Seeing Through Black Boxes : Tracking Transactions through Queues under Monitoring Resource Constraints[☆]

Animashree Anandkumar[a,*], Ting He[b], Chatschik Bisdikian[b], Dakshi Agrawal[b]

[a]*EECS Dept., University of California, Irvine, CA 92697, USA.*
[b]*Networking group, IBM Watson Research, Hawthorne, NY 10532, USA.*

Abstract

The problem of optimal allocation of monitoring resources for tracking transactions progressing through a distributed system, modeled as a queueing network, is considered. Two forms of monitoring information are considered, viz., locally unique transaction identifiers, and arrival and departure timestamps of transactions at each processing queue. The timestamps are assumed available at all the queues but in the absence of identifiers, only enable imprecise tracking since parallel processing can result in out-of-order departures. On the other hand, identifiers enable precise tracking but are not available without proper instrumentation. Given an instrumentation budget, only a subset of queues can be selected for production of identifiers, while the remaining queues have to resort to imprecise tracking using timestamps. The goal is then to optimally allocate the instrumentation budget to maximize the overall tracking accuracy. The challenge is that the optimal allocation strategy depends on accuracies of timestamp-based tracking at different queues, which has complex dependencies on the arrival and service processes, and the queueing discipline. We propose two simple heuristics for allocation by predicting the order of timestamp-based tracking accuracies of different queues. We derive sufficient conditions for these heuristics to achieve optimality through the notion of stochastic comparison of queues. Simulations show that our heuristics are close to optimality, even when the parameters deviate from these conditions.

Keywords: Probabilistic transaction monitoring, Queueing networks, Stochastic comparison, Bipartite matching

1. Introduction

Transaction processing has been at the heart of information technology since the 1950s when the first large online reservation system went into operation [2, 3]. Today transaction processing is at the core of enterprise IT systems operated by telecommunication service providers, financial institutions and virtual retailers. The scope of transaction processing has widened to incorporate multiple software components and applications, servers, middleware, backend databases, and multiple information sources [4].

The growing complexities of transaction processing presents new challenges to system management and support. Today's support helpdesks are no longer knowledgeable with the intimate details of transaction processing. The presence of heterogeneous components, legacy systems and third-party "black box" components [5] makes debugging, a slow and an expensive ordeal. It is thus highly desirable to speed up debugging through automated monitoring solutions.

Although tools may be available for independent trouble-shooting within each of the components, they cannot capture the entire life-cycle of a transaction, and thus cannot support diagnosis at the transaction level. Instead, an integrated end-to-end solution which tracks the entire path of transaction processing

[☆]The work is presented in part in [1].

[*]Corresponding author
Email addresses: a.anandkumar@uci.edu (Animashree Anandkumar), the@us.ibm.com (Ting He), bisdik@us.ibm.com (Chatschik Bisdikian), agrawal@us.ibm.com (Dakshi Agrawal)

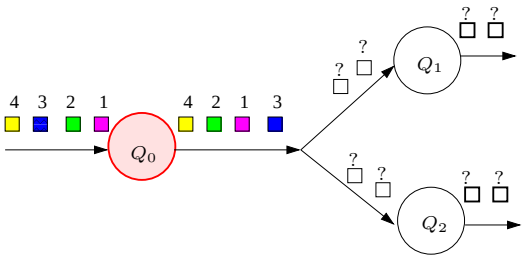


Figure 1: Introducing identifiers to timestamps at queue Q_0 through instrumentation precisely tracks transactions progressing through it. On the other hand, non-instrumented queues Q_1 and Q_2 have to track transactions using only arrival and departure timestamps may incur errors due to uncertainty in the order of departures.

is required [6]. An end-to-end monitor collects transaction records from different components and then correlates or matches them to obtain the complete transaction path. If all the components are instrumented properly, e.g., using techniques in [7, 8, 6], then each transaction record at every component is tagged with a unique *identifier* corresponding to the transaction generating it. Using these identifiers, correlation of transaction records at different components can then be done precisely.

In many practical scenarios, however, complete instrumentation of all the components is rarely the norm. This is due to the presence of legacy systems and third-party components with monitors producing incompatible transaction records, which in effect, is a set of "black boxes". In the extreme case when none of the components is instrumented, monitoring solutions have to fall back on other generic features in the records such as timestamps to statistically "guess" the set of records likely generated by the same transaction, and thereby infer the path taken by that transaction [5, 9], with the caveat that the results may be erroneous.

Most real systems lie somewhere in the middle of the spectrum between the extreme scenarios of fully instrumented and fully non-instrumented systems. In fact, most system integration and instrumentation is a gradual process which starts from an ensemble of black boxes and slowly transitions to a system of "clear" or "open" boxes as the support staff acquaint themselves with various components. Given sufficient time and efforts, skilled programmers are able to *retrofit* instrumentation[1] to components by injecting monitoring code or building an extra layer of middleware [6]. A complete instrumentation, however, can incur daunting costs and is nevertheless wasteful in components where statistical tracking using timestamps already has good accuracy. Our goal is then to systematically characterize the performance of partially instrumented monitoring systems and identify components where retrofitting instrumentation is most required.

We answer the following questions: given a limited budget for instrumentation, what is the optimal allocation strategy to maximize overall accuracy of tracking transactions? What is the influence of various system parameters, such as the queueing arrival and the service rates, on the instrumentation strategy and the tracking accuracy? Are there simple easy-to-implement heuristics that also have good performance guarantees? What follows is a set of systematic answers to these questions.

1.1. Technical Approach and Contributions

We consider the problem of tracking transactions through a distributed system with limited instrumentation support. Our goal is to select an optimal subset of components for instrumentation under a budget constraint such that when combined with statistical tracking (using timestamps) at the non-instrumented components, the overall tracking accuracy is maximized.

Our contributions are three fold. First, we analyze the accuracy of statistical tracking using timestamps at a queue and characterize its dependency on different queueing parameters. Second, using these insights, we propose two simple heuristics for the instrumentation allocation problem. Third, we derive sufficient

[1] Note that with partial instrumentation here the identifiers are local, defined only within each queue, which is different from the global identifiers in fully instrumented systems [8, 6].

conditions for these heuristics to achieve optimality, based on the arrival and the service distributions at the queues.

Model: We model the progress of the transactions in a distributed system as a queueing network, where each queue represents a system component. By default, we assume the availability of (an ordered) set of arrival and departure timestamps at each queue while identifiers are only available upon instrumentation (queue Q_0 in Fig.1). Due to parallel processing of transactions, e.g., in infinite server or processor-sharing queues[2], the order of departures is not unique, and in the absence of identifiers, tracking transactions through a queue requires statistical matching techniques. We analyze tracking accuracies using timestamps under two simple statistical matching policies. Identifiers are available only upon instrumentation and by instrumenting a queue, we mean injecting code or building a middleware wrapper which tags each timestamp with an identifier unique to the transaction, leading to error-free tracking at those queues.

Formulation: Based on the above model, we formulate a resource allocation problem, where we optimally allocate the available amount of monitoring resources by selecting queues for instrumentation such that the overall tracking accuracy is maximized. The optimal allocation strategy thus selects queues for instrumentation in the increasing order of their timestamp-based tracking accuracies, until the budget constraints are met. However, the exact expression of tracking accuracy at each non-instrumented queue is not tractable to compute in general, and has complex dependencies on the arrival and service statistics, and also on the queueing discipline.

Heuristic Solutions: To overcome this obstacle, we propose two simple heuristics for instrumentation allocation which predict the order of the timestamp-based tracking accuracies at different queues without computing the exact expressions. The first heuristic predicts that the order of tracking accuracies is in the reverse order of their queueing load factors. The second heuristic predicts the order of accuracies using an approximation for the tracking accuracy, which becomes tight in the light load regime. The two heuristics represent different tradeoffs in that the load-factor heuristic requires only the knowledge of the queueing load factors while the approximation-based heuristic requires the full knowledge of arrival and service processes but is a more efficient allocation strategy (demonstrated through both theory and simulations).

Optimality conditions: We provide sufficient conditions for these heuristics to achieve optimality, i.e., to correctly rank the order of the tracking accuracies, based on the notions of *stochastic* and *convex orders* of the arrival and service distributions of the queues. The conditions have intuitive explanations in terms of the rate and the "variability" of arrivals and services. In particular, these heuristics are always optimal when all the arrival distributions and all the service distributions belong to the same family. Simulations verify the optimality of our heuristics under the derived conditions and also show that our heuristics are close to optimality even when the parameters deviate from these conditions.

Alternative Formulation: Besides allocating instrumentation resources, our heuristics are also applicable in other scenarios of monitoring. For instance, for a large system, the overhead in collecting timestamp records from all the components may be too large. In this case, the optimal monitoring resource allocation is to *a priori* select only a subset of components (queues) with the highest timestamp-based tracking accuracies for data collection. Our heuristics and their optimality guarantees are directly applicable here.

Non-goals: We emphasize some of our "non-goals". Our formulation and solutions have a strong theoretical foundation and are meant to provide guidelines for efficient instrumentation or data collection in different scenarios. We do not attempt to replace existing instrumentation-based monitoring tools (See Section 1.2 for a discussion) and exploit them when available. Our belief is that existing monitoring solutions will have broader application by allowing for partial instrumentation, and we have a systematic approach for pursuing it. Moreover, our solutions are not meant to automatically diagnose or correct faults, characterize overall system performance, or provide real-time analysis, although such exercises can be carried out after monitoring the transaction paths.

[2]There is no uncertainty in the order of departures for single-server queues with fixed order processing. Hence, their timestamp-based tracking is error-free, and we do not consider them for allocation.

1.2. Related Work

The early literature on monitoring distributed systems relies on deep understanding of internal system structures so that instrumentation code can be injected into proper places to record system activities at process or object levels [10, 11, 12]. These solutions become difficult to implement in modern systems where components are typically developed independently. Most existing monitoring solutions rely on certain types of instrumentation that can expose the activities of interest [13, 14, 15, 6]. There are also a number of commercially-available products for monitoring and trouble shooting in distributed systems [7, 5, 16], which are again based on instrumenting the system software.

While instrumentation provides reliable monitoring information, it has limited use in heterogeneous systems where many components are from third-party vendors or legacy systems. One approach is to make the instrumentation as component-independent as possible, e.g., by limiting changes to system code rather than user-space code [17]. Another approach is to treat each component as a black box and only rely on external activities of these black boxes for monitoring [5, 16, 8]. These existing black-box based solutions can be divided into two approaches: identifier-based approach [8] which tags each incoming transaction with a unique identifier that is associated with it throughout the system, converting the problem to the instrumented case, and trace-based approach [5, 16] which uses statistical techniques to extract monitoring information from non-tagged activities. For example, [5, 16] use messages between components to infer causal paths and bottlenecks. We share a similar view as [5, 16] in that a monitoring solution should be as non-intrusive and agnostic as possible to allow for broad application, especially in systems involving black boxes, but there are two key differences that distinguish our work from this literature: (i) we are interested in monitoring individual transactions rather than aggregate system behaviors such as causal paths and bottlenecks, and (ii) we take a hybrid approach of using both passive monitoring (via timestamp-based tracking) and instrumentation (that introduces identifiers), but treat the latter as a limited resource to be allocated judiciously.

In [9], tracking of individual transactions in a distributed system based solely on timestamps is considered. However, [9] focuses on developing optimal matching policies for timestamp-based transaction monitoring, whereas we focus on the comparison of tracking accuracies at different subsystems while leveraging statistical matching policies discussed in [9] for tracking in the non-instrumented states. The stochastic comparison techniques used in this paper has a rich history and has been applied compare different queueing parameters such as delay and throughput [18, Ch. 14]. To the best of our knowledge, comparison of monitoring accuracies at different queues has not been considered before.

Organization: The paper is organized as follows. In Section 2, we describe the system model and problem formulation. In Section 3, we analyze the policies for matching timestamps. In Section 4, we propose the two heuristics for monitoring resource allocation. In Section 5, we introduce the notion of stochastic comparison. In Section 6, we derive sufficient conditions for the optimality of the two heuristics for network of infinite-server queues. Section 7 deals with extensions to general product-form queues. In Section 8, we evaluate the efficiency of heuristics through simulations. Section 9 concludes our paper.

2. System Model and Formulation

We now describe the queueing model in detail and then formulate the problem of optimal monitoring resource allocation. Before we proceed, here are a few comments regarding the notation used in this paper. Vectors are represented by boldface, e.g., $\mathbf{X}$ and $X(i)$ is its i^{th} element. Let $f_X(x)$, $F_X(x)$ and $\bar{F}_X(x)$ denote the probability density function (pdf), cumulative distribution function (cdf) and complementary cumulative distribution function (ccdf) of a continuous variable X. Let $\mathbb{E}[X]$ denote its expectation and let $\text{supp}(f_X)$ denote the support of f_X.

2.1. System Model

We consider a queueing network, and initially limit to the case where all the queues are infinite server ($GI/GI/\infty$). The arrival and service times are drawn i.i.d. from general continuous pdfs f_X and f_T. In Section 7, we generalize some of our results to the product-form queues. We assume that the sequence

$\mathbf{X}_k$	vector of i.i.d inter-arrival times
$\mathbf{T}_k$	vector of i.i.d service times
$\lambda_k := \frac{1}{\mathbb{E}[X_k(1)]}$	arrival rate
$\mu_k := \frac{1}{\mathbb{E}[T_k(1)]}$	service rate
$\rho_k := \frac{\lambda_k}{\mu_k}$	load factor
V_k	$T_k(1) - T_k(2)$: spread of service time
$\mathbf{Y}_k$	vector of arrival times
$\mathbf{D}_k$	vector of departure times
$\boldsymbol{\pi}_k^t$	true matching btw. arrivals & departures
$\boldsymbol{\pi}_k^\gamma$	matching according to policy γ
B_k	(random) no. of arrivals in a busy period
$P^\gamma(k)$	prob. of correct matching in a busy period
$P_b^\gamma(k)$	cond. prob. of correct match given $B_k = b$

Table 1: Symbol list. Subscript k means queue Q_k.

of queues visited by each transaction is a Markov chain, and the service is independent of the transition sequence. The list of notations for different queueing parameters is given in Table 1. The propagation delays and synchronization errors between different queues are assumed independent of the service or arrival realizations.

Given a set of ordered arrival and departure timestamps, $\mathbf{Y}_k$ and $\mathbf{D}_k$ at queue Q_k, there is a relationship between the service times and the true matching $\boldsymbol{\pi}_k^t$ between the arrivals and the departures, as

$$T_k(i) = D_k(\pi_k^t(i)) - Y_k(i), \quad i \in \mathbb{N}. \tag{1}$$

Hence, $\pi_k^t(i)$ is the rank of a departure timestamp corresponding to the i^{th} arrival to the queue Q_k. Since we have access to only the arrival and departure timestamps $\mathbf{Y}_k$ and $\mathbf{D}_k$, and *not* to the actual service times $\mathbf{T}_k$, the true matching $\boldsymbol{\pi}_k^t$ is unknown. A *bipartite matching policy* γ comes up with a probable matching $\boldsymbol{\pi}^\gamma$ between the arrival timestamps $\mathbf{Y}_k$ and the departure timestamps $\mathbf{D}_k$, which yields correct matchings with a certain degree of accuracy, and is discussed in detail in Section 3. In addition, we assume that identical policies γ are employed for matching at all the queues to facilitate comparison of their tracking accuracies.

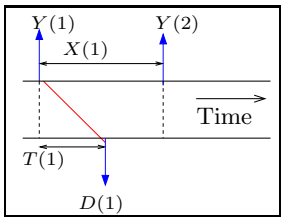
Our analysis will be on a typical busy period, i.e., a period of time, starting from an empty queue until the next time the queue becomes free, as shown in Fig.2. Let $P^\gamma(k)$ be the probability that the policy outputs a correct matchings between *all* the arrivals and departures in a typical busy period at queue Q_k. We use $P^\gamma(k)$ as the measure of timestamp-based tracking accuracy, given by

$$P^\gamma(k) = \sum_{b=1}^{\infty} \mathbb{P}[\boldsymbol{\pi}^\gamma = \boldsymbol{\pi}^t, B_k = b]. \tag{2}$$

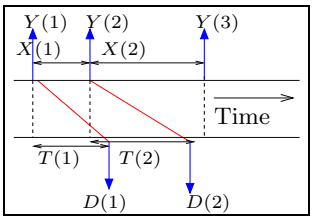
2.2. Problem Formulation

We are now ready to state the problem of optimal monitoring resource allocation. Given a budget constraint of instrumenting at most E number of queues to enable precise tracking through the production of identifiers, our goal is to select E number of queues in $\mathcal{Q}$ such that the overall tracking accuracy is maximized. For each queue Q_k, let $z_k \in \{0, 1\}$ be the indicator if it is selected for instrumentation. Then, the effective tracking accuracy at queue Q_k after instrumentation decisions is

$$z_k + (1 - z_k)P^\gamma(k),$$



(a) Busy Period $B = 1$



(b) Busy Period $B = 2$

Figure 2: Random arrivals and departures lead to random busy period sizes.

since the tracking accuracy is unity when identifiers are available and $P^{\gamma}(k)$ is the accuracy based on using only timestamps under a statistical matching policy γ. Formally, the optimization is

$$\mathbf{z}_*(E;\mathcal{Q}) := \arg\max_{\mathbf{z}} \sum_{Q_k \in \mathcal{Q}} \{z_k + (1-z_k)P^{\gamma}(k)\}, \tag{3}$$

$$s.t. \quad \sum_{Q_k \in \mathcal{Q}} z_k \leq E, \quad z_k \in \{0,1\}, \mathbf{z} := \{z_k : Q_k \in \mathcal{Q}\}.$$

We can see that the optimal allocation strategy is to select E number of queues with the lowest timestamp-based tracking accuracies P^{γ}. The challenge, as we will see, is in finding the tracking accuracy P^{γ} since it has complex dependencies on the arrival and service processes.

3. Timestamp-based Tracking

In this section, we describe the matching policies γ employed for associating the arrival and the departure timestamps at a queue, and perform some preliminary analysis on the tracking accuracy of a policy.

3.1. Bipartite Matching Policies

We now briefly describe two matching policies γ that can be employed to match timestamps in the absence of identifiers, viz., the first-in first-out (FIFO) rule and the random matching rule. The relative performance of these policies depends on the arrival and service statistics. These policies are *non-parametric*, in the sense that they require minimal knowledge about the service statistics for implementation.

Perhaps the simplest matching rule between the arrival and departure timestamps is the FIFO rule, which is an in-order matching rule, i.e., for a given busy-period size $B = b$, we have a fixed rule $\boldsymbol{\pi}^{\text{FIFO}} = \mathbf{I}$, where $\mathbf{I} := [1, 2, \ldots]^T$ is the identity vector. The FIFO matching rule is fully distribution-free: it does not require the knowledge of arrival or service distribution and is always valid. By valid, we mean that the FIFO match has a strict positive likelihood of being the true match between the arrivals and the departures. An expression for the expected matching accuracy under FIFO rule can be found in Appendix A.

In addition to the FIFO matching rule, we consider another simple rule called random matching, where given a realization of arrivals and departures in a busy period, we uniformly pick a valid matching among all possible matchings. The random matching rule is almost distribution-free: it only requires the knowledge of $\text{supp}(f_{T_k})$, the support of the service pdf, in order to ensure the validity of different matchings. This is because a valid matching $\boldsymbol{\pi}$ at queue Q_k in a busy period of size $B_k = b$ satisfies

$$\boldsymbol{\pi} : \prod_{i=1}^{b} f_{T_k}[D_k(\pi(i)) - Y_k(i)] > 0, \tag{4}$$

and the above expression only requires the knowledge of the support bounds. An expression for tracking accuracy P^{RAND} under random matching is given in Appendix B.

In contrast to the non-parametric FIFO and random matching rules, the parametric *maximum-likelihood* matching rule [9] requires the full knowledge of the service distribution. The maximum-likelihood rule is defined as the rule which maximizes the probability of correctly matching all the arrivals and departures. However, it is not tractable to analyze this rule since it is fully adaptive to the realization of arrivals and departures, and depends on the arrival and service statistics in a complex manner. In many cases, the simple FIFO and random matching policies coincide with the maximum-likelihood rule or have close to optimal performance, as discussed below.

The effectiveness of using the FIFO or the random policy crucially depends on the nature the service distribution (for a given realization of arrivals). For instance, under light-tailed services, the probability of out-of-order departures is small and hence, the FIFO rule is expected to have good tracking accuracy. In fact, for Weibull[3] family of distributions, with shape parameter greater than one (and hence, light tailed), FIFO is the optimal matching policy coinciding with the maximum-likelihood rule. More generally, FIFO rule is optimal whenever the service pdf is log-concave [9].

For heavy-tailed distributions, on the other hand, the chances of out-of-order departures are high, and the FIFO rule is not close to the maximum-likelihood rule. In this case, the random matching rule may have better tracking accuracy than the FIFO rule. This is observed in our simulations in Fig.5b for Weibull distribution with shape parameter smaller than one. Moreover, random matching is optimal in case of batch arrivals to the infinite-server queue where all possible matchings between the arrivals and departures are equally likely, although we do not study this scenario in the paper. Hence, the relative performance of FIFO and random matching rule depends on the service distribution.

3.2. Tracking Accuracy

Recall we consider the probability of matching all timestamps in a typical busy period to be the measure of tracking accuracy. Perhaps, a more straightforward measure of accuracy is the probability of correctly matching only a typical pair of arrival and departure timestamps. This however depends on the probability of correctly matching other arrivals and departures. On the other hand, the matching across busy periods is independent, since a valid matching between arrival and departure timestamps occurs only within busy periods not across them. See Fig.3. Hence, the probability of correct matching in a typical busy period P^{γ} is the relevant measure for tracking accuracy.

The challenge is in computing P^{γ} in (2). Consider FIFO matching as an example. Its accuracy is equal to (see Appendix A)

$$P^{\text{FIFO}} = \sum_{b=1}^{\infty} \mathbb{P}\Big(\bigcap_{i=1}^{b-1} \{T(i) \in [X(i), X(i)+T(i+1)]\} \cap \{T(b) < X(b)\}\Big),$$

where the events $T(i) \in [X(i), X(i)+T(i+1)]$ and $T(b) < X(b)$ cannot be evaluated separately since are correlated with one another other. We can see that the expression becomes intractable as we increase b, the size of the busy period.

More generally, a matching policy γ may select any one of the valid matchings or permutations with a certain probability, and the tracking accuracy P^{γ} from (2) becomes

$$P^{\gamma} = \sum_{b=1}^{\infty} \sum_{\boldsymbol{\pi}_j} \mathbb{P}[\boldsymbol{\pi}^{\gamma} = \boldsymbol{\pi}^{t} = \boldsymbol{\pi}_j, B = b],$$

where the sum is over all the permutation vectors $\boldsymbol{\pi}_j$ over $\{1, 2, \ldots, b\}$. Since there are $b!$ number of permutation vectors, we require exponential number of computations in b.

It is therefore not tractable to compute the tracking accuracies $P^{\gamma}(k)$ at different queues Q_k, in order to find the optimal resource allocation strategy in (3). Moreover, it is useful to obtain some general guidelines

[3] The pdf of a Weibull variable is $f(x) = (\frac{w}{c})(\frac{x}{c})^{w-1} \exp(-(\frac{x}{c})^{w})$ for $x > 0$, where w and c are shape and scale parameters. When $w > 1$, the distribution is light tailed, when $w < 1$, it is heavy tailed and $w = 1$ is the exponential distribution.

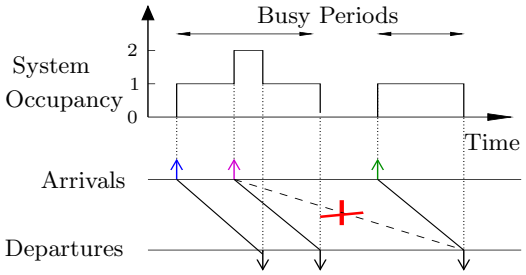


Figure 3: Matching arrival and departure timestamps decomposes across different busy periods.

about the influence of different queueing parameters on the resulting tracking accuracy. Fortunately, we note that we do not need to know the exact accuracies at different queues in the network to obtain the optimal solution to instrumentation allocation in (3). In fact, it suffices to know the relative order of these accuracies. The goal of this paper is to establish simple heuristics that can be used to infer the order of matching accuracies without directly computing them. To this end, we now propose two approaches with different complexities and generality. Later in Section 5 and 6, we derive sufficient conditions for these heuristics to achieve optimality according to (3).

4. Two Heuristics for Optimal Resource Allocation

We propose two approaches to instrumentation allocation through prediction of the order of timestamp-based tracking accuracies $P^\gamma(k)$ at different queues Q_k. One approach is to avoid computation of $P^\gamma(k)$ altogether and instead infer their order through simple queueing parameters such as the load factors. The other approach is to approximately compute $P^\gamma(k)$ by only considering small busy-period sizes. Both these simple approaches instrument queues independent of the policies γ employed for timestamp matching. We now describe these two approaches in detail.

4.1. Approach 1: Order of Load Factors

The load factor $\rho_k = \frac{\lambda_k}{\mu_k}$ of a queue Q_k, which is the ratio of the arrival rate λ_k to the service rate μ_k, is perhaps the most commonly used queueing parameter for performance evaluation of queues. We propose the load-factor heuristic for instrumentation allocation which selects queues for instrumentation in the decreasing order of their load factors until the budget constraint is met. The load-factor heuristic is *robust* since the selected set of queues is invariant under small perturbations in the arrival and service statistics.

The load-factor heuristic predicts queues with higher load factors to have lower timestamp-based tracking accuracies. This is intuitive since a lighter load implies a smaller number of simultaneously-served arrivals in the infinite-server queue on average leading to a lower uncertainty in the order of departures. The intuition, however, does not extend when we consider queues with different arrival and service distributions. The arrival and service processes influence the tracking accuracy in a complex manner, and the load factor may not always capture the required effects for comparison of tracking accuracies at different queues.

A simple example is two queues with same arrival rate, one with uniform service $\text{Unif}(0, 2m)$ on support $[0, 2m]$ and the other with deterministic service of value $m_d > m$. Here, the load-factor heuristic incorrectly predicts the deterministic service to have worse tracking accuracy, while, in fact, it actually has perfect accuracy. Hence, the load-factor heuristic is not universally optimal for instrumentation allocation.

An intuitive reason for the sub-optimality of the load-factor heuristic is that there are two sources of errors impacting the tracking accuracy: variability in service times leading to uncertainty in the order of departures and high load factor resulting in more simultaneous servicing in infinite-server queues on average. The load-factor heuristic only captures the latter effect and completely ignores the former. As we saw in

the above example, simultaneous servicing does not always lead to bad accuracy and is also governed by the variability in the service times.

In many cases, different subsystems in a distributed system may have similar service distributions (such as from the same family), but with different load factors. Here, the load-factor heuristic may correctly predict the order of the tracking accuracies. We prove a sufficient set of conditions for the optimality of the load-factor heuristic in Section 6 by precisely investigating the dependency of the arrival and the service processes on the tracking accuracy.

4.2. Approach 2: Small-Batch Approximation

The load-factor heuristic described in the previous section avoids computation of the tracking accuracy altogether. We now propose an alternative heuristic which approximates tracking accuracy through a simple expression, and makes instrumentation decisions based on the approximation. We later demonstrate the superiority of this heuristic over the load-factor heuristic, both through theory and simulations.

The approximation for tracking accuracy is based on the series expansion

$$P^{\gamma}(k) = \mathbb{P}[B_k = 1] + \sum_{b=2}^{\infty} P_b^{\gamma}(k)\mathbb{P}[B_k = b], \tag{5}$$

where $P_1^{\gamma} = 1$ since when there is only one transaction in the busy period, tracking is perfect. Under sufficient variability of the service times (i.e., not deterministic services), the probability of correct matching typically decays with the busy-period size,

$$\lim_{b\to\infty} P_b^{\gamma}(k) = 0,$$

since the number of possible matchings grows exponentially with the busy-period size b and we make an error almost surely as the busy period size goes to infinity. Hence, the terms corresponding to larger busy-period sizes in (5) can be dropped and an approximate tracking accuracy can be efficiently computed by limiting to small busy-period sizes.

The simplest approximation is when we ignore all the terms in (5) except for the first one, which is simple to evaluate. We refer to this as the *unit-batch* approximation and use it to allocate instrumentation resources to queues. Note that the unit-batch approximation is slightly more complex than the load-factor heuristic. We demonstrate, both through theory and simulations, that this leads to superior performance over the load-factor heuristic; the intuition being that this heuristic captures additional features of the arrival and service statistics.

At low arrival rate, this approximation (and also more refined ones with more terms) becomes tight in the limit. Intuitively, at low arrival rates, the dominant event is having a single arrival in each busy period since the arrivals are widely separated on average.

Proposition 1. (Tightness at Low Arrival Rate). *As the arrival rate to a queue Q_k goes to zero, and the service distribution is kept fixed, we have*

$$\lim_{\lambda_k\to 0} \frac{\mathbb{P}[B_k = 1]}{P^{\gamma}(k)} = 1. \tag{6}$$

Proof: As $\lambda_k \to 0$, we have $\mathbb{P}[B_k = 1] = \mathbb{P}[X_k > T_k] \to 1$ and $P^{\gamma} \to 1$ since the probability of out-of-order departures goes to zero. □

Hence, the tracking accuracy P^{γ} is well approximated by the probability of unit busy period in the low arrival rate or the light load regime. However, simulations in Section 8 show that the unit-batch approximation correctly captures the trend of P^{γ} and is hence, an efficient strategy for instrumentation allocation over a wider regime of loads.

5. Preliminaries: Stochastic Comparison

We have so far proposed two simple heuristics for optimal instrumentation resource allocation which circumvent the challenges in computing the tracking accuracies at various queues. Our goal is to establish a general set of conditions on the arrival and service processes, under which these simple heuristics coincide with the optimal allocation strategy. To this end, we introduce the notion of stochastic comparison of random variables.

Perhaps the simplest notion of comparing two random variables is through their mean values. But very often, this comparison turns out to be too loose to draw useful conclusions since the probability distribution of the two variables can be very different. In the context of this paper, comparing only queueing load factors, which is just the average system behavior, is not enough to always guarantee an order of the tracking accuracies of the queues and hence, optimality of the load-factor heuristic for instrumentation allocation.

Instead, we impose stronger constraints on the distributions of the variables under comparison to obtain useful conclusions. Here, we employ two notions of stochastic comparison, viz., the stochastic order and the convex order. The stochastic order is a stronger form of comparing the mean values, while the convex order is a stronger form of comparing the variances of random variables. The detailed definitions are given in Appendix C. We use these notions in Section 6 to compare tracking accuracies at different queues, and to derive sufficient conditions for the optimality of the two proposed heuristics for resource allocation.

5.1. Stochastic Comparison of Busy Periods

We now provide some preliminary results on comparing the busy-period sizes of queues under stochastic or convex orders of arrival and service processes. We use these results in Section 6 to obtain an order on the tracking accuracies of the queues thereby establishing the optimality of our heuristics for instrumentation allocation.

We now show that under a stochastic order of arrival processes and service processes at two queues, we can guarantee a stochastic order of the size of their busy periods.

Lemma 1. (Comparison of Busy Periods under Stochastic Order). *For two $GI/GI/\infty$ queues Q_k, Q_m with i.i.d arrivals X_k, X_m and i.i.d service times T_k, T_m, we have*

$$X_k \overset{st}{\leq} X_m, T_k \overset{st}{\geq} T_m \Rightarrow B_k \overset{st}{\geq} B_m. \tag{7}$$

Proof: See Appendix D □

The above result confirms our intuition that the size of the busy period increases with faster arrivals and slower services (and hence, higher load factors), formalized under the notion of stochastic order.

We now consider an alternative scenario where one queue has a higher (normalized) service variability than the other, formalized by the presence of a convex order. We show that this also implies a stochastic order on their busy-period sizes for the special case of Poisson arrivals at all the queues.

Lemma 2. (Comparison of Busy Periods under Convex Order & Poisson Arrivals). *For two $M/GI/\infty$ queues Q_k, Q_m with i.i.d Poisson arrivals with rates λ_k, λ_m and i.i.d service times T_k, T_m, we have*

$$\lambda_k T_k \underset{cx}{\leq} \lambda_m T_m \Rightarrow B_k \overset{st}{\leq} B_m. \tag{8}$$

Proof: See Appendix E. □

Informally, the above result states that a more variable service distribution (normalized by the arrival rate) results in larger busy periods.

The results in (7) and (8) form an integral component of our proofs in the comparison of tracking accuracies since, larger busy periods leads to lower tracking accuracies. However, we see in the subsequent sections that certain additional conditions, in addition to stochastic or convex orders of arrivals and services, are needed to guarantee the order of the tracking accuracies, and hence, optimality of our heuristics for instrumentation allocation.

6. Optimality in $GI/GI/\infty$ Queues

6.1. Load-Factor Heuristic

Recall that the load-factor heuristic, described in Section 4.1, predicts queues with higher load factors to have lower timestamp-based tracking accuracy and hence, selects them for introducing identifiers through instrumentation. We now provide sufficient conditions on the arrival and service processes under which the load-factor heuristic is the optimal resource allocation strategy.

A stochastic order on the arrival and the service times is a prerequisite condition in our approach since it leads to a stochastic order on the busy periods from Lemma 1. In addition to the stochastic order on the arrival and service processes, we need additional conditions to establish the order of the tracking accuracies, depending on the matching policy employed. These additional conditions turn out to be different for the FIFO and the random matching rule. This is because the tracking accuracies of the two rules are sensitive to different kind of events. For the FIFO rule, any out-of-order departure results in an error, which implies its sensitivity to the *spread* of the service distribution, defined precisely in Section 6.1.1. On the other hand, random matching is somewhat less sensitive to the service spread since it uniformly picks a matching out of all valid matchings, and this is reflected in our results. We first provide sufficient conditions for optimality of the load-factor heuristic under the FIFO rule and then consider the random matching rule. Finally, in Section 6.1.3, we provide examples where these conditions are satisfied.

6.1.1. Optimality Under FIFO Matching Rule

We now provide conditions for the optimality of the load-factor heuristic when FIFO is the matching policy employed at all the queues. Since overtaking or out-of-order departures cause errors in FIFO matching, we relate the tendency for overtaking to the *spread* of the service distribution, given by

$$V_k := T_k(1) - T_k(2), \tag{9}$$

where $T_k(1)$ and $T_k(2)$ are independent samples of the service time T_k at queue Q_k. Note that $V_k \equiv 0$, if the service is deterministic. The spread of a distribution is thus related to the variability; a more "spread out" service distribution has higher variability, and thus, has higher tendency for generating out-of-order departures.

We now show the main result that the order of the tracking accuracies under FIFO rule follow the reverse order of the load factors in the presence of a stochastic order.

Theorem 1. (Optimality of Load-Factor Heuristic Under FIFO Rule). *At queues Q_k, Q_m, under a stochastic order on arrival times X_k and X_m, service times T_k and T_m and their spreads V_k and V_m, we have*

$$\begin{aligned} &X_k \overset{st}{\leq} X_m,\, T_k \overset{st}{\geq} T_m,\, |V_k| \overset{st}{\geq} |V_m| \\ &\Rightarrow \rho_k \geq \rho_m,\, P^{FIFO}(k) \leq P^{FIFO}(m). \end{aligned} \tag{10}$$

Hence, if the arrival, service and service spread distributions at all the queues satisfy the above stochastic order, then the load-factor heuristic for allocation of instrumentation resources is optimal, according to optimization in (3).

Proof: See Appendix F. □

Hence, slower arrivals, faster services (which thus imply a lower load factor), and lower service spreads result in more accurate tracking under the FIFO rule, when the comparison is formalized by the notion of stochastic order.

The combined conditions of service speed and spread in (10) places constraints on the service distributions under comparison. Informally, we need one service to be simultaneously slower and more spread out than the other, i.e., one service distribution has more probability mass concentrated closer to zero than the other. For example, the Weibull distribution with different shape parameters but same scale parameter satisfies this condition, as shown in Fig.4.

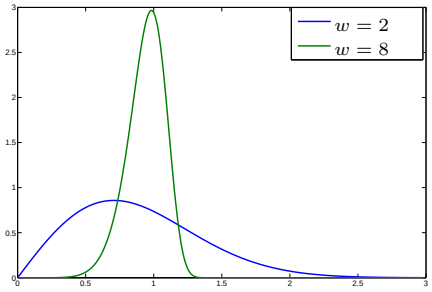


Figure 4: Comparison of two Weibull distributions with pdf $f(x) = (\frac{w}{c})(\frac{x}{c})^{w-1}\exp(-(\frac{x}{c})^w)$ for $x > 0$. The distribution with lower shape parameter w has higher FIFO tracking accuracy. See Theorem 1.

6.1.2. Optimality Under Random Matching

We now provide sufficient conditions for optimality of the load-factor heuristic when the random matching rule is employed for matching arrival and departure timestamps at all the queues. Recall that random matching rule uniformly chooses a matching among all valid matchings in the busy period.

We now show the main result of this section that the order of the tracking accuracies under the random matching rule follow the reverse order of the load factors in the presence of a stochastic order.

Theorem 2. (Optimality of Load-Factor Heuristic Under Random Matching Rule). *At queues Q_k, Q_m, under random matching rule with arrival times X_k and X_m, service times T_k and T_m with supports,* $\text{supp}(f_{T_k}) = [\alpha_k, \beta_k]$ *and* $\text{supp}(f_{T_m}) = [\alpha_m, \beta_m]$*, we have*

$$\begin{aligned} &X_k \overset{st}{\leq} X_m,\ T_k \overset{st}{\geq} T_m,\ \alpha_k \leq \alpha_m \\ &\Rightarrow \rho_k \geq \rho_m,\ P^{RAND}(k) \leq P^{RAND}(m). \end{aligned} \tag{11}$$

Hence, if the arrival, service and service support at all the queues satisfy the above stochastic order, then the load-factor heuristic for allocation of instrumentation resources is optimal, according to optimization in (3).

Proof: See Appendix G. □

Hence, slower arrivals and faster services along with a mild condition on the support lower bounds of the service distribution imply lower tracking accuracy under the random matching rule, when the comparison is formalized by a stochastic order.

The condition in (11) on the support of the service distributions is mild and is usually satisfied since one mostly encounters service distributions with a lower bound of support equal to zero. However, it cannot be dropped as seen in this example when $T_k \equiv \mu_k$ and $T_m = \text{Unif}(0, 2\mu_m)$, the uniform distribution, with $\mu_k > \mu_m$. Since $\alpha_k = \mu_k > \alpha_m = 0$, (11) does not hold, which is indeed true since in fact, $P^{\text{RAND}}(k) = 1 > P^{\text{RAND}}(m)$ in this example.

6.1.3. Special Case: Same Distribution Family

We have so far established sufficient conditions for optimality of the load-factor heuristic when all the queues employ either the FIFO or the random matching rules. We now consider a special case of arrival and service distributions belonging to the same distribution family where optimality of the load-factor heuristic is guaranteed under both FIFO or random matching rules, without the need for additional conditions.

Corollary 1. (Optimality of Load-factor Heuristic Under Same Distribution Family). *When the service distributions at different queues are linearly scaled versions of the same distribution, and the same holds for all the arrival distributions as well, then the tracking accuracies at the queues are in the reverse order of their load factors under both FIFO and random matching rules. Hence, here, the load factor heuristic is optimal for resource allocation, according to optimization in* (3).

Proof: We show that the conditions for FIFO rule in Theorem 1 are satisfied. For random matching rule, the condition on lower bound of support in Theorem 2 is, however, violated. Hence, we need to prove the above statement from scratch. See Appendix H. □

The above result holds if all the service distributions are say exponential, uniform and so on. In practice, the service distributions of different subsystems may be similar and hence, this result may be relevant. The constraint on the arrival processes is however more restrictive in case of an inter-connected network of queues, since it limits to Poisson arrivals to the system.

6.2. Unit-Batch Approximation

We have so far demonstrated the effectiveness of the load-factor heuristic when the arrival and service distributions are similar or more generally, constrained to satisfy a stochastic order. Next, we provide sufficient conditions to establish the optimality of the alternative heuristic for instrumentation allocation based on unit-batch approximations, described in Section 4.2. Recall that the unit-batch approximation selects queues for instrumentation in the increasing order of their probability of having a unit-sized busy period.

6.2.1. Optimality Under Stochastic Order

We now show that the conditions given in Theorems 1 and 2, which guarantee optimality of the load-factor heuristic, also guarantee the optimality of the unit-batch approximation.

Theorem 3. (Optimality of Unit-Batch Approximation Under Stochastic Order). *We have for two queues Q_k and Q_m,*

$$\begin{aligned} &X_k \overset{st}{\leq} X_m,\, T_k \overset{st}{\geq} T_m,\, |V_k| \overset{st}{\geq} |V_m| \\ &\Rightarrow \mathbb{P}[B_k = 1] \leq \mathbb{P}[B_m = 1],\, P^{FIFO}(k) \leq P^{FIFO}(m). \end{aligned} \tag{12}$$

$$\begin{aligned} &X_k \overset{st}{\leq} X_m,\, T_k \overset{st}{\geq} T_m,\, \alpha_k \leq \alpha_m \\ &\Rightarrow \mathbb{P}[B_k = 1] \leq \mathbb{P}[B_m = 1],\, P^{RAND}(k) \leq P^{RAND}(m). \end{aligned} \tag{13}$$

Hence, the above conditions guarantee that the heuristic based on unit-batch approximation coincides with the load-factor heuristic and hence, also achieves optimality in (3).

Proof: It is easy to see that $\mathbb{P}[B_k = 1] = \mathbb{P}[X_k > T_k] \leq \mathbb{P}[B_m = 1]$ since $X_k - T_k \overset{st}{\leq} X_m - T_m$. □

Hence, the unit-batch approximation achieves optimality in the above scenario where the load-factor heuristic is also optimal. We now demonstrate the superiority of the unit-batch approximation over the load-factor heuristic by considering a different scenario.

6.2.2. Optimality Under Convex Order

We now consider a special scenario where all the queues have the same load factor but with different service variabilities. In this case, the load-factor heuristic fails to distinguish the tracking accuracies of different queues and its performance is equivalent to a random selection of queues for instrumentation. On the other hand, we show below that the unit batch approximation achieves optimality when the queueing services satisfy a convex order.

Theorem 4. (Optimality of Unit-Batch Approx. Under Convex Order and FIFO Rule). *For two $M/GI/\infty$ queues Q_k, Q_m with i.i.d Poisson arrivals with rates λ_k, λ_m and i.i.d service times T_k, T_m, we have*

$$\begin{aligned} &\lambda_k T_k \underset{cx}{\leq} \lambda_m T_m \\ &\Rightarrow \mathbb{P}[B_k = 1] \geq \mathbb{P}[B_m = 1],\, P^{FIFO}(k) \geq P^{FIFO}(m). \end{aligned} \tag{14}$$

Hence, under Poisson arrivals, convex order of normalized services and FIFO matching rule, the unit-batch approximation is the optimal strategy for allocation of instrumentation resources, according to optimization in (3).

Proof: $\mathbb{P}[B_k = 1] = \mathbb{E}[e^{-\lambda_k T_k}]$ is a concave function in $\lambda_k T_k$ and hence, $\mathbb{P}[B_k = 1] \geq \mathbb{P}[B_m = 1]$. For the order of $P^{\text{FIFO}}(k)$ and $P^{\text{FIFO}}(m)$, see Appendix I. □

Hence, the unit-batch approximation achieves optimality over a wider range of distributions than the load factor heuristic. The relative performance of the load-factor heuristic and unit-batch approximation for instrumentation allocation depends on the queues under consideration. For queues with similar service distributions but significantly different load factors, the load-factor heuristic suffices to achieve efficient allocation. On the other hand, if all the load factors are close to one another, the effect of service variability and higher-order moments become significant and are not captured by the load-factor heuristic. In such scenarios, there is significant advantage in employing the unit-batch approximation.

7. Product-Form Networks

We have so far considered comparison of monitoring performance for different service distributions when all the queues are infinite-server queues. In this section, we extend some of our results to the more general queueing networks consisting of egalitarian processor sharing (PS) queues (with load factors less than one to ensure stability) and the infinite-server queues. These are part of the well-known product-form queues[4] [21].

7.1. Processor-Sharing Network

We first consider all the queues to be processor-sharing queues which makes comparison between them tractable. In the (egalitarian) processor sharing, each waiting transaction gets an equal share of service capacity. Since there is simultaneous processing of transactions, out-of-order departures are possible and there is uncertainty in matching arrival and departure timestamps.

In a nutshell, we now show that the comparison results for infinite-server queues under random matching in Theorem 2 holds for processor-sharing queues as well. However, the proof is more involved since the sojourn time distributions of different transactions are correlated under processor-sharing discipline.

We use the term *job-length* to refer to the amount of service required, and we use the term *sojourn time* to denote the amount of time spent in the system. We denote the job-lengths by $\mathbf{J} = [J(1), J(2), \ldots]$ and assume that $J(i) \overset{i.i.d.}{\sim} f_J$.

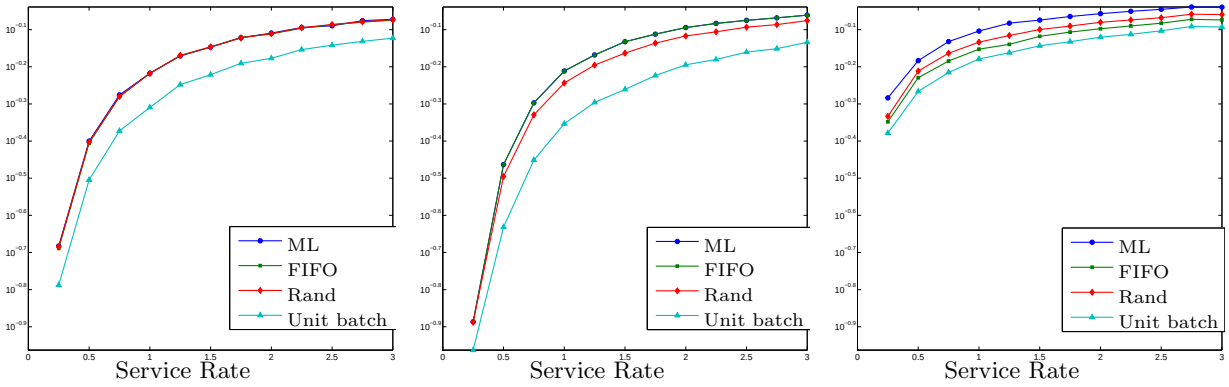
Theorem 5. (Optimality of Heuristics in Processor-Sharing Queues Under Random Matching). *Given two processor-sharing queues with job lengths J_k and J_m and supports $[\alpha_k, \beta_k]$ and $[\alpha_m, \beta_m]$, we have*

$$\begin{aligned} &J_k \overset{st}{\geq} J_m, \alpha_k \leq \alpha_m, \\ &\Rightarrow \rho_m \geq \rho_k, \mathbb{P}[B_m = 1] \leq \mathbb{P}[B_k = 1], P^{RAND}(k) \leq P^{RAND}(m). \end{aligned} \tag{15}$$

Proof: See Appendix J. □

The above results on comparison of two processor-sharing queues under random matching are identical to those comparing two infinite-server queues in Theorem 2. Hence, our heuristics are optimal for instrumentation under the above stochastic-order conditions when all the queues are either infinite-server or processor-sharing queues. However, when we have both infinite-server and processor-sharing queues, the above results are no longer valid and we consider this scenario in the next section.

[4] The tracking accuracy of $GI/M/1$ with first-come first-serve (FCFS) or last-come first-serve with preemption (LCFS-PR), which are part of a product-form network, is unity. This is because there is a fixed order of departures. Hence, they are ignored for instrumentation allocation.



(a) Shape Parameter $w = 1$ (b) Shape Parameter $w = 1.5$ (c) Shape Parameter $w = 0.5$

Figure 5: Matching Accuracies of ML, FIFO, and random matching together with the low-arrival rate approximation. See Sections 3.1 and 4.2.

7.2. Product-Form Network

We now compare monitoring performance of a processor-sharing queue with an infinite-server queue. This analysis is more complicated since the sojourn times of the two queues have different dependency structures. We limit to the scenario when the job lengths in the processor-sharing queue stochastically dominate the service times of the infinite-server queue.

Theorem 6. (Optimality in Product-Form Networks Under Random Matching). *Given a processor-sharing queue with job-lengths J_{PS} with support $[\alpha_{PS}, \beta_{PS}]$ and infinite-server queue with service T_{INF}, and arrivals X_{PS} and X_{INF},*

$$\begin{aligned} &X_{PS} \overset{st}{\leq} X_{INF},\ J_{PS} \overset{st}{\geq} T_{INF},\ \alpha_{PS} \leq \alpha_{INF} \\ &\Rightarrow \rho_{PS} \geq \rho_{INF},\ \mathbb{P}[B_{PS} = 1] \leq \mathbb{P}[B_{INF} = 1],\ P_{PS}^{RAND} \leq P_{INF}^{RAND}. \end{aligned} \tag{16}$$

Proof: See Appendix K □

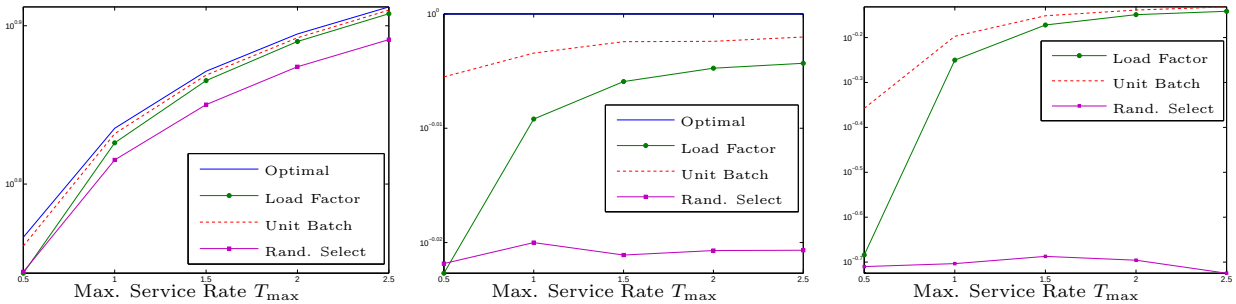
Hence, in a product-form network, under the above stochastic order, our two heuristics coincide with the optimal instrumentation strategy.

8. Numerical Analysis

We have so far provided a precise set of theoretical conditions when the two proposed heuristics coincide with the optimal instrumentation allocation strategy. In this section, we compare the performance of various instrumentation strategies through simulations. There are mainly two questions we seek to answer: How do our heuristics compare with the optimal solution when the theoretical conditions in Sections 6 and 7 for optimality are not met? What is the relative performance of the two heuristics in different load regimes?

We consider infinite-server queues with service distributions belonging to the Weibull family. The Weibull distribution is a rich family allowing us to tune the rate and the randomness of the service time separately by varying the scale and the shape parameters, and also includes the exponential distribution (shape parameter $w = 1$). Note that for the same scale parameter c, the variance decreases with the shape parameter w. Hence, distributions with $w < 1$ have higher variance than the exponential distribution, and vice versa.

Instrument $E = 2$ out of $|\mathcal{Q}| = 10$ states, unit arrival rate ($\lambda = 1$) of Poisson process, service rates $\mu_k \overset{i.i.d.}{\sim}$ Unif$[0.5, T_{\max}]$, Weibull shape parameter $w_k \overset{i.i.d.}{\sim}$ Unif$[0.1, 2]$, 1000 configurations.



(a) Obj. Value (b) Ratio $= \dfrac{\text{Obj. under heuristic}}{\text{Optimal Obj.}}$ (c) Fraction of Overlap with Opt.

Figure 6: Comparison of instrumentation strategies. Obj. $= E + \sum_{Q_k \in \mathcal{Q}} (1 - z_k) P^{\text{FIFO}}(k)$, see (3).

8.1. Effect of Matching Policies

In Fig.5, we compare the tracking accuracies P^γ of policies γ given by the FIFO, random matching and the optimal maximum-likelihood (ML) policies. We also compare the unit-batch approximation with the exact tracking accuracy. In Fig.5a, for the shape parameter $w = 1$, we have the exponential distribution, and all the matching policies, viz., ML, random, and FIFO matchings have equal performance, consistent with the analytical results in [9]. In Fig.5b, for the shape parameter $w > 1$, FIFO has the same performance as ML, and is better than random matching, again consistent with theory in [9][5]. In Fig.5c, for the shape parameter $w < 1$, we have heavy-tailed services, and here, random matching has better accuracy than FIFO rule. This is intuitive since out-of-order departures are more likely under heavy-tailed services. Moreover, the tracking accuracy in all these cases increases with the service rate as predicted.

In all the cases, there is a non-trivial gap between the actual tracking accuracies and the unit-batch approximation (up to about 10%); however, the approximation correctly follows the trend of the true values. Hence, we can expect solutions based on the exact and approximate evaluation to pick a similar set of queues for instrumentation, thereby leading to efficient allocation of monitoring resources, as discussed below.

8.2. Comparison of Instrumentation Strategies

In Fig.6, we compare our instrumentation strategies based on the load factor and the unit-batch approximation with the optimal strategy under the optimization rule in (3). As a benchmark, we also compare the proposed strategies with random instrumentation, i.e., uniformly selecting a subset of queues for instrumentation.

We consider Weibull service times and FIFO matching (similar results are observed under random matching). We run simulations under randomly chosen parameters for each queue and then average the results of different configurations. Specifically, the service rates are drawn i.i.d. uniformly between a minimum and a maximum service rate, and so are the shape parameters. We vary the maximum service rate to obtain more diverse set of service distributions for the queues under consideration for instrumentation allocation. Since the parameters are randomly chosen, the sufficient conditions for optimality of our heuristics proven in Section 6 are not met, and we do not expect our heuristics to exactly coincide with the optimal allocation strategy.

[5]It is shown in [9] that FIFO matching coincides with the optimal ML tracking when the shape parameter $w > 1$, i.e, there is less variation in service times.

In Fig.6, we see that the performance of the two heuristics gets closer to that of the optimal strategy as the maximum service rate increases leading to a more diverse set of queues. For the load-factor heuristic, this is because the load factors of different queues are well separated as the queues become more diverse. For the unit-batch heuristic, this is because, in addition, the service rates are increasing on average, leading to tighter approximation of the tracking accuracy. On the other hand, the gap between optimal allocation and random allocation increases with the maximum service rates since random allocation performs poorly when the queues are diverse. We also note that the performance of the unit-batch approximation is superior over the load-factor heuristic but they become close when the queues have well-separated load factors, as predicted in Section 4.

9. Conclusion

In this paper, we considered the problem of optimal instrumentation allocation for tracking transactions in a queueing network. Two types of monitoring resources are considered in the form of identifiers and timestamps. Identifiers provide precise tracking but are limited while timestamps are imprecise but available everywhere. The optimal allocation strategy selects queues with least timestamp-based tracking accuracies for introducing identifiers. We proposed two simple heuristics for allocation which coincides with the optimal strategy under certain conditions on arrival and service processes. Simulations show that our solutions are effective even when there is a deviation from the optimality conditions.

While providing a strong theoretical foundation and effective solutions for instrumentation allocation, we acknowledge that the overall problem has a broader range of challenges. For instance, in practice, the model for arrivals and services may not be known and needs to be estimated from data as well. There may be systems where complete timestamp information may not be available. We have assumed equal costs for instrumenting different components, while with unequal costs, we need to investigate new optimality conditions for our heuristics. We have assumed an infinite-server queueing system, while in reality there are a finite number of servers. The optimality results can in principle be extended to this scenario. However, direct analysis of such a system is much more involved since the service times of different packets are not independent. Moreover, the infinite-server system is the worst-case scenario for timestamp-based tracking since a finite-server system is less likely to produce out-of-order transactions. In this sense, the recommended instrumentation solution can be viewed as maximizing a lower bound on the tracking accuracy under finite-server queueing. Other challenges involve analyzing the effect of admission control and allowing for dynamic switching of data collection between different systems.

Acknowledgements

The authors thank R. Nunez Queija for discussions on the processor-sharing queue and Varun Gupta for discussions on the notion of convex order at the MAMA 2009 workshop.

Appendix A. Accuracy Under FIFO

Lemma 3. *The tracking accuracy in* (2) *simplifies under FIFO rule as*

$$P^{FIFO} = \sum_{b=1}^{\infty} \mathbb{P}[\boldsymbol{\pi}^t = \mathbf{I}, B = b],$$

where each term in the series $\mathbb{P}[\boldsymbol{\pi}^t = \mathbf{I}, B = b]$ *is given by*

$$= \mathbb{P}(\bigcap_{i=1}^{b-1} \{T(i) \in [X(i), X(i) + T(i+1)]\} \cap \{T(b) < X(b)\}),$$

where $X(i)$ *and* $T(i)$ *are the inter-arrival and service times.*

Proof: Given the busy-period size $B = b$, the event that FIFO rule is correct is

$$\mathcal{A}_b^{\text{FIFO}} = \bigcap_{i=1}^{b-1} \{T(i) < X(i) + T(i+1)\},$$

since i^{th} transaction needs to depart sooner than the $(i+1)^{\text{th}}$ transaction. The event that the busy-period size is $B = b$ is given by

$$\{B = b\} = \bigcap_{i=1}^{b-1} \{T(i) \in [X(i), \sum_{j=i}^{b} X(j)]\} \cap \{T(b) < X(b)\}.$$

$P^{\text{FIFO}} = \sum_{b=1}^{\infty} \mathbb{P}[\mathcal{A}_b^{\text{FIFO}} \cap \{B = b\}]$ and result follows. □

Appendix B. Accuracy Under Random Rule

I n order to compute the tracking accuracy P^{RAND} under random matching rule, we need to find the number of valid matchings. The number of such valid matchings is given by the number of perfect matchings in the 0-1 biadjacency matrix $\mathbf{A}_k$ defined as follows: for a bipartite graph with arrivals $\mathbf{Y}_k$ in one bipartition and departures $\mathbf{D}_k$ in the other, the presence of edge (i, j) in $\mathbf{A}_k$ indicates positive likelihood of i^{th} arrival corresponding to the j^{th} departure

$$A_k(i, j) = 1 \iff f_{T_k}[D_k(j) - Y_k(i)] > 0, \quad \forall 1 \leq i, j \leq b. \tag{B.1}$$

Any valid matching between the arrivals and the departures is a perfect matching on the biadjacency matrix $\mathbf{A}_k$, where a perfect matching is defined as a set of pairwise non-adjacent edges where all vertices are matched. The number of perfect matchings for the biadjacency matrix $\mathbf{A}$ is given by its permanent

$$\text{perm}(\mathbf{A}) := \sum_{\boldsymbol{\pi}} \prod_{i=1}^{b} A(i, \pi(i)), \tag{B.2}$$

where the sum is over all the permutation vectors $\boldsymbol{\pi}$ over $\{1, \ldots, b\}$ conditioned on busy period size $B = b$. Denote the perfect matching chosen by random matching as $\boldsymbol{\pi}^{\text{RAND}}$. Since each perfect matching is chosen with uniform probability and there are $\text{perm}(\mathbf{A})$ number of them, the probability of choosing one of them is $\text{perm}(\mathbf{A})^{-1}$. Using this fact, it is easy to now derive the expression for tracking accuracy under random matching

$$\begin{aligned} P^{\text{RAND}} &= \sum_{b=1}^{\infty} \mathbb{P}[\boldsymbol{\pi}^{\text{RAND}} = \boldsymbol{\pi}^t, B = b], \\ &= \sum_{b=1}^{\infty} \sum_{\mathbf{a}} \frac{\mathbb{P}[\mathbf{A} = \mathbf{a}, B = b]}{\text{perm}(\mathbf{a})}. \end{aligned} \tag{B.3}$$

Appendix C. Introduction to Stochastic Order

Appendix C.0.1. Stochastic Order

The stochastic order (also known as the usual stochastic order) is defined as follows [19, 20].

Definition 1 (Stochastic Order). *A variable Z_1 is said to be stochastically dominant with respect to a variable Z_2, denoted by $Z_1 \overset{st}{\geq} Z_2$, if*

$$Z_1 \overset{st}{\geq} Z_2 \iff \mathbb{E}[\phi(Z_1)] \geq \mathbb{E}[\phi(Z_2)], \tag{C.1}$$

for all increasing functions ϕ for which expectations exist.

Naturally, the above definition implies

$$Z_1 \overset{st}{\geq} Z_2 \Rightarrow \mathbb{E}[Z_1] \geq \mathbb{E}[Z_2]. \tag{C.2}$$

We intend to compare tracking accuracies at queues when their arrival processes satisfy a certain stochastic order and their service processes satisfy the reverse stochastic order. We leverage on the stochastic orders to guarantee an order on the tracking accuracies at different queues and hence, optimality of our heuristics.

Appendix C.0.2. Convex Order

We define another notion of comparison of random variables known as the *convex order* [19, Ch. 3].

Definition 2 (Convex Order). *A variable Z_1 is said to be smaller than Z_2, denoted by $Z_1 \underset{cx}{\leq} Z_2$, if for all convex functions $\phi : \Re \mapsto \Re$, $\mathbb{E}[\phi(Z_1)] \leq \mathbb{E}[\phi(Z_2)]$.*

The convex order compares the variability of random variables and requires equal mean values,

$$Z_1 \underset{cx}{\leq} Z_2 \Rightarrow \mathbb{E}[Z_1] = \mathbb{E}[Z_2], \mathrm{Var}[Z_1] \leq \mathrm{Var}[Z_2].$$

In our context, we intend to compare queues under the same load factor but with different variability in services. Intuitively, a service distribution with higher variability results in more uncertainty in the order of departures implying lower tracking accuracy, and we use the notion of convex order to capture this effect.

The stochastic and convex orders thus deal with different aspects of comparison of random variables: the former deals with the magnitudes while the latter deals with variability, and one does not imply the other. There are many sufficient conditions which can be easily checked for the stochastic or convex order to hold [19]. For a set of queues, we can use these conditions to check if the stochastic or the convex orders hold, in which case, we can draw conclusions about the optimality of our heuristics for instrumentation allocation.

Appendix D. Proof of Lemma 1

We have for $b \geq 1$,

$$\mathbb{P}[B_k = b] = \mathbb{P}[\bigcap_{i=1,\ldots,b-1} X_k(i) \leq T_k(i) \leq \sum_{j=i}^{b} X_k(j), X_k(b) > T_k(b)].$$

We have $\mathbb{P}[B_k > 1] = \bar{F}_{T_k}[X_k]$ and hence, $\mathbb{P}[B_k > 1] \geq \mathbb{P}[B_m > 1]$. Now consider,

$$\begin{aligned} p_k(x) &:= \mathbb{P}[B_k > b+1 | B_k > b, X_k(b+1) = x] \\ &= \mathbb{P}[T_k(b+1) \bigcup_{i=1}^{b} \{T_k(i) - \sum_{j=i}^{b} X_k(b)\} > x], \\ &= \mathbb{P}[\max\{T_k(b+1), T_k(b) - X_k(b), \ldots, \} > x]. \end{aligned} \tag{D.1}$$

We now claim that for $b \geq 1$,

$$X_k \overset{st}{\leq} X_m, T_k \overset{st}{\geq} T_m \Rightarrow p_k(x) \geq p_m(x). \tag{D.2}$$

This is because each term in (D.1) satisfies stochastic dominance for $i = 1, \ldots, b-1$,

$$X_k \overset{st}{\leq} X_m, T_k \overset{st}{\geq} T_m \Rightarrow T_k(i) - \sum_{j=i}^{b} X_k(b) \overset{st}{\geq} T_m(i) - \sum_{j=i}^{b} X_k(b).$$

Indeed the above terms are correlated, but they have the same dependency relationship for both queues Q_k and Q_m. Technically, this means that they share the same copula. The copula C for a multivariate variable $\mathbf{Z}$ is the mapping on the distribution functions such that

$$F_{\mathbf{Z}}(\mathbf{z}) = C[F_{Z(1)}(z(1)), F_{Z(2)}(z(2))\ldots]. \tag{D.3}$$

By [19, Thm. 6.B.14], under the same copula, we have the multivariate stochastic order

$$[T_k(b+1), T_k(b) - X_k(b), \ldots] \overset{st}{\geq} [T_m(b+1), T_m(b) - X_k(b), \ldots].$$

Hence, their maxima also satisfy stochastic order and (D.2) is true. Since $p_k(x)$ and $p_m(x)$ are decreasing in x, (7) holds. □

Appendix E. Proof of Lemma 2

Let $T' := \lambda T$ be the normalized service time and let $X'(i)$ be i.i.d. Poisson arrivals with unit rate. $\mathbb{P}[B > b|\mathbf{X}' = \mathbf{x}]$ is given by

$$\begin{aligned} &= \mathbb{P}[\max(T'(1), T'(2) + x(1), \ldots, T'(b) + \sum_{i=1}^{b-1} x(i)) > X'(b)] \\ &= 1 - \mathbb{E}[e^{-\max(T'(1), T'(2)+x(1), \ldots, T'(b)+\sum_{i=1}^{b-1} x(i))}]. \end{aligned} \tag{E.1}$$

Now, from convex order,

$$\begin{aligned} T'_k \underset{cx}{\leq} T'_m &\Rightarrow \max(T'_k(1), T'_k(2) + x(1), \ldots, T'_k(b) + \sum_{i=1}^{b-1} x(i))) \\ &\underset{cx}{\leq} \max(T'_m(1), T'_m(2) + x(1), \ldots, T'_m(b) + \sum_{i=1}^{b-1} x(i))). \end{aligned} \tag{E.2}$$

Since (E.1) is convex in the argument, it follows the same order of the service distributions. Since the convex order is closed under mixtures [19, Thm. 3.A.12], marginalizing over the arrival times $\mathbf{X}'$ preserves the order. Hence,

$$T'_k \underset{cx}{\leq} T'_m \Rightarrow \mathbb{P}[B_k > b] \leq \mathbb{P}[B_m > b],$$

which in turn is equivalent to a stochastic order. □

Appendix F. Proof of Theorem 1

Given the busy-period size $B = b$, denote the vector of spreads as $\mathbf{V}_k$, where the i^{th} element is given by

$$V_k(i) := T_k(i) - T_k(i+1), \quad 1 \leq i \leq b-1. \tag{F.1}$$

Note that the elements in the spread vector $\mathbf{V}_k$ have identical distributions but are dependent on one another, unlike the service times of the infinite-server queue which are independent. We have

$$P_b^{\text{FIFO}} = \mathbb{P}[\bigcap_{i=1}^{b-1} \{T_k(i) < X_k(i) + T_k(i+1)\}].$$

since i^{th} transaction needs to depart sooner than the $(i+1)^{\text{th}}$ transaction. From the definition of spread vector in (F.1), this is equal to

$$\begin{aligned} P_b^{\text{FIFO}} &= \mathbb{P}[\bigcap_{i=1}^{b-1}\{V_k(i) < X_k(i)\}], \\ &= \mathbb{P}[T_k(1) < T_k(2) < \ldots] + \mathbb{P}[\bigcap_{i=1}^{b-1}\{0 < V_k(i) < X_k(i)\}] \\ &= \frac{1}{b!} + \frac{1}{2}\mathbb{P}[\bigcap_{i=1}^{b-1}\{|V_k(i)| < X_k(i)\}], \end{aligned}$$

since V_k is symmetric around zero. We individually have

$$|V_k(i)| \overset{st}{\geq} |V_m(i)|,\ X_k(i) \overset{st}{\leq} X_m(i), \quad 1 \leq i \leq b,$$

which implies

$$|V_k(i)| - X_k(i) \overset{st}{\geq} |V_m(i)| - X_m(i), \quad 1 \leq i \leq b.$$

Since the spreads $V_k(1), V_k(2), \ldots$ are correlated, we use [19, Thm. 6.B.14] to prove the multi-variate stochastic order

$$|\mathbf{V}_k| - \mathbf{X}_k \overset{st}{\geq} |\mathbf{V}_m| - \mathbf{X}_m, \tag{F.2}$$

since $|\mathbf{V}_k| - \mathbf{X}_k$ and $|\mathbf{V}_m| - \mathbf{X}_m$ share the same copula, defined in (D.3). From (F.2),

$$P_b^{\text{FIFO}}(k) \overset{st}{\leq} P_b^{\text{FIFO}}(m)$$

This implies the order of tracking accuracies in (10) by marginalizing over the busy-period sizes since P_b^{FIFO} decreases in busy period b and the busy periods satisfy stochastic order, from Lemma 1. □

Appendix G. Proof of Theorem 2

In order for (11) to hold, it suffices to show that

$$\text{perm}(\mathbf{A}_k)|\{B_k = b\} \overset{st}{\geq} \text{perm}(\mathbf{A}_m)|\{B_m = b\}, \tag{G.1}$$

since the tracking accuracy under random matching is given by (B.3), and taking expectation over B_m and B_k preserves the order since $B_k \overset{st}{\geq} B_m$ from Lemma 1. Since the perm($\mathbf{A}$) is the number of matchings for biadjacency matrix $\mathbf{A}$, more edges in $\mathbf{A}$ implies higher perm($\mathbf{A}$). Let $[\alpha_k, \beta_k]$ be the support of T_k and $[\alpha_m, \beta_m]$ of T_m. From (B.1) for k, the departure of i^{th} arrival has an edge with j^{th} arrival, for $1 \leq i < j \leq b$ iff.

$$\alpha_k \leq T_k(i) - \sum_{a=i}^{j-1} X_k(a) \leq \beta_k. \tag{G.2}$$

By definition of support bound, $T_k(i) \leq \beta$ a.s. Hence, the upper bound in (G.2) always holds. Since $X_k(i) \geq 0$, we have the probability of edge as

$$\mathbb{P}[A(i,j) = 1] = \bar{F}_{T_k}[\alpha_k + \sum_{a=i}^{j-1} X_k(a)]. \tag{G.3}$$

Conditioning on the same arrival realizations $\mathbf{X}_k, \mathbf{X}_m = \mathbf{x}$, from the definition of stochastic dominance,

$$\begin{aligned}\bar{F}_{T_k}[\alpha_k + \sum_{a=i}^{j-1} x(a)] &\geq \bar{F}_{T_m}[\alpha_k + \sum_{a=i}^{j-1} x(a)] \\ &\geq \bar{F}_{T_m}[\alpha_m + \sum_{a=i}^{j-1} x(a)],\end{aligned}$$

when $\alpha_k \leq \alpha_m$. Now since the functions are decreasing in x and $X_k \overset{st}{\leq} X_m$, the order is preserved on removing the conditioning. Hence, (G.1) holds implying (11). □

Appendix H. Proof of Corollary 1

Let $T_i' := \lambda T_i$ for $i = 1, 2$ be the normalized service times and let $X'(i)$ be i.i.d. arrivals with unit rate. For any positive variable T_1' and $T_2' = cT_1'$ with $0 < c < 1$, we have $T_1' \overset{st}{\geq} T_2'$. First consider FIFO matching rule,

$$|V_1| \overset{st}{\geq} |V_2| = c|V_1|, \quad \forall 0 < c \leq 1, \tag{H.1}$$

and hence, conditions in Theorem 1 for the order of accuracies under FIFO matching is satisfied.

For random matching rule, let $[\alpha, \beta]$ be the support of T_1'. We have $\alpha > c\alpha$, and hence, the condition in Theorem 2 is in fact, violated. We revisit the probability of having an edge in the biadjacency matrix $\mathbf{A}$

$$\mathbb{P}_{T_1'}[A(i,j) = 1] = \bar{F}_{T_1'}[\alpha + \sum_{k=i}^{j-1} x(i)].$$

For the service time $T_2' = cT_1'$ with $c < 1$, we have

$$\mathbb{P}_{T_2'}[A(i,j) = 1] = \bar{F}_{T_1'}[\alpha + \frac{1}{c}\sum_{k=i}^{j-1} x(i)] \leq \mathbb{P}_{T_1'}[A(i,j) = 1],$$

and hence, the result holds. □

Appendix I. Proof of Theorem 4

Let $T' := \lambda T$ be the normalized service time and let $X'(i)$ be i.i.d. Poisson arrivals with unit rate. Given the busy-period size $B = b$, we have

$$\begin{aligned}P_b^{\text{FIFO}}|\{B = b\} &= \mathbb{P}[\bigcap_{i=1}^{b-1}\{T'(i) - T'(i+1) < X'(i)\}] \\ &= \mathbb{E}[\exp(-\sum_{i=1}^{b}(T'(i) - T'(i+1))^+)] \\ &= \mathbb{E}[\exp(-\sum_{i=1}^{b} a_{i,\pi}T'(i))|\mathbf{\Pi}(\mathbf{T}') = \boldsymbol{\pi}],\end{aligned}$$

where $a_{i,\pi} = 0, \pm 1$ are fixed coefficients conditioned on the event that the service times $\mathbf{T}'$ follow a certain permutation $\boldsymbol{\pi}$. Now $\exp(-\sum_{i=1}^{b} a_{i,\pi}T'(i))$ is a concave function of $\sum_{i=1}^{b} a_{i,\pi}T'(i)$ and all permutations $\boldsymbol{\pi}$ of the service times are equiprobable at both the queues (since all the service times are i.i.d.).

On the lines of [19, Thm. 3.A.19], we can show that when $\mathbf{T}'_k$ and $\mathbf{T}'_m$ are conditioned on the same permutation $\boldsymbol{\pi}$,

$$T'_k \underset{cx}{\leq} T'_m \Rightarrow \sum_{i=1}^{b} a_{i,\pi} T'_k(i) \underset{cx}{\leq} \sum_{i=1}^{b} a_{i,\pi} T'_m(i).$$

Hence,

$$T'_k \underset{cx}{\leq} T'_m \Rightarrow P_b^{\text{FIFO}}(k)|\{B_k = b\} \geq P_b^{\text{FIFO}}(m)|\{B_m = b\}.$$

Since P_b^{FIFO} is decreasing in b and the busy-period sizes at k and m follow the stochastic order, the order carries through when we marginalize over the busy-period sizes. □

Appendix J. Proof of Theorem 5

Let $\mathbf{T}_k$ and $\mathbf{T}_m$ be the sojourn times of the jobs in the two queues. The sojourn times satisfy

$$X_k \overset{st}{\leq} X_m,\, J_k \overset{st}{\geq} J_m \Rightarrow T_k(i) \overset{st}{\geq} T_m(i).$$

Now $\mathbf{T}_k$ and $\mathbf{T}_m$ are correlated, unlike the infinite-server case. However, $\mathbf{T}_k$ and $\mathbf{T}_m$ have the same copula since they are both processor sharing queues and by [19, Thm. 6.B.14],

$$X_k \overset{st}{\leq} X_m,\, J_k \overset{st}{\geq} J_m \Rightarrow \mathbf{T}_k \overset{st}{\geq} \mathbf{T}_m.$$

On lines of Lemma 1,

$$X_k \overset{st}{\leq} X_m,\, \mathbf{T}_k \overset{st}{\geq} \mathbf{T}_m \Rightarrow B_k \overset{st}{\geq} B_m.$$

Note that the lower bound of support of each sojourn time is the same as the job lengths. On lines of Appendix G, (15) holds. □

Appendix K. Proof of Theorem 6

We first provide a result that under the stochastic dominance assumption, the sojourn times of the processor-sharing queue dominate those of the infinite-server queue.

Proposition 2. (Sojourn Times in Infinite Server and Processor Sharing Queues). *We have*

$$J_{PS} \overset{st}{\geq} T_{INF} \Rightarrow \mathbf{T}_{PS} \overset{st}{\geq} \mathbf{T}_{INF}. \tag{K.1}$$

Proof: The multivariate ordering is implied by the conditional ordering

$$T_{\text{PS}}(i)|\bigcap_{k=1}^{i-1}\{T_{\text{PS}}(k) = t_k\} \overset{st}{\geq} T_{\text{INF}}.$$

Now the sojourn times $T_{\text{PS}}(i)$ at the processor-sharing queue are at least the job lengths with probability 1. Hence, upon any conditioning

$$T_{\text{PS}}(i)|\bigcap_{k=1}^{i-1}\{T_{\text{PS}}(k) = t_k\} \overset{st}{\geq} J_{\text{PS}}(i), \quad i = 1, 2, \ldots.$$

Hence, the result in (K.1) holds. □

The above result follows the intuition that when larger jobs are arriving to the processor-sharing queue than to the infinite-server queue, the sojourn times in the processor-sharing queue are longer. However, the

converse is not always true since even longer jobs can have shorter sojourn times in the infinite-server queue due to simultaneous processing of the jobs.

We now use the above proposition to provide a result on the busy-period sizes. From (K.1), we have the multivariate stochastic order. Now,

$$\mathbf{X}_{\text{PS}} \overset{st}{\leq} \mathbf{X}_{\text{INF}}, \mathbf{T}_{\text{PS}} \overset{st}{\geq} \mathbf{T}_{\text{INF}} \Rightarrow B_{\text{PS}} \overset{st}{\geq} B_{\text{INF}},$$

on lines of Lemma 1. On lines of Theorem 2, we have (16). □

[1] A. Anandkumar, C. Bisdikian, T. He, D. Agrawal, Selectively Retrofitting Monitoring in Distributed Systems, Seattle, USA, 2009, workshop on Mathematical Performance Modeling and Analysis (MAMA).

[2] P. Bernstein, E. Newcomer, Principles of Transaction Processing, Morgan Kaufmann, 1997.

[3] A. Borr, Transaction monitoring in Encompass [TM]: Reliable distributed transaction processing, in: Proc. of VLDB, 1981.

[4] A. Spector, J. Swainson, D. Sabbah, Introduction to Transaction Processing, IBM Systems Journal 43 (2) (2004) 207–208.

[5] M. Aguilera, J. Mogul, J. Wiener, P. Reynolds, A. Muthitacharoen, Performance debugging for distributed systems of black boxes, in: Proc. of SOSP, 2003, pp. 74–89.

[6] E. Thereska, B. Salmon, J. Strunk, M. Wachs, M. Abd-El-Malek, J. Lopez, G. Ganger, Stardust: tracking activity in a distributed storage system, in: Proc. of ACM Sigmetrics, 2006, pp. 3–14.

[7] Application Response Management, `http://www.opengroup.org/tech/management/arm`.

[8] P. Barham, A. Donnelly, R. Isaacs, R. Mortier, Using Magpie for request extraction and workload modelling, in: Symp. on Operating Sys. Design & Implementation, 2004.

[9] A. Anandkumar, C. Bisdikian, D. Agrawal, Tracking in a Spaghetti Bowl: Monitoring Transactions Using Footprints, in: Proc. of ACM SIGMETRICS, Annapolis, Maryland, USA, 2008.

[10] J. Joyce, G. Lomow, K. Slind, B. Unger, Monitoring Distributed Systems, ACM Tran. on Computer Systems 5 (2) (1987) 121–150.

[11] K. Chandy, L. Lamport, Distributed Snapshots: Determining Global States of Distributed Systems, ACM Tran. on Computer Systems 3 (1) (1985) 63–75.

[12] M. Mansouri-Samani, M. Sloman, Monitoring distributed systems, Network, IEEE 7 (6) (1993) 20–30.

[13] M. Schmid, M. Thoss, T. Termin, R. Kroeger, A Generic Application-Oriented Performance Instrumentation for Multi-Tier Environments, in: IEEE Intl. Symposium on Integrated Network Management, 2007, pp. 304–313.

[14] M. Chen, E. Kiciman, E. Fratkin, A. Fox, E. Brewer, Pinpoint: problem determination in large, dynamic Internet services, Dependable Sys. & Networks (2002) 595–604.

[15] M. Chen, A. Accardi, E. Kiciman, J. Lloyd, D. Patterson, A. Fox, E. Brewer, Path-based failure and evolution management, in: Symposium on Networked System Design and Implementation, 2004, pp. 23–23.

[16] H. Liu, H. Zhang, R. Izmailov, G. Jiang, X. Meng, Real-time Application Monitoring and Diagnosis for Service Hosting Platforms of Black Boxes, in: IEEE Intl. Symposium on Integrated Network Management, 2007, pp. 216–225.

[17] B. Tak, C. Tang, C. Zhang, S. Govindan, B. Urgaonkar, R. Chang, vPath: Precise Discovery of Request Processing Paths from Black-Box Observations of Thread and Network Activities, in: Proc. of Usenix Annual Tech. Conf., 2009.

[18] M. Shaked, J. Shanthikumar, Stochastic Orders And Their Applications, Academic Press, 1994.

[19] M. Shaked, J. Shanthikumar, Stochastic Orders, Springer, 2007.

[20] A. Müller, D. Stoyan, Comparison Methods for Stochastic Models and Risks, Wiley, 2002.

[21] F. Baskett, K. Chandy, R. Muntz, F. Palacios, Open, closed, and mixed networks of queues with different classes of customers, J. Assoc. Comput. Mach 22 (2) (1975) 248–260.