Maximizing throughput in zero-buffer tandem lines with constrained flexible servers

Mohammad H. Yarmand and Douglas G. Down

Department of Computing and Software
McMaster University
{yarmanmh,downd}@mcmaster.ca

Abstract. For tandem queues with no buffer spaces and both dedicated and flexible servers, we study how flexible servers should be assigned to maximize the throughput. We focus on systems in which flexible servers are constrained to serve at specific stations. With three stations and one or two constrained flexible servers, we completely characterize the optimal policies and compare throughput improvement with systems in which flexible servers are not constrained. Using numerical experiments, we discuss the impact of constrained flexibility on performance and dimensioning of systems.

Keywords: Server Allocation; Zero Buffer; Tandem Queues; Constrained Flexible Servers

1 Introduction

Consider a tandem queueing network with $N \geq 2$ stations, $M \geq 1$ dedicated servers, and $F \geq 1$ flexible servers. At any given time, each station can be assigned multiple servers and each server can work only on one job, and a job may have at most one server assigned to it. Assume that the service times of each job at station $i \in \{1, \dots, N\}$ are independent and identically distributed exponential random variables with rate μ_i, i.e. the service rate of each server at the i^{th} station is only dependent on the station.

We assume that dedicated servers are already assigned to stations. We are interested in determining the dynamic assignment policy for flexible servers that maximizes the long-run average throughput. For simplicity, we assume that the travel times for jobs to progress and also the travel times for flexible servers to move between stations are negligible. We also assume there is an infinite supply of jobs in front of the first station and infinite space for jobs completed at the last station. There are no buffer spaces between stations – blocking occurs after service (manufacturing blocking).

In this paper, we study situations where flexible servers are constrained to operate between two adjacent stations. The motivation for this study is that in practice, there might be situations where moving flexible servers among all stations is not possible (or if possible, it might be costly).

The concept of "hand-off" for flexible servers was introduced in [9]. Hand-off happens when a flexible server passes the job it is serving to a dedicated server at the same station. Although it is possible to perform hand-off at any time, we let it occur only in the following two cases. When a station has a busy flexible server and a free dedicated server, the flexible server can pass its job to the dedicated server and become free. When a station has a busy flexible server and a blocked dedicated server, jobs can be swapped between the two servers. In either of the two cases, we say a hand-off has taken place. With the insights gained from [9] (Theorems 1 and 2), our problem is significantly simplified, as we are able to restrict our analysis to policies that perform hand-offs as described above and are non-idling. (Note that is such hand-offs were not allowed, then optimal policies may allow idling, to avoid flexible servers being blocked for extended periods of time.)

Any allocation policy should define appropriate actions when blocking or starvation occurs. In cases where there are multiple blocked or starved servers, policies should prioritize resolving blocking or starvation of the involved servers. We show that for systems with three stations, with balanced service rates, the optimal policy clears blocking from the end of the line to the beginning and avoids starving servers.

The literature on tandem lines with multiple servers and finite buffers is not large, see the discussion in van Vuuren et al. [7]. We have previously studied related problems in the assignment of dedicated and flexible servers in zero-buffer tandem lines. In [8], we studied the problem of how to assign dedicated servers in such settings, and in [9], we examined the problem of how to coordinate dedicated and flexible servers, where the locations of where flexible servers could work was not constrained. Constraining the servers in this manner is the focus of the current paper.

Our work has been motivated by related work in two application domains: hospital bed management and assembly line design. In the hospital bed setting, Bekker et al. [1] discuss a number of issues with respect to how to manage flexible hospital beds. In particular, they develop the insight that full flexibility is beneficial for smaller systems, with less flexibility required for larger systems. Our work complements this, as rather than studying the impact of the number of flexible servers (as in [1]), we explore the scenario when movement of servers is constrained. We imagine that such constrained flexibility may be appropriate in such a setting – it may be problematic to move a flexible bed to all locations in a ward/hospital. Our main conclusion is that there can still be significant benefit, even if flexibility is constrained. For further discussion of the bed management problem, see de Bruin et al. [2], Green [3], and Hall [4]. In terms of assembly line design, an example of a zero-buffer tandem line can be seen in Hu et al. [5], where a subset of fixtures can be reconfigured.

This paper is organized as follows. In Section 2 we use Markov Decision Process theory to derive the optimal policy for tandem lines with three stations, a dedicated server at each station, and one or two constrained flexible servers. We further show how to employ the Policy Iteration algorithm to construct the optimal policy. In Section 3 we examine larger systems and discuss the impact of constrained flexibility. Finally, Section 4 concludes the paper and discusses future work.

2 Tandem lines with three stations

In this section, we study the following four cases: when there is a flexible server which only moves between the first and second stations; when there is a flexible server which only moves between the second and third stations; when there are two flexible servers, one moving only between the first and second stations and the other one moving only between the second and third stations; and finally when there are two flexible servers that can move among all of the stations (which we call the fully flexible case). We provide a brief comparison between the last two cases. Before looking at these cases, we first discuss the framework for such problems with an arbitrary number of servers and stations. The performance metric that we seek to optimize is the throughput.

We use a Markov Decision Process (MDP) model. For our controlled continuous-time Markov chain (CTMC), let S, A, and A_s represent the state space, the action space, and the action space conditioned on the Markov chain being in state $s \in S$, respectively. We assume that the system employs hand-offs in the following manner. If a station has a busy flexible server and an idle dedicated server, then the busy flexible server can pass its job to an idle dedicated server. Also, when a station has a busy flexible server and a blocked dedicated server, the jobs are swapped between the servers. We also assume that servers are non-idling. Theorems 1 and 2 from [9] can be applied to show that an optimal policy performs hand-offs and is non-idling. With these assumptions, the choice of a state is simplified, as we do not need to keep track of which servers (dedicated or flexible) are busy at each station. Thus our choice of state $s \in S$ is defined by the tuple

$$s = (x_1, y_1, \ \ldots, \ x_i, y_i, \ \ldots, \ x_N)$$

where x_i is the number of busy servers in station i and y_i is the number of blocked servers in station i. Note that we do not need to include y_N in the state, as no servers can be blocked at the last station. We uniformize the CTMC (see Lippman [6]) to convert the continuous time problem to discrete time. The normalization constant that we will employ is denoted by q, and is defined below.

Constructing the transition matrices is a two stage process. One needs to start from the initial state (s) and follow possible actions (a) to determine the new states (s'). The state s' is an intermediate state which does not appear in the transition matrix. The transition between s and s' is immediate. From there, it is possible to follow further transitions and reach new states (s'') with the probabilities defined in the transition matrix. We have:

$$s \xrightarrow{a} s' \xrightarrow{P_a} s''$$

that is reflected in the transition matrix as $P_a(s, s'') = \frac{\gamma_{a,s,s''}}{q}$ where $q = \max_{s \in S, a \in A_s} \sum_{s'' \in \{S-s\}} \gamma_{a,s,s''}$ and $\gamma_{a,s,s''}$ is the transition rate from s' to s'', and s' is the state transitioned to from state s using action a.

The size of the action space is $|A| = \binom{F+N-1}{N-1}$. An action $a \in A$ is denoted by $a_{i_1 i_2 \cdots i_K}$, where i_j is the location of the jth flexible server.

2.1 Constrained servers

We first consider a tandem line with three stations and a dedicated server at each station. Assume a flexible server exists which can only move between the first and second stations. Theorem 1 describes the optimal policy. In the interest of space, we have not included the proof, as it is similar to the proof of Theorem 3. (Theorem 3 is the key analytic result, so as such it is the one result for which we provide a proof outline.)

Theorem 1. *The optimal policy prioritizes clearing blocking. It performs hand-off and allocates the flexible server to the first station, whenever possible. Otherwise the flexible server is assigned to the second station. The optimal policy is independent of the service rates.*

Next, we consider a tandem line with three stations and a dedicated server at each station. Assume a flexible server exists which can only move between the second and third stations. Theorem 2 describes the optimal policy. As for Theorem 1, its proof is not provided.

Theorem 2. *The optimal policy prioritizes clearing blocking. It performs hand-off and allocates the flexible server to the second station, whenever possible. Otherwise the flexible server is assigned to the third station. The optimal policy is independent of the service rates.*

Comparing the policies described in Theorems 1 and 2 with the fully flexible case, described in [9], all policies have similar structures. They clear blocking and send the flexible server to upstream stations whenever possible.

We now move to the case where there are flexible servers between both stations. Unfortunately, the optimal server assignment becomes more complicated. In general, the optimal policy is rate dependent. However, in the case where the service rates are identical across all stations, the optimal policy again clears blocking (here from the end to the beginning). This would suggest that such a policy would be optimal when the service rates are near balanced across stations. Theorem 3 provides the optimal policy for equal service rates in a zero-buffer tandem line with three stations, a dedicated server at each station, and two flexible servers, one constrained between the first and second stations, the second constrained between the second and third stations.

Theorem 3. *Suppose that $\mu_1 = \mu_2 = \mu_3 = \mu$. The optimal policy clears blocking from the end to the beginning (i.e. the policy prioritizes clearing any blocking at the second station over clearing blocking at the first station, when possible).*

Proof. In the proof, we leave μ_1, μ_2, and μ_3 general, as we will later comment on the dependence of the optimal policy on these rates. To be able to construct the discrete-time MDP, the normalization factor is $q = \mu_1 + \mu_2 + \mu_3 + \max\{\mu_1 + \mu_2, \mu_1 + \mu_3, \mu_2 + \mu_3, 2\mu_2\}$. The MDP details are as follows.

$$A = \{a_{12}, a_{13}, a_{22}, a_{23}\}$$

$S = \{(2,0,2,0,1), (2,0,2,0,0), (2,0,1,1,1), (2,0,1,0,2), (2,0,1,0,1), (2,0,1,0,0), (2,0,0,1,2), (2,0,0,1,1), (2,0,0,0,2), (2,0,0,0,1), (2,0,0,0,0), (1,1,2,0,0), (1,1,1,0,2), (1,1,1,0,1), (1,1,1,0,0), (1,1,0,1,2), (1,1,0,1,1), (1,0,3,0,1), (1,0,3,0,0), (1,0,2,1,1), (1,0,2,0,2), (1,0,2,0,1), (1,0,2,0,0), (1,0,1,2,1), (1,0,1,1,2), (1,0,1,1,1), (1,0,0,2,2), (1,0,0,2,1), (0,1,3,0,1), (0,1,3,0,0), (0,1,2,1,1), (0,1,2,0,2), (0,1,2,0,1), (0,1,2,0,0), (0,1,1,2,1), (0,1,1,1,2), (0,1,1,1,1), (0,1,1,0,2), (0,1,1,0,1), (0,1,0,2,2), (0,1,0,2,1), (1,0,1,0,2), (1,0,1,0,1), (1,0,1,0,0), (1,0,0,1,2), (1,0,0,1,1), (1,0,0,0,2), (1,0,0,0,1), (1,0,0,0,0)\}$

Let $\bar{s}$ represent the index of a state $s \in S$, according to the order represented above. In what follows, we will use the label $\bar{s}$ interchangeably with the corresponding state s, i.e. $A_1 = A_{(2,0,2,0,1)}$.

$$A_{\bar{s}} = \begin{cases} a_{12} & \text{for } \bar{s} = 1,2 \\ a_{13} & \text{for } \bar{s} = 4,7,9 \\ a_{22} & \text{for } \bar{s} = 18,19,29,30 \\ a_{23} & \text{for } \bar{s} = 21,25,27,32,36,40 \\ \{a_{12}, a_{13}\} & \text{for } \bar{s} = 3,5,6,8,10,11 \\ \{a_{12}, a_{22}\} & \text{for } \bar{s} = 12 \\ \{a_{13}, a_{23}\} & \text{for } \bar{s} = 13,16,38,42,45,47 \\ \{a_{22}, a_{23}\} & \text{for } \bar{s} = 20,24,31,35 \\ \{a_{12}, a_{22}, a_{23}\} & \text{for } \bar{s} = 22,23,33,34,41 \\ \{a_{13}, a_{22}, a_{23}\} & \text{for } \bar{s} = 28 \\ \{a_{12}, a_{13}, a_{22}, a_{23}\} & \text{for } \bar{s} = 14,15,17,26,37,39,43,44,46,48,49 \end{cases}$$

Our candidate optimal policy d_0 is:

$$d_0(\bar{s}) = \begin{cases} a_{12} & \text{for } \bar{s} = 1,2,5,6,8,10,11,14,15,22,23,39,43,44,48,49 \\ a_{13} & \text{for } \bar{s} = 3,4,7,8,9,17,26,28,37,42,45,46,47 \\ a_{22} & \text{for } \bar{s} = 12,18,19,29,30,33,34 \\ a_{23} & \text{for } \bar{s} = 13,16,20,21,24,25,27,31,32,35,36,38,40,41 \end{cases}$$

and associated reward function is:

$$r_{d_0}(\bar{s}) = \begin{cases} 0 & \text{for } \bar{s} = 2, 6, 11, 12, 15, 19, 23, 30, 34, 39, 44, 49 \\ \mu_3 & \text{for } \bar{s} = 1, 3, 5, 8, 10, 14, 17, 18, 20, 22, 24, 26, 29, 31, 33, 35, 37, 43, 46, 48 \\ 2\mu_3 & \text{for } \bar{s} = 4, 7, 9, 13, 16, 21, 25, 27, 28, 32, 36, 38, 40, 41, 42, 45, 47 \end{cases}$$

Again in the interests of space, we explicitly provide only the first row of the transition matrix $P_{a_{12}}(s, s'')$. It is a straightforward excercise to calculate the remaining rows of $P_{a_{12}}(s, s'')$, as well as the other matrices $P_{a_{i_1 i_2}}(s, s'')$.

$$\begin{aligned} P_{a_{12}}(1, 18) &= \frac{2\mu_1}{q} \\ P_{a_{12}}(1, 4) &= \frac{2\mu_2}{q} \\ P_{a_{12}}(1, 2) &= \frac{\mu_3}{q} \\ P_{a_{12}}(1, 1) &= \frac{q - 2\mu_1 - 2\mu_2 - \mu_3}{q} \\ P_{a_{12}}(1, k) &= 0,\ k \neq 1, 2, 4, 18. \end{aligned}$$

To prove the optimality of d_0, we need to show that $d_1(s) = d_0(s)$, where:

$$d_1(s) = \text{argmax}_{a \in A_s}\{r(s, a) + \sum_{j \in S} P_a(s, j)h_0(j)\}, \forall s \in S,$$

is the result of one iteration of the Policy Iteration algorithm. This is equivalent to showing that for all s

$$r(s, a) + \sum_{j \in S} P_a(s, j)h_0(j) - \left(r(s, d_0(s)) + \sum_{j \in S} P_a(s, j)h_0(j) \right) \leq 0. \quad (1)$$

When $\mu = \mu_1 = \mu_2 = \mu_3$, we directly verified inequality that (1) holds for each s and therefore $d_0(s)$ is optimal. The algebra is straightforward (but somewhat lengthy). □

If the service rates are arbitrary, there are several states where the optimal action is rate independent. These states are (2,0,0,0,1), (2,0,0,0,0), (1,0,0,0,1), (1,0,0,0,0) with a_{12}; (1,1,2,0,0), (0,1,2,0,1), (0,1,2,0,0) with a_{22}; and (1,1,1,0,2), (1,1,0,1,2), (0,1,1,2,1), (0,1,1,0,2) with a_{23}.

The remaining states have optimal actions which depend on the service rates. For example, for state (2,0,1,1,1), action a_{13} is the optimal action for equal service rates (and also rates sufficiently close to each other). Action a_{12} becomes the optimal choice when the service rate at the first station is much faster than the second station and a constrained flexible server will be required at the second station, but at the same time the third station is fast enough to clear the blocking with its dedicated server. Looking at an extreme set of rates like $\mu_1 = 20, \mu_2 = 1, \mu_3 = 17$ makes it easier to comprehend why clearing blocking is not the immediate chosen action, but this also occurs for less extreme rates.

Unfortunately, explicitly characterizing the boundary between the optimality of actions a_{13} and a_{12} appears difficult in general.

Another example is state (0,1,1,1,1), where action a_{13} is the optimal action for equal service rates. Action a_{23} becomes the optimal choice when service rates are skewed such that the first station is much slower than the second and third stations, in which case admitting jobs becomes the priority. An example set of rates where this holds is $\mu_1 = 1, \mu_2 = 4, \mu_3 = 5$.

2.2 Two fully flexible servers

Consider a tandem line with three stations and a dedicated server at each station. Assume two flexible servers exist that can move between all stations. The optimal policy is rate dependent. When the service rates are equal, Theorem 4 describes the optimal policy. Its proof is similar to the proof of Theorem 3.

Theorem 4. *When $\mu_1 = \mu_2 = \mu$, the optimal policy clears blocking from the end to the beginning. The policy sends the flexible servers to the first station, whenever possible.*

Now we compare the structure of the optimal policies with two flexible servers. When the service rates are equal, in both constrained and fully flexible cases, the optimal policies prioritize clearing blocking. Both of the policies coordinate allocations such that flexible servers are freed to send them to the first station. Both of the policies are rate dependent.

In terms of throughput results, Table 1 compares the two policies for a number of workloads. In this table, the entries in the header row represent service rate vectors and the other table entries are throughput values.

Rates / Flexibility	$(1,1,1)$	$(2,1,1)$	$(1,2,1)$	$(1,1,2)$
Dedicated (5)	0.8873	1.2812	1.1186	1.2888
Dedicated (6)	1.3314	1.4779	1.5597	1.4785
Constrained	1.3606	1.6355	1.4974	1.6027
Fully flexible	1.4600	1.6407	1.7252	1.6435
All fully flexible	1.6667	2.0	2.0	2.0

Table 1. Comparison of throughputs for different flexibility situations

The first row considers an allocation where there is a dedicated server at each station and two of the stations have one extra dedicated server each. For each column of the first row, the highest throughput resulting from different possible server allocations is represented. The second row represents an allocation where there are two dedicated servers at each station. The third and fourth rows give throughput results for systems with a dedicated server at each station, the third row also has two constrained servers as in Theorem 3, the fourth row has two fully flexible servers as in Theorem 4. The final row has all five servers fully flexible.

Examining Table 1 in more detail, we see that moving to a system with a single constrained server between each pair of servers achieves a significant percentage of the gains made by making servers flexible (compare rows one, three and four). So, without increasing the total number of servers, significant performance gains are possible, even with constrained flexibility. Also, we see that with two constrained servers, we essentially have the same throughput with five servers as with six dedicated servers, a potential savings in required resources.

3 Larger systems

In this section, we examine how our insights for $N = 3$ stations extends to systems with both a larger number of stations and servers. The numerical results in this section are obtained from simulation, where each simulation is a long run (100 million departures).

We begin with systems where there is one constrained flexible server between each pair of stations. The service rates at all stations are equal to one. In Table 2, the first column, N, shows the number of stations. In the case of dedicated servers (represented in the second column), there are two dedicated servers at each station; in the case of constrained flexible servers (represented in the third column), each station has a dedicated server and there are $N - 1$ constrained flexible servers, one between each consecutive pair of stations (so there is one less server in total than for the dedicated system); in the case of fully flexible servers (represented in the fourth column), there are $N-1$ fully flexible servers and each station has one dedicated server. For the fully flexible server system (here and throughout this section), we use "Policy I" as described in [9]: "clear blocking from end to beginning only if it does not cause starvation in the $\lfloor \frac{2}{3}N \rfloor$ previous stations; uses hand-off", which was shown to perform well for longer lines. Looking at Table 2, constrained flexibility offers roughly 40% of the throughput improvement that full flexibility provides. Note that the constrained flexibility appears to make the resulting throughput relatively insensitive to N. As a result, the throughput gain (as a percentage) appears to be a slightly increasing function of N.

N	**dedicated**	**const flx**	**fully flx**
4	1.2420	1.3438	1.6001
5	1.1939	1.3434	1.6520
8	1.1192	1.3514	1.7286
15	1.0610	1.3561	1.7868
30	1.0282	1.3585	1.8213

Table 2. Throughput for larger homogeneous systems

We extend our study to systems where there is more than one dedicated server per station. Table 3 considers a configuration with three stations. The third column of this table shows the number of constrained flexible servers among each pair of stations. For example (2,2,0) means that there are two constrained flexible servers between the first and second stations and two constrained flexible servers between the second and third stations. Comparing the first and eighth rows, a configuration with 21 dedicated servers and six constrained flexible servers has throughput close to a configuration with 30 dedicated servers, meaning constrained flexibility can compensate for a reduction of three servers. Also, comparing the eighth and ninth rows, it appears that when there are multiple servers per station, the throughput difference between constrained and full flexibility is less compared to configurations which have one or two servers per stations.

N	**Dedicated Alloc**	**Const flx Alloc**	**Throughput**
30	(10, 10, 10)		8.31077
30	(9,10,9)	(1,1,0)	8.63140
29	(9,9,9)	(1,1,0)	8.39741
30	(8,9,9)	(2,2,0)	8.87239
30	(9,9,8)	(2,2,0)	8.86033
30	(8,8,8)	(3,3,0)	9.14992
28	(8,8,8)	(2,2,0)	8.35096
27	(7,7,7)	(3,3,0)	8.22025
27	(7,7,7)	$F = 6$ (fully)	8.63805

Table 3. Throughput for configurations with $N = 3$ and multiple dedicated and flexible servers per station

To give an idea of how the effect scales, we examine a system with 10 stations, see Table 4. We see that with $N = 91$ or 92 servers and 18 of these being (constrained) flexible servers, we can achieve close to the same throughput as with $N = 100$ servers, all dedicated. In general, it appears that we can reduce the number of servers by approximately 10 percent by adding a small amount of constrained flexibility. Finally, the gap between constrained flexibility and full flexibility increases with the number of stations. This is not at all surprising, as full flexibility allows the flexible servers to work at all of the stations, providing more opportunities to leverage them.

4 Conclusion

Based on our observations, optimal policies under constrained flexibility have a similar structure to optimal policies under full flexibility. All of the policies perform hand-off such that the flexible server is freed to send it to upstream stations. They also clear blocking if any exists. Also as expected, the throughput improvement under constrained flexibility is less compared to full flexibility.

N	Allocations	Throughput
100	dedicated: (10,10,10,10,10,10,10,10,10,10)	7.65041
99	dedicated: (9,9,9,9,9,9,9,9,9,9) const flx: (1,1,1,1,1,1,1,1,1,0)	8.06876
99	dedicated: (9,9,9,9,9,9,9,9,9,9) fully flx: 9	8.81186
89	dedicated: (8,8,8,8,8,8,8,8,8,8) const flx: (1,1,1,1,1,1,1,1,1,0)	7.19769
89	dedicated: (8,8,8,8,8,8,8,8,8,8) fully flx: 9	7.93998
98	dedicated: (8,8,8,8,8,8,8,8,8,8) const flx: (2,2,2,2,2,2,2,2,2,0)	8.30885
98	dedicated: (8,8,8,8,8,8,8,8,8,8) fully flx: 18	9.26543
88	dedicated: (7,7,7,7,7,7,7,7,7,7) const flx: (2,2,2,2,2,2,2,2,2,0)	7.41644
91	dedicated: (7,7,7,7,7,8,7,8,7,8) const flx: (2,2,2,2,2,2,2,2,2,0)	7.61384
92	dedicated: (7,7,7,8,7,8,7,8,7,8) const flx: (2,2,2,2,2,2,2,2,2,0)	7.69727
88	dedicated: (7,7,7,7,7,7,7,7,7,7) fully flx: 18	8.28586
97	dedicated: (7,7,7,7,7,7,7,7,7,7) const flx: (3,3,3,3,3,3,3,3,3,0)	8.37838
97	dedicated: (7,7,7,7,7,7,7,7,7,7) fully flx: 27	9.36535
87	dedicated: (6,6,6,6,6,6,6,6,6,6) const flx: (3,3,3,3,3,3,3,3,3,0)	7.47718
87	dedicated: (6,6,6,6,6,6,6,6,6,6) fully flx: 27	8.38774
96	dedicated: (6,6,6,6,6,6,6,6,6,6) const flx: (4,4,4,4,4,4,4,4,4,0)	8.33007
96	dedicated: (6,6,6,6,6,6,6,6,6,6) fully flx: 36	9.45340
86	dedicated: (5,5,5,5,5,5,5,5,5,5) const flx: (4,4,4,4,4,4,4,4,4,0)	7.39672
86	dedicated: (5,5,5,5,5,5,5,5,5,5) fully flx: 36	8.46073
95	dedicated: (5,5,5,5,5,5,5,5,5,5) const flx: (5,5,5,5,5,5,5,5,5,0)	8.14238
95	dedicated: (5,5,5,5,5,5,5,5,5,5) fully flx: 45	9.36825
85	dedicated: (4,4,4,4,4,4,4,4,4,4) const flx: (5,5,5,5,5,5,5,5,5,0)	7.17594
85	dedicated: (4,4,4,4,4,4,4,4,4,4) fully flx: 45	8.40664

Table 4. Throughput for configurations with $N = 10$ and multiple dedicated and flexible servers per station

Unlike full flexibility, the optimal policy under constrained flexibility is not rate independent for arbitrary configurations. The trade-off between the cost of making servers flexible (fully or constrained) and the throughput improvement can be used to decide if flexible servers should be constrained or not. In the future, it would be instructive to explore if structural results could be developed for systems where the service rates are heterogeneous and where the service distributions are not exponential. It would also be instructive to examine more general structures for how servers are constrained. For example, each flexible server could have a "zone" in which they could work – here, the zones are simply pairs of servers, but these zones could be more general in applications.

Acknowledgments. This work was supported by the Discovery Grant program of the Natural Sciences and Engineering Research Council of Canada.

References

1. R. Bekker, G. Koole, and D. Roubos. Flexible bed allocations for hospital wards. *Health Care Management Science*, 20:453–466, 2017.
2. A. de Bruin, R. Bekker, L. Zanten, and G. Koole. Dimensioning clinical wards using the Erlang loss model. *Annals of Operations Research*, 178:23–43, 2010.
3. L. Green. Capacity planning and management in hospitals. In M. Brandeau, F. Sainfort, and W. Pierskella, editors, *Operations Research and Health Care*, pages 15–41. 2005.
4. R. Hall. Bed assignment and bed management. In R. Hall, editor, *Handbook of Healthcare System Scheduling*, pages 177–200. 2012.
5. S. Hu, J. Ko, L. Weyand, H. ElMaraghy, T. Lien, Y. Koren, H. Bley, G. Chryssolouris, N. Nasr, and M. Shpitalni. Assembly system design and operations for product variety. *CIRP Annals - Manufacturing Technology*, 60:715–733, 2011.
6. S. A. Lippman. Applying a new device in the optimization of exponential queuing systems. *Operations Research*, 23(4):687–710, 1975.
7. M. van Vuuren. Performance analysis of tandem queues with small buffers. *IIE Transactions*, 41:882–892(11), October 2009.
8. M. H. Yarmand and D. G. Down. Server allocation for zero buffer tandem queues. *European Journal of Operational Research*, 230(3):596 – 603, 2013.
9. M. H. Yarmand and D. G. Down. Maximizing throughput in zero-buffer tandem lines with dedicated and flexible servers. *IIE Transactions*, 47(1):35–49, 2015.