpha_{(k)})^{-1} \left(\sum_{k=1}^{J-1} \phi_k \frac{\alpha_{(k)}}{\beta_{(k)} - \alpha_{(k)}\chi_k} \right.$$

$$\left. \prod_{l=1}^{k-1} \frac{\beta_{(l)} - \alpha_{(l)}}{\beta_{(l)} - \alpha_{(l)}\chi_l} + \frac{\phi_J}{\phi_J + \psi_J} \prod_{l=1}^{J-1} \frac{\beta_{(l)} - \alpha_{(l)}}{\beta_{(l)} - \alpha_{(l)}\chi_l} \right), \tag{3}$$

where the summation is over $\mathcal{P}_J$, the set all the permutations of $\mathcal{S}$. For more details on the notation used see [1].

If complete resource pooling does not hold the system decomposes uniquely to subsystems, $(\mathcal{C}^{(1)}, \mathcal{S}^{(1)}), \dots, (\mathcal{C}^{(L)}, \mathcal{S}^{(L)})$ defined recursively, the first of them being:

$$\mathcal{C}^{(1)} = \arg\min_{C \subseteq \mathcal{C}} \frac{\beta_{\mathcal{S}(C)}}{\alpha_C}, \quad \mathcal{S}^{(1)} = \mathcal{S}(\mathcal{C}^{(1)}).$$

3. A DESIGN ALGORITHM

We now present an algorithm for the design of the queuing system of Section 1. We assume that we want to operate it in ED (efficiency driven) mode, with overloaded servers, and controlled abandonments. We let:

Input: The compatibility graph G, the arrival rates λ_i, the patience distributions F_i, and the service-time distributions G_{ij}, of which we only need the means m_{ij}.

Quality of service decision: Partition customer types into $\mathcal{C}^{(l)}$, $l = 1, \dots, L$, where higher l implies more preferential service. Let $\mathcal{S}^{(l)} = \mathcal{S}(C^{(l)}) \setminus \bigcup_{l' < l} \mathcal{S}^{(l')}$. Specify target waiting times $W_1 > W_2 > \cdots > W_L$.

Continue to design each subsystem separately. We drop the superscripts and use notation $c_i \in \mathcal{C}$, $s_j \in \mathcal{S}$, W within each subsystem:

Division of labor decision: Specify the fraction of services performed by each type of server, β_j.

Calculations: Define total service rate $\hat{\lambda} = \sum \lambda_i(1 - F_i(W))$ and let $\alpha_i = \lambda_i(1 - F_i(W))/\hat{\lambda}$. Verify complete resource pooling within the subsystem, and compute the matching rates r_{c_i,s_j} using (3). Calculate required staffing:

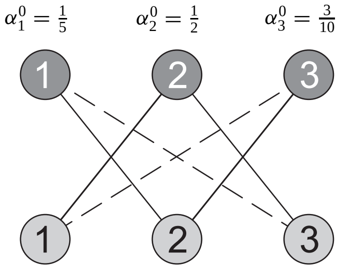
$$n_j = \hat{\lambda} \sum_{c_i \in \mathcal{C}(s_j)} r_{c_i,s_j} m_{ij} \tag{4}$$

Output: Number of servers of each type, n_j, the matching rates r_{ij}, the cut-off waiting times W_i, and the abandonment rates $F_i(W_i)$.

The idea behind the algorithm is that if λ is large relative to the typical waiting time W, there will be many customers in the system at all times, with a fraction $F(W)$ of abandonments, and the types of successive customers which will receive service will still be i.i.d. α. At the same time, with large λ the number of servers n_j will be large, and the instants of service completions will be a superposition of many almost stationary, almost independent, point processes which should be approximately Poisson. As a result types of successive available servers will be i.i.d. β.

We now test our algorithm on simple graph G, to show that it performs quite well with moderate number of servers. $\mathcal{C}$, $\mathcal{S}$ and G are shown in Figure 2. We use the following input data. The proportion of arrivals for the c_i are $\alpha^0 = (0.2, 0.5, 0.3)$. The patience distribution for all customers is exponential with mean 10, so $F_i(t) = 1 - e^{-0.1t}$. The service time distributions $G_{i,j}$ are as follows:

Service time distributions : Means			
$G_{i,j}$	s_1	s_2	s_3
c_1		pareto(3, 3) :4.5	pareto(2, 3) :3
c_2	exp(0.2) :5		exp(0.125) :8
c_3	uniform(2, 6) :4	uniform(1, 5) :3	



$\alpha_1^0 = \frac{1}{5}$
$\alpha_2^0 = \frac{1}{2}$
$\alpha_3^0 = \frac{3}{10}$
1
2
3
1
2
3

Figure 2: The "almost complete" network. Nodes on the top row represent customer types, and the ones on the bottom row server types. The dashed arcs are present in the case of complete resource pooling, but vanish in the case of incomplete resource pooling.

We explore two designs. The first proivdes equal service quality to all types of customers, the second design gives preferential service to types c_1, c_3 over type c_2. We ran extensive simulations, of 100 runs with 100,000 customers each, for every experiment. Some of our results are:

First design: equal service quality

We choose common waiting time $W = 1$, so that abandonment rate is targeted as $\theta_i = F_i(W_i) = 1 - e^{-0.1} \approx 0.095$. This leaves $\alpha = \alpha^0 = (0.2,\ 0.5,\ 0.3)$. We choose $\beta = (0.3, 0.4, 0.3)$. This satisfies complete resource pooling. The calculated matching rates are

$$[r_{ij}] = \begin{bmatrix} 0 & \frac{176}{1115} & \frac{47}{1115} \\ \frac{54}{223} & 0 & \frac{115}{446} \\ \frac{129}{2230} & \frac{54}{223} & 0 \end{bmatrix}.$$

and the required staffing, as a function of the total arrival rate λ works out at:

$$n(\lambda) \approx \lambda(1.305, 1.300, 1.981).$$

We use several values of λ. As expected the results improve as λ increases, but are already quite good for a moderate number of servers.

Second design: preferential service

Here we use a waiting time of $W_1 = 2$, for $\mathcal{C}^{(1)} = \{c_2\}$, and waiting time $W_2 = 0.5$ for $\mathcal{C}^{(2)} = \{c_1, c_3\}$. This partitions the servers into $\mathcal{S}^{(1)} = \{s_1, s_3\}$, and $\mathcal{S}^{(2)} = \{s_2\}$ For $\mathcal{S}^{(1)}$ we choose $\beta_1 = \beta_3 = 0.5$, while of course for $\mathcal{S}^{(2)}$ $\beta_2 = 1$. The staffing calculations lead to

$$n(\lambda) \approx \lambda(1.023, 1.712, 1.637).$$

Results of the simulation are given in the following tables.

4. REFERENCES

[1] Adan, I.J.B.F., Weiss, G. (2012) *Operations Research* 60(2):475–489.

[2] Adan, I.J.B.F., Weiss, G. (2012) *Eurandom Report* 2012-011.

[3] 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. In preparation.

Simulated mean waiting times and abandonment rates

λ	n	$E[W_1]$	$E[W_2]$	$E[W_3]$	θ_1	θ_2	θ_3
10	$(13, 13, 20)$	0.870	1.226	0.884	.086	.119	.088
30	$(39, 39, 59)$	0.924	1.110	0.928	.089	.105	.089
50	$(65, 65, 99)$	0.929	1.053	0.932	.089	.100	.089
100	$(130, 130, 198)$	0.956	1.025	0.958	.090	.096	.090
Target		1.000	1.000	1.000	.095	.095	.095

Matching rates

	Simulated $\lambda = 10$			Simulated $\lambda = 100$			Target		
$r_{i,j}$	s_1	s_2	s_3	s_1	s_2	s_3	s_1	s_2	s_3
c_1		.153	.050		.158	.043		.158	.042
c_2	.234		.257	.240		.258	.242		.258
c_3	.067	.238		.060	.242		.058	.242	

Simulated densities of the waiting times

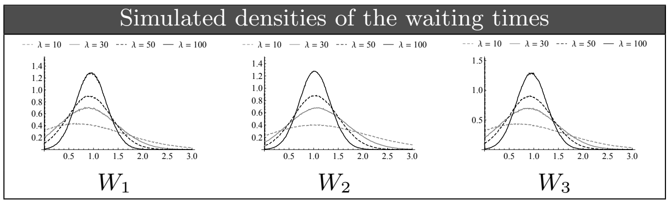


Table 1: Simulation results for equal service level.

Simulated mean waiting times and fractions of abandonment

λ	n	$E[W_1]$	$E[W_2]$	$E[W_3]$	θ_1	θ_2	θ_3
10	$(10, 17, 16)$	0.685	2.404	0.687	.068	.221	.069
30	$(31, 51, 49)$	0.596	2.011	0.596	.059	.184	.059
50	$(51, 86, 82)$	0.504	2.017	0.505	.049	.183	.050
100	$(102, 171, 164)$	0.525	2.010	0.526	.051	.180	.051
Theoretical		0.500	2.000	0.500	.050	.180	.050

Table 2: Simulation results for preferential treatment.

[4] Adan, I.J.B.F., Busic, A., Mairesse, J., Weiss, G. (2013) Reversibility and further properties of FCFS infinite matching. In preparation.

[5] Busic, A., Gupta, V. and Mairesse, J. (2010) arxiv:1003.3477v1 [cs.DM], *Advances in Applied Probability*, to appear.

[6] Caldentey, R., Kaplan, E.H., Weiss, G., (2009) *Advances in Applied Probability* 41:695-730.

[7] Gurvich I., Whitt, W. (2010) *Operations Research* 58:316–328.

[8] Harchol-Balter, M. (2013) *Performance Modeling and Design of Computer Systems: Queueing Theory in Action* Cambridge University Press, UK.

[9] Kaplan, E.H. (1988) *Decision Sciences* 19:383–391.

[10] Talreja, R., Whitt, W. (2007) *Management Science* 54:1513–1527.

[11] Visschers, J., Adan, I.J.B.F., Weiss, G. (2012) *Queueing Systems* 70:269–298.

[12] Wallace R.B., Whitt, W. (2005) *Manufacturing and Service Operations Management* 7:276–294.

[13] Whitt, W. (2006) *Operations Research* 54:37–54.