Resource Allocation in One-dimensional Distributed Service Networks with Applications

Nitish K Panigrahy, Prithwish Basu, Philippe Nain, Don Towsley, Ananthram Swami, Kevin S Chan, Kin K Leung

Resource Allocation in One-dimensional Distributed Service Networks with Applications

Nitish K. Panigrahy[a,*], Prithwish Basu[b], Philippe Nain[c], Don Towsley[a], Ananthram Swami[d], Kevin S. Chan[d] and Kin K. Leung[e]

[a]University of Massachusetts Amherst, MA 01003, USA
[b]Raytheon BBN Technologies, Cambridge, MA 02138, USA
[c]Inria, 06902 Sophia Antipolis Cedex, France
[d]Army Research Laboratory, Adelphi, MD 20783, USA.
[e]Imperial College London, London SW72AZ, UK.

ARTICLE INFO

Keywords:
Resource Allocation
1-D service network
Queueing Theory
Distributed Network
Dynamic Programming

ABSTRACT

We consider assignment policies that allocate resources to users, where both resources and users are located on a one-dimensional line $[0, \infty)$. First, we consider unidirectional assignment policies that allocate resources only to users located to their left. We propose the Move to Right (*MTR*) policy, which scans from left to right assigning nearest rightmost available resource to a user, and contrast it to the Unidirectional Gale-Shapley (*UGS*) matching policy. While both policies among all unidirectional policies, minimize the expected distance traveled by a request (*request distance*), *MTR* is fairer. Moreover, we show that when user and resource locations are modeled by statistical point processes, and resources are allowed to satisfy more than one user, the spatial system under unidirectional policies can be mapped into bulk service queueing systems, thus allowing the application of many queueing theory results that yield closed form expressions. As we consider a case where different resources can satisfy different numbers of users, we also generate new results for bulk service queues. We also consider bidirectional policies where there are no directional restrictions on resource allocation and develop an algorithm for computing the optimal assignment which is more efficient than known algorithms in the literature when there are more resources than users. Numerical evaluation of performance of unidirectional and bidirectional allocation schemes yields design guidelines beneficial for resource placement. Finally, we present a heuristic algorithm, which leverages the optimal dynamic programming scheme for one-dimensional inputs to obtain approximate solutions to the optimal assignment problem for the two-dimensional scenario and empirically yields request distances within a constant factor of the optimal solution.

1. Introduction

The past few years have witnessed significant growth in the use of distributed network analytics involving agile code, data and computational resources. In many such networked systems, for example, Internet of Things [5], a large number of computational and storage resources are widely distributed in the physical world. These resources are accessed by various end users/applications that are also distributed over the physical space. Assigning users or applications to resources efficiently is key to the sustained high-performance operation of the system.

In some systems, requests are transferred over a network to a server that provides a needed resource. In other systems, servers are mobile and physically move to the user making a request. Examples of the former type of service include accessing storage resources over a wireless network to store files and requesting computational resources to run image processing tasks; whereas an example of the latter type of service is the arrival of ride-sharing vehicles to the user's location over a road transportation network.

Not surprisingly, the spatial distribution of resources and users[1] in the network is an important factor in determining the overall performance of the service. A key measure of performance is *average request distance*, that is average

*Corresponding author

**The material in this paper was presented in part at the IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS), Rennes, France in 2019 and in Workshop on MAthematical performance Modeling and Analysis (MAMA 2018).

nitish@cs.umass.edu (N.K. Panigrahy); prithwish.basu@raytheon.com (P. Basu); philippe.nain@inria.fr (P. Nain); towsley@cs.umass.edu (D. Towsley); ananthram.swami.civ@mail.mil (A. Swami); kevin.s.chan.civ@mail.mil (K.S. Chan); kin.leung@imperial.ac.uk (K.K. Leung)

[1]We use the terms "users" and "requesters" interchangeably and same holds true for the terms "resources" and "servers".

distance between a user and its allocated resource/server (where distance is measured on the network). This directly translates to latency incurred by a user when accessing the service, which is arguably among the most important criteria in distributed service applications. For example, in wireless networks, signal attenuation is strongly coupled to request distance, therefore developing allocation policies to minimize request distance can help reduce energy consumption, an important concern in battery-operated wireless networks. Another important practical constraint in distributed service networks is *service capacity*. For example, in network analytics applications, a networked storage device can only support a finite number of concurrent users; similarly, a computational resource can only support a finite number of concurrent processing tasks. Likewise, in physical service applications like ride-sharing, a vehicle can pick up a finite number of passengers at once.

Therefore, a primary problem in such distributed service networks is to efficiently assign each user to a suitable resource so as to minimize average request distance and ensure no resource serves more users than its capacity. If the entire system is being managed by a single administrative entity such as a ride sharing service, or a datacenter network where analytics tasks are being assigned to available CPUs, there are economic benefits in minimizing the average request distance across all (user, resource) pairs, which is tantamount to minimizing the average delay in the system.

The general version of this capacitated assignment problem can be solved by modeling it as a *minimum cost flow* problem on graphs [4] and running the *network simplex algorithm* [17]. However, if the network has a low-dimensional structure and some assumptions about the spatial distributions of users and resources hold, more efficient methods can be developed.

In this paper, we consider two one-dimensional network scenarios that motivate the study of this special case of the user-to-resource assignment problem.

The first scenario is ride-hailing on a one-way street where vehicles move right to left. If the vehicles of a ride-sharing company are distributed along the street at a certain time, and users equipped with smartphone ride-hailing apps request service, the system attempts to assign vehicles with spare capacity located towards the right of the users so as to minimize average “pick up" distance. Abadi et al. [1] introduced this problem and presented a policy known as Unidirectional Gale-Shapley[2] matching (*UGS*) minimize average pick up distance. In this policy, all users concurrently emit rays of light toward their right and each user is matched with the vehicle that first receives the emitted ray. While the well-known Gale-Shapley matching algorithm [9] matches user-resource pairs that are mutually nearest to each other, its unidirectional variant, UGS, matches a user to the nearest resource on its right. Note that, this one-dimensional network setting also applies to vehicular wireless ad-hoc networks on a one-lane roadway [11, 14][3], where users are in vehicles and servers are attached to fixed infrastructure such as lamp posts. Users attempt to allocate their computation tasks over the wireless network to servers located to their right so that they can retrieve the results with little effort while driving by.

In this paper, we propose another policy “Move to Right” policy (or *MTR*) which has the same “expected distance traveled by a request” (*request distance*) as UGS but has a lower variance. *MTR* sequentially allocates users to the geographically nearest available vehicle located to his/her right. When user and resource locations are modeled by statistical point processes the one-dimensional unidirectional space behaves similar to time and notions from queueing theory can be applied. In particular, when user and vehicle locations are modeled by independent Poisson processes, average request distance can be characterized in closed form by considering inter-user and inter-server distances as parameters of a *bulk service* M/M/1 queue where the bulk service capacity denotes the maximum number of users that can be handled by a server. We equate request distance in the spatial system to the expected *sojourn time* in the corresponding queuing model[4]. This natural mapping allows us to use well-known results from queueing theory and in some cases to propose new queueing theoretic models to characterize request distances for a number of interesting situations beyond M/M/1 queues.

A natural extension to our spatial framework is to consider more general communication costs associated with each resource allocation. Assuming communication cost for each allocation is a function of request distance, we provide closed form expressions for the expected communication cost for specific user-server distributions and specific server capacities.

The second scenario involves a convoy of vehicles traveling on a one-dimensional space, for example, trucks on a highway or boats on a river. Some vehicles have expensive camera sensors (image/video) but have inadequate com-

[2] We rename *queue matching* defined in [1] as Unidirectional Gale-Shapley Matching to avoid overloading the term *queue*.

[3] Furthermore, [11] confirms that vehicle location distribution on the streets in Central London can be closely approximated by a Poisson distribution.

[4] Sojourn time is the sum of waiting and service times in a queue.

putational storage or processing power. On the other hand, cheap storage and processing is easily available on several other vehicles. The cameras periodically take photos/videos as they move through space and want them processed / stored. In such case, bidirectional assignment schemes are more suitable. Since no directionality restrictions are imposed on the allocation algorithms, computing the optimal assignment is not as simple as in the unidirectional case.

We explore the special structure of the one-dimensional topology to develop an optimal algorithm that assigns a set of requesters R to a set of resources S such that the total assignment cost is minimized. This problem has been recently solved for $|R| = |S|$ [7]. However, we are interested in the case when $|R| < |S|$. We propose a dynamic Programming based algorithm which solves this case with time complexity $O(|R|(|S| - |R| + 1))$. Note that other assignment algorithms in literature such as the Hungarian primal-dual algorithm and Agarwal's variant [3] have time complexities $O(|R|^3)$ and $O(|R|^{2+\epsilon})$ respectively and assume $|R| = |S|$ for general and Euclidean distance measures.

We leverage the optimal dynamic programming scheme for one-dimensional inputs to obtain approximate solutions to the optimal assignment problem for the two-dimensional scenario where users and servers are located on the two-dimensional plane, $\mathbb{R}^2$. More precisely, we *embed* the points denoting $R, S \subset \mathbb{R}^2$ into new locations in $\mathbb{R}$ such that the distances between a user and its nearest servers are approximately preserved. Our approximation algorithm empirically yields request distances within a constant factor of the optimal solution with $O(|R|^2)$ time complexity.

Our contributions are summarized below:

1. Analysis of simple unidirectional allocation policies *MTR* and *UGS* yielding closed form expressions for mean request distance.
    - When inter-requester and inter-resource distances are exponentially distributed, we model unidirectional policies as a bulk service M/M/1 queue.
    - When inter-requester distances are generally distributed but the inter-resource distances are exponentially distributed, we model the situation using an accessible batch service G/M/1 queue.
    - When inter-requester distances are exponentially distributed but inter-resource distances are generally distributed, we model the spatial system as an accessible batch service M/G/1 queue with the first batch having exceptional service time. To the best of our knowledge this system has not been studied previously in the queueing theory literature.
    - We include several generalizations of our framework. In the first place we discuss a simulation driven conjecture for evaluating request distance for general distance distributions under heavy traffic. We also investigate the heterogeneous server capacity scenario where server capacity is a random variable and to the best of our knowledge this system has not been studied previously in the queueing theory literature. We derive expressions for expected request distance when servers have infinite capacity. We include communication cost associated with each resource allocation and provide a closed form expression for expected communication cost for some specific scenarios. Finally we extend our framework to compute expected request distance for the case where each user requests two resources residing in two different set of servers, by mapping it to a two queue fork-join system.
2. A novel algorithm for optimal (bidirectional) assignment with time complexity $O(|R|(|S| - |R| + 1))$.
3. A numerical and simulation study of different assignment policies: UGS , MTR, bi-directional heuristic allocation policies (Gale-Shapley and Nearest Neighbor) and the optimal policy.
4. A heuristic based approximate solution to the optimal assignment problem for the two-dimensional scenario with an empirically observed constant factor approximation of the optimal solution.

The paper is organized as follows. The next section discusses related work. Section 3 contains technical preliminaries. We show the equivalence of UGS and MTR w.r.t expected request distance in Section 4, and present results associated with the case when servers are Poisson distributed in Section 5. In Section 6, we develop formulations for expected request distance when either user or server placements are described by Poisson processes. We include some generalizations of our framework such as analysis under general distance distributions, results for heterogeneous server capacity and uncapacitated allocation in Section 7. The optimal bidirectional allocation strategy is presented in Section 8. We compare the performance of various local allocation strategies in Section 9. In Section 10, we extend our one-dimensional framework to solve two-dimensional problem. We conclude the paper in Section 11.

2. Related Work

Poisson Matching: Holroyd et al. [12] first studied translation invariant matchings between two d-dimensional Poisson processes with equal densities. Their primary focus was obtaining upper and lower bounds on expected matching distance for stable matchings. Abadi et al. [1] introduced "Unidirectional Gale-Shapley" matching (*UGS*) and derived bounds on the expected matching distance for stable matchings between two one-dimensional Poisson processes with different densities. In this paper, we propose another unidirectional allocation policy: "Move To Right" policy (*MTR*) and provide explicit expressions for the expected matching distance for both MTR and UGS when either requesters or servers are distributed according to a renewal process and the according to a Poisson process.
Exceptional Queueing Systems and Accessible Batches: Welch et al. [22] first studied an M/G/1 queue where a customer arriving when the server is idle has a different service time than the others. Bulk service M/G/1 queues has been studied in [6]. Authors in [10] analyzed a bulk service G/M/1 queue with accessible or non-accessible batches where an accessible batch is considered to be a batch in service allowing subsequent arrivals, while the service is on. In this work, we model the spatial system using an accessible batch service queue with the first batch having exceptional service time. To the best of our knowledge this system has not been studied previously in queueing theory literature.
Euclidean Bipartite Matching: The optimal user-server assignment problem can be modeled as a minimum-weight matching on a weighted bipartite graph where weights on edges are given by the Euclidean distances between the corresponding vertices [15]. Well-known polynomial time solutions exist for this problem, such as the modified Hungarian algorithm proposed by Agarwal et al. [3] with a running time of $O(|R|^{2+\epsilon})$, where $|R|$ is the total number of users. In the case of an equal number of users and servers, the optimal user-server assignment on a real line is known [7]. In this paper, we consider the case when there are fewer users than servers.

3. Technical Preliminaries

Consider a set of users R and a set of servers S. Each user makes a request that can be satisfied by any server. Assume that each server $j \in S$ has capacity $c_j \in \mathbb{Z}^+$ corresponding to the maximum number of requests that it can process. Suppose users and servers are located on a line $\mathcal{L}$. Formally, let $r : R \to \mathcal{L}$ and $s : S \to \mathcal{L}$ be the location functions for users and servers, respectively, such that a distance $d_{\mathcal{L}}(r, s)$ is well defined for all pairs $(r, s) \in R \times S$. Initially we assume that all servers have equal capacities i.e. $c_j = c \ \forall j \in S$. Later in Section 7.2 we extend our analysis to a case in which server capacities are integer random variables.

3.1. User and server spatial distributions

Let $0 \le r_1 \le r_2 \le \cdots$ represent user locations and $0 \le s_1 \le s_2 \le \cdots$ be the server locations. Let $X_j = s_j - s_{j-1}, j \ge 1, s_0 = 0$, denote the inter-server distances and $Y_i = r_i - r_{i-1}, i \ge 1, r_0 = 0$, the inter-user distances. We assume $\{X_j\}_{j\ge1}$ to be a renewal process with cumulative distribution function (cdf)

$$\mathbb{P}(X_j \le x) = F_X(x). \tag{1}$$

We also assume $\{Y_i\}_{i\ge1}$ to be a renewal process with cdf $F_Y(x)$, i.e.,

$$\mathbb{P}(Y_i \le x) = F_Y(x). \tag{2}$$

We denote $\alpha_X = 1/\mu$ and σ_X^2 to be the mean and variance associated with F_X. Similarly let $\alpha_Y = 1/\lambda$ and σ_Y^2 be the mean and variance associated with F_Y. We let $\rho = \lambda/\mu$ and assume that $\rho < c$. Denote by $F_X^*(s) = \int_0^\infty e^{-sx} dF_X(x)$ and $F_Y^*(s)$ the Laplace-Stieltjes transform (*LST*) of F_X and F_Y with $s \ge 0$.

In our paper, we consider various inter-server and inter-user distance distributions, including exponential, deterministic, uniform and hyperexponential.

3.2. Allocation policies

One of our goals is to analyze the performance of various request allocation policies using expected request distance as a performance metric. We define various allocation policies as follows.

- **Unidirectional Gale-Shapley (*UGS*):** In UGS, each user simultaneously emits a ray to their right. Once the ray hits an unallocated server s, the user is allocated to s.

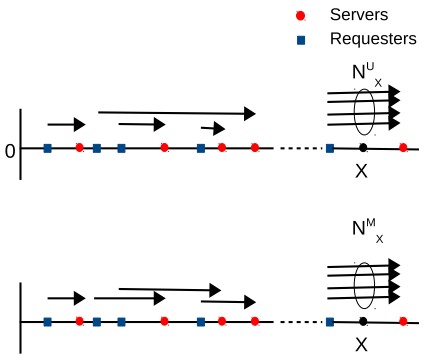


Figure 1: Allocation of users to servers on the one-dimensional network. Top: UGS, Bottom: MTR allocation policy.

- **Move To Right** (*MTR*)**:** In MTR, starting from the left, each user is allocated sequentially to the nearest available server to its right.
- **Nearest Neighbor** (*NN*) **[21]:** In this matching, starting from the left, each user is allocated sequentially to the nearest available server. This policy can be viewed as the bidirectional version of MTR policy.
- **Gale-Shapley** (*GS*) **[9]:** In this matching, each user selects the nearest server and each server selects its nearest user. Remove reciprocating pairs, and continue.
- **Optimal Matching:** This matching minimizes average request distance among all feasible allocation policies.

4. Unidirectional Allocation Policies

In this Section, we establish the equivalence of UGS and MTR w.r.t number of requests that traverse a point and expected request distance. Define N_x^P and D_i^P to be random variables for the number of requests that traverse point $x \in \mathcal{L}$ and distance between user i and its allocated server under policy P, respectively. Thus N_x^U and N_x^M denote the number of requests that traverse point $x \in \mathcal{L}$ under UGS and MTR, respectively, as shown in Figure 1. Consider the following definition of busy cycle in a service network.

Definition 1. *A busy cycle for a policy P is an interval* $I = [a, b] \subset \mathcal{L}$ *such that* $\exists\ i, j$ *with* $r_i = a, s_j = b$ *for which* $N_x^P > 0, \forall x \in I$ *and* $N_x^P = 0$ *for* $x = a - \epsilon$ *and* $x = b + \epsilon$ *with* ϵ *being an infinitesimal positive value.*

We have the following theorem.

Theorem 1. $N_x^U = N_x^M, x \geq 0.$

Proof. Due to the unidirectional nature of matching, both UGS and MTR have the same set of busy cycles. Denote $\mathcal{I}$ as the set of all busy cycles in the service network. In the case when $x \in \mathcal{L} \setminus \bigcup_{I \in \mathcal{I}} I$ we already have $N_x^U = N_x^M = 0$. Let us now consider a busy cycle $I^U = [a^U, b^U]$ under UGS policy. Let $x \in I^U$. Let $L_{x,R}^U = |\{r_i | a^U \leq r_i \leq x\}|$ and $L_{x,S}^U = |\{s_j | a^U \leq s_j \leq x\}|$. $N_x^U = L_{x,R}^U - L_{x,S}^U$. Similarly define $L_{x,R}^M$ and $L_{x,S}^M$ for MTR policy. Clearly $N_x^M = L_{x,R}^M - L_{x,S}^M$. As both policies have the same set of busy cycles we have $L_{x,R}^U = L_{x,R}^M$ and $L_{x,S}^U = L_{x,S}^M$. Thus we get

$$N_x^U = N_x^M,\ x \in \mathbb{R}^+, \tag{3}$$

□

Corollary 1. $\mathbb{E}[D^U] = \mathbb{E}[D^M]$ *i.e. the expected request distances are the same for both UGS and MTR under steady state.*

Distribution	Parameters	$F_X(x)$	$B(x)$
Exponential	μ: rate	$1 - e^{-\mu x}$	$\frac{1}{\lambda}\left[1 - e^{-\lambda x}\right] - \frac{1}{\lambda+\mu}\left[1 - e^{-(\lambda+\mu)x}\right]$
Uniform	b : maximum value	$x/b, \quad 0 \leq x \leq b$	$\frac{1}{\lambda^2 b}\left[1 - e^{-\lambda b}\right] - \frac{e^{-\lambda x}}{\lambda}$
Deterministic	d_0 : constant	$1, \quad x \geq d_0$	$\frac{e^{-\lambda d_0} - e^{-\lambda x}}{\lambda}$
Hyper -exponential	l: order p_j : phase probability μ_j : phase rate	$1 - \sum_{j=1}^{l} p_j e^{-\mu_j x}$	$\frac{1}{\lambda}\left[1 - e^{-\lambda x}\right] - \sum_{j=1}^{l} \frac{p_j}{\lambda+\mu_j}\left[1 - e^{-(\lambda+\mu_j)x}\right]$

Table 1
Properties of specific inter-server distance distributions.

Proof. Under steady state both N_x^U and N_x^M converge to a random variable. Applying Little's law we have $\mathbb{E}[D^U] = \mathbb{E}[D^M]$. □

Remark 1. *Note that Theorem 1 applies to any inter-server or inter-user distance distribution. It also applies to the case where servers have capacity $c > 1$.*

Remark 2. *Although MTR and UGS are equivalent w.r.t. the expected request distance, MTR tends to be fairer, i.e., has low variance*[5] *w.r.t. request distance.*

5. Unidirectional Poisson Matching

In this section, we characterize request distance statistics under unidirectional policies when both users and servers are distributed according to two independent Poisson processes. We first analyze MTR as follows.

5.1. MTR

Under this allocation policy, the service network can be modeled as a bulk service M/M/1 queue. A bulk service M/M/1 queue provides service to a group of c or fewer customers. The server serves a bulk of at most c customers whenever it becomes free. Also customers can join an existing service if there is room which is an example of accessible batch. In Section 6 we describe the notion of accessible batches in greater detail. The service time for the group is exponentially distributed and customer arrivals are described by a Poisson process. The distance between two consecutive users in the service network can be thought of as inter-arrival time between customers in the bulk service M/M/1 queue. The distance between two consecutive servers maps to a bulk service time.

Having established an analogy between the service network and the bulk service M/M/1 queue, we now define the state space for the service network. Consider the definition of N_x as the number of requests[6] that traverse point $X \in L$ under MTR. In steady state, N_x converges to a random variable N provided $\lambda < c\mu$. Let π_k denote $\Pr[N = k]$ with $k \geq 0$.

Following the procedure in Section 4.2.1 of [20], we obtain the steady state probability vector $\pi = [\pi_i, i \geq 0]$. In the service network, request distance corresponds to the sojourn time in the bulk service M/M/1 queue. By applying Little's formula, we obtain the following expression for the expected request distance

$$\mathbb{E}[D] = \frac{r_0}{\lambda(1 - r_0)}, \tag{4}$$

where r_0 is the only root in the interval $(0, 1)$ of the following equation (with r as the variable)

$$\mu r^{c+1} - (\lambda + \mu)r + \lambda = 0. \tag{5}$$

[5] It is well known in queueing theory that among all service disciplines the variance of the waiting time is minimized under FCFS policy for Poisson arrivals and exponential service times [13]. In Section 5 we show that MTR maps to a temporal FCFS queue.

[6] We drop the superscript (M) for brevity.

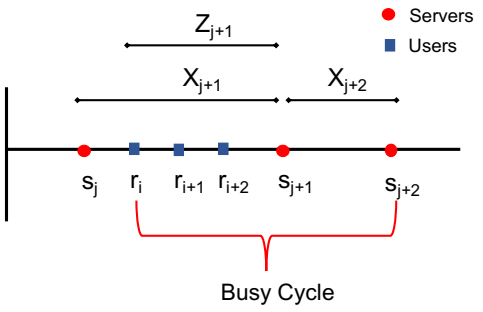


Figure 2: Allocation of users to servers under MTR policy.

5.1.1. When server capacity: $c = 1$

When $c = 1$, $r_0 = \rho$ is a solution of (5). Thus we can evaluate the expected request distance as

$$\mathbb{E}[D] = \frac{\rho}{\lambda(1-\rho)} = \frac{1}{\mu - \lambda}. \tag{6}$$

Note that, when server capacity is one, the service network can be modeled as an M/M/1 queue. In such a case, (6) is the mean sojourn time for an M/M/1 queue.

5.2. UGS

When both users and servers are Poisson distributed and servers have unit capacity, the request distance in UGS has the same distribution as the busy cycle in the corresponding Last-Come-First-Served Preemptive-Resume (*LCFS-PR*) queue having the density function [1]

$$f_{D^U}(x) = \frac{1}{x\sqrt{\rho}} e^{(\lambda+\mu)x} I_1(2x\sqrt{\lambda\mu}),\ x > 0, \tag{7}$$

where $\rho = \lambda/\mu$ and I_1 is the modified Bessel function of the first kind. Thus the expected request distance is equivalent to the average busy cycle duration in a LCFS-PR queue given by $1/(\mu - \lambda)$ [1].

When servers have capacities $c > 1$ it is difficult to characterize the expected request distance explicitly. However, by Theorem 1, the expected request distance under UGS is the same as that of MTR given by (4).

6. Unidirectional General Matching

We now derive expressions for the expected request distance when either users or servers are distributed according to a Poisson process and the other by renewal process.

6.1. Notion of exceptional service and accessible batches

We discuss the notion of exceptional service and accessible batches applicable to our service network as follows. Consider a service network with $c = 2$ as shown in Figure 2. Consider a user r_i. Let s_j be the server immediately to the left of r_i. We assume all users prior to r_i have already been allocated to servers $\{s_k, 1 \le k \le j\}$. MTR allocates both r_i and r_{i+1} to s_{j+1} and allocates r_{i+2} to s_{j+2}. We denote $[r_i, s_{j+2}]$ as a busy cycle of the service network. We have the following queueing theory analogy.

User r_i can be thought of as the first customer in a queueing system that initiates a busy period while r_{i+1} sees the system busy when it arrives. Because only r_i is in service at the arrival of r_{i+1}, r_{i+1} enters service with r_i and the two customers form a batch of size 2. and depart at time s_{j+1}. This is an example of an *accessible batch* [10]. An accessible batch admits subsequent arrivals, while the service is on, until the server capacity c is reached.

The service time for the batch, r_i, r_{i+1}, is described by the random variable Z_{j+1} which is different or *exceptional* when compared to service times of successive batches such as the one consisting of r_{i+2}. The service time for the second batch is X_{j+2}. Note that, Z_{j+1} only depends on X_{j+2} and Y_{i+2}. Thus when either X_{j+2} or Y_{i+2} is described by a Poisson process and the other by renewal process, Z_{j+1} converges to a random variable Z under steady state conditions. Denote $F_Z(x)$ and $f_Z(x)$ as the distribution and density functions for the random variable Z. Thus the

service network can be mapped to an exceptional service with accessible batches queueing (*ESABQ*) model. We formally define ESABQ as follows.

ESABQ: *Consider a queueing system where customers are served in batches of maximum size c. A customer entering the queue and finding fewer than c customers in the system joins the current batch and enters service at once, otherwise it joins a queue. After a batch departs leaving k customers in the buffer,* $\min(c, k)$ *customers form a batch and enter service immediately. There are two different service times cdfs,* $F_Z(x)$ *(exceptional batch) with mean* $\alpha_Z = 1/\mu_Z$ *and* $F_X(x)$ *(ordinary batch) with mean* $\alpha_X = 1/\mu$*. A batch is exceptional if its oldest customer entered an empty system, otherwise it is a regular batch. When the service time expires, all customers in the server depart at once, regardless of the nature of the batch (exceptional or regular).*

6.1.1. Evaluation of the distribution function: $F_Z(x)$

In this Section, we compute explicit expressions for the distribution function $F_Z(x)$ applicable to our service network.

When $F_X(x) \sim$ Expo(μ): In this case, we invoke the memoryless property of the exponential distribution F_X. Thus the exceptional distribution, F_Z, is

$$F_Z(x) = F_X(x) = 1 - e^{-\mu x}, x \geq 0. \tag{8}$$

When $F_Y(x) \sim$ Expo(λ): Using the memoryless property of F_Y, F_Z can be computed as

$$\begin{aligned} F_Z(x) &= \Pr(X - Y < x | Y < X) = \Pr(X - Y < x | X - Y > 0) = \frac{\Pr(X - Y < x) - \Pr(X - Y < 0)}{1 - \Pr(X - Y < 0)} \\ &= \frac{D_{XY}(x) - D_{XY}(0)}{1 - D_{XY}(0)}, \ x \geq 0, \end{aligned} \tag{9}$$

where $D_{XY}(x)$ is the distribution of the random variable $X - Y$ (also known as difference distribution). $D_{XY}(x)$ can be expressed as

$$\begin{aligned} D_{XY}(x) &= \Pr(X - Y \leq x) = \int_0^\infty \Pr(X - y \leq x) \Pr(Y = y) dy = \int_0^\infty F_X(x + y) \lambda e^{-\lambda y} dy = \int_x^\infty F_X(z) \lambda e^{-\lambda(z-x)} dz \\ &= \lambda e^{\lambda x} \left[\int_0^\infty F_X(z) e^{-\lambda z} dz - \int_0^x F_X(z) e^{-\lambda z} dz \right] = \lambda e^{\lambda x} \left[\mathcal{A}(F_X) - \mathcal{B}(x) \right], \end{aligned} \tag{10}$$

where $\mathcal{A}$ is the Laplace Transform operator on the function F_X and $\mathcal{B}(x)$ is denoted by

$$\mathcal{B}(x) = \int_0^x F_X(z) e^{-\lambda z} dz$$

Clearly $\mathcal{B}(0) = 0$. Thus combining (9) and (10) yields

$$F_Z(x) = \frac{\lambda e^{\lambda x} \left[\mathcal{A}(F_X) - \mathcal{B}(x) \right] - \lambda \mathcal{A}(F_X)}{1 - \lambda \mathcal{A}(F_X)}, \tag{11}$$

$$f_Z(x) = \frac{\lambda^2 e^{\lambda x} \left[\mathcal{A}(F_X) - \mathcal{B}(x) \right] - \lambda F_X(x)}{1 - \lambda \mathcal{A}(F_X)}, \tag{12}$$

$$\alpha_Z = \int_0^\infty x f_Z(x) dx, \ \sigma_Z^2 = \left[\int_0^\infty x^2 f_Z(x) dx \right] - \alpha_Z^2. \tag{13}$$

Expressions for $\mathcal{B}(x)$ are presented in Table 1. We can evaluate $\mathcal{A}(F_X)$ by setting $\mathcal{A}(F_X) = B(\infty)$. Detailed derivations are relegated to Appendix 13.1.

6.2. General requests and Poisson distributed servers (*GRPS*)

From our discussion in Section 6.1.1, it is clear that when servers are distributed according to a Poisson process, the exceptional service time distribution equals the regular batch service time distribution. In such a case we have the following queueing model.

Under GRPS, inter-arrival times and batch service times are, respectively, arbitrarily and exponentially distributed. Before initiating a service, a server finds the system in any of the following conditions. (i) $1 \le n \le c-1$ and (ii) $n \ge c$. Here n is the number of customers in the waiting buffer. For case (i) the server provides service to all n customers and admits subsequent arrivals until c is reached. For case (ii) the server takes c customers with no admission for subsequent customers arriving within its service time.

In such a case ESABQ can directly be modeled as a special case of a renewal input bulk service queue with accessible and non-accessible batches proposed in [10] with parameter values $a = 1$ and $d = b = c$. Let N_s and N_q denote random variables for numbers of customers in the system and in the waiting buffer respectively for ESABQ under GRPS. We borrow the following definitions from [10].

$$P_{n,0} = \Pr[N_s = n]; 0 \le n \le c-1,\ P_{n,1} = \Pr[N_q = n]; n \ge 0. \tag{14}$$

Using results from [10] we obtain the following expressions for equilibrium queue length probabilities.

$$P_{0,1} = \frac{C}{\mu}\left[\frac{r_0^{c-1} - r_0^c}{1 - r_0^c} + \frac{1}{r_0} - 1\right],\ P_{n,1} = \frac{C r_0^{n-1}(1 - r_0)}{\mu(1 - r_0^c)}; n \ge 1, \tag{15}$$

where $0 < r_0 < 1$ is the real root of the equation $r = F_Y^*(\mu - \mu r^c)$ and C is the normalization constant[7] given by

$$C = \lambda\left[\frac{1-\omega^c}{1-\omega} + \frac{1}{1-r_0} - \frac{\omega(r_0 - F_Y^*(\mu))}{r_0^c(1 - r_0\omega)}\left(\frac{1-r_0^c}{1-r_0} - r_0^{c-1}\frac{1-w^c}{1-w}\right)\right]^{-1}, \tag{16}$$

with $\omega = 1/F_Y^*(\mu)$. We then derive the expected queue length as

$$\mathbb{E}[N_q] = \sum_{n=0}^{\infty} n P_{n,1} = \sum_{n=1}^{\infty} n \frac{C r_0^{n-1}(1 - r_0)}{\mu(1 - r_0^c)} = \frac{C(1 - r_0)}{\mu(1 - r_0^c)} \sum_{n=1}^{\infty} n r_0^{n-1} = \frac{C}{\mu(1 - r_0^c)(1 - r_0)}. \tag{17}$$

Applying Little's law and considering the analogy between our service network and ESABQ we obtain the following expression for the expected request distance.

$$\mathbb{E}[D] = \frac{C}{\lambda\mu(1 - r_0^c)(1 - r_0)} + \frac{1}{\mu}. \tag{18}$$

6.3. Poisson distributed requests and general distributed servers (*PRGS*)

As discussed in Section 6.1.1, if servers are placed on a 1-d line according to a renewal process with requests being Poisson distributed, the service time distribution for the first batch in a busy period differs from those of subsequent batches. Below we derive expressions for queue length distribution and expected request distance for ESABQ under PRGS.

6.3.1. Queue length distribution

We use a supplementary variable technique to derive the queue length distribution for ESABQ under PRGS as follows.

Let $L(t)$ be the number of customers at time $t \ge 0$, $R(t)$ the residual service time at time $t \ge 0$ (with $R(t) = 0$ if $L(t) = 0$), and $I(t)$ the type of service at time $t \ge 0$ with $I(t) = 1$ (resp. $I(t) = 2$) if exceptional (resp. ordinary) service time.

Let us write the Chapman-Kolmorogov equations for the Markov chain $\{(L(t), R(t), I(t)),\ t \ge 0\}$.

For $t \ge 0$, $n \ge 1$, $x > 0$, $i = 1, 2$ define

$$p_t(n, x; i) = \mathbb{P}(L(t) = n, R(t) < x, I(t) = i) \quad \text{and} \quad p_t(0) = \mathbb{P}(L(t) = 0).$$

[7] The normalization constant C derived in [10] is incorrect. The correct constant for our case is given in (16).

Also, define for $x > 0$, $i = 1, 2$,

$$p(n, x; i) = \lim_{t\to\infty} p_t(n, x; i) \quad \text{and} \quad p(0) = \lim_{t\to\infty} p_t(0).$$

By analogy with the analysis for the M/G/1 queue we get

$$\frac{\partial}{\partial t}p_t(0) = -\lambda p_t(0) + \sum_{k=1}^{c}\frac{\partial}{\partial x}p_t(k, 0; 1) + \sum_{k=1}^{c}\frac{\partial}{\partial x}p_t(k, 0; 2),$$

so that, by letting $t \to \infty$,

$$\lambda p(0) = \sum_{k=1}^{c}\left(\frac{\partial}{\partial x}p(k, 0; 1) + \frac{\partial}{\partial x}p(k, 0; 2)\right). \tag{19}$$

With further simplification (See Appendix 13.2.1), for $n \geq 1$, $x > 0$ we get

$$\frac{\partial}{\partial x}g(n, x) - \lambda g(n, x) - \frac{\partial}{\partial x}g(n, 0) + \lambda g(n-1, x)\mathbf{1}(n \geq 2) + \lambda p(0)F_Z(x)\mathbf{1}(n = 1) + F_X(x)\frac{\partial}{\partial x}g(n + c, 0) = 0, \tag{20}$$

where $g(n, x) = p(n, x; 1) + p(n, x; 2)$ for $n \geq 1$, $x > 0$. Introduce

$$G(z, s) := \sum_{n\geq 1} z^n \int_0^\infty e^{-sx}g(n, x)dx \quad \forall |z| \leq 1,\ s \geq 0.$$

Denote by $F_Z^*(s) = \int_0^\infty e^{-sx}dF_Z(x)$ the LST of F_Z for $s \geq 0$. Note that

$$\int_0^\infty e^{-sx}F_{Z\text{or}X}(x)dx = \frac{F_{Z\text{or}X}^*(s)}{s}, \quad \forall s > 0.$$

Multiplying both sides of (20) by $z^n e^{-sx}$, integrating over $x \in [0, \infty)$ and summing over all $n \geq 1$, yields

$$s\,(\lambda(1 - z) - s)\,G(z, s) = \lambda z p(0)F_Z^*(s) - \sum_{n\geq 1} z^n \frac{\partial}{\partial x}g(n, 0) + F_X^*(s)\sum_{n\geq 1} z^n \frac{\partial}{\partial x}g(n + c, 0)) \tag{21}$$

where $\lambda p(0) = \sum_{k=1}^{c}\frac{\partial}{\partial x}g(k, 0)$ from (19). We have

$$\frac{1}{z^c}\sum_{n\geq 1} z^{n+c}\frac{\partial}{\partial x}g(n + c, 0)) = \frac{1}{z^c}\sum_{n\geq 1} z^n \frac{\partial}{\partial x}g(n, 0) - \frac{1}{z^c}H(z) \tag{22}$$

where $H(z) = \sum_{k=1}^{c} z^k a_k$ with $a_k := \frac{\partial}{\partial x}g(k, 0)$, for $k = 1, \ldots, c$. Introducing the above into (21) gives

$$s\,(\lambda(1 - z) - s)\,G(z, s) = \left(\frac{F_X^*(s)}{z^c} - 1\right)\Psi(z) - F_X^*(s)\frac{H(z)}{z^c} + \lambda z p(0)F_Z^*(s) \tag{23}$$

where $\Psi(z) := \sum_{n\geq 1} z^n \frac{\partial}{\partial x}g(n, 0)$. Since $G(z, s)$ is well-defined for $|z| \leq 1$ and $s \geq 0$, the r.h.s. of (23) must vanish when $s = \lambda(1 - z)$. This gives the relation

$$\Psi(z) = \frac{z^c}{z^c - F_X^*(\theta(z))}\left[-F_X^*(\theta(z))\frac{H(z)}{z^c} + \lambda z p(0)F_Z^*(\theta(z))\right]$$

with $\theta(z) = \lambda(1 - z)$ and $|z| \leq 1$. Introducing the above in (23) gives

$$s\,(\lambda(1 - z) - s)\,G(z, s) = -F_X^*(s)\frac{H(z)}{z^c} + \lambda z p(0)F_Z^*(s) + \frac{F_X^*(s) - z^c}{z^c - F_X^*(\theta(z))}\left[\lambda z p(0)F_Z^*(\theta(z)) - F_X^*(\theta(z))\frac{H(z)}{z^c}\right]. \tag{24}$$

Let $N(z)$ be the z-transform of the stationary number of customers in the system. Integrating by part, we get for $n \geq 1$,

$$s \int_0^\infty e^{-sx} g(n,x)dx = \int_0^\infty e^{-sx} dg(n,x),$$

so that

$$\lim_{s\to\infty} s \int_0^\infty e^{-sx} g(n,x)dx = \lim_{s\to 0} \int_0^\infty e^{-sx} dg(n,x) = \int_0^\infty dg(n,x) = g(n,\infty), \tag{25}$$

where the interchange between the limit and the integral sign is justified by the bounded convergence theorem. Therefore,

$$\begin{aligned} N(z) &= \sum_{n\geq 1} z^n g(n,\infty) + p(0) \\ &= \sum_{n\geq 1} z^n \lim_{s\to\infty} s \int_0^\infty e^{-sx} g(n,x)dx \quad \text{from (25)} \\ &= \lim_{s\to 0} sG(z,s) + p(0), \end{aligned} \tag{26}$$

where the interchange between the summation over n and the integral sign is again justified by the bounded convergence theorem. Letting now $s \to 0$ in (24) and using (26), gives

$$\theta(z)N(z) = \frac{1-z^c}{z^c - F_X^*(\theta(z))} \left[-F_X^*(\theta(z)) \frac{H(z)}{z^c} + \lambda z p(0) F_Z^*(\theta(z)) \right] - \frac{H(z)}{z^c} + \lambda p(0). \tag{27}$$

By noting that $\lambda p(0) = \sum_{k=1}^c a_k$ (cf. (19)), Eq. (27) can be rewritten as

$$N(z) = \frac{1}{\theta(z)} \left(\frac{z(1-z^c)}{z^c - F_X^*(\theta(z))} \sum_{k=1}^c a_k \left[F_Z^*(\theta(z)) - z^{k-c-1} F_X^*(\theta(z)) \right] + \sum_{k=1}^c a_k (1 - z^{k-c}) \right). \tag{28}$$

The r.h.s. of (28) contains c unknown constants $a_1, \ldots, a_c$ yet to be determined. Define $A(z) = F_X^*(\theta(z))$. It can be shown that $z^c - A(z)$ has $c-1$ zeros inside and one on the unit circle, $|z| = 1$ (See Appendix 13.2.3). Denote by $\xi_1, \ldots, \xi_q$ the $1 \leq q \leq c$ distinct zeros of $z^c - A(z)$ in $\{|z| \leq 1\}$, with multiplicity $n_1, \ldots, n_q$, respectively, with $n_1 + \cdots + n_q = c$. Hence,

$$z^c - F_X^*(k(z)) = \gamma \prod_{i=1}^q (z - \xi_i)^{n_i}.$$

Since $z^c - A(z)$ vanishes when $z = 1$ and that $\frac{d}{dz}(z^c - A(z))|_{z=1} = c - \rho > 0$, we conclude that $z^c - A(z)$ has one zero of multiplicity one at $z = 1$.

Without loss of generality assume that $\xi_q = 1$ and let us now focus on the zeros $\xi_1, \ldots, \xi_{q-1}$. When $z = \xi_i$, $i = 1, \ldots, q-1$, the term $F_Z^*(\theta(z)) - z^{k-c-1} F_X^*(\theta(z))$ in (28) must have a zero of multiplicity (at least) n_i since $N(\xi_i)$ is well defined. This gives $c-1$ linear equations to be satisfied by $\xi_1, \ldots, \xi_q$. In the particular case where all zeros have multiplicity one (see Appendix 13.2.2), namely $q = c$, these $c-1$ equations are

$$\sum_{k=1}^c a_k \left[F_Z^*(\theta(\xi_i)) - \xi_i^{k-c-1} F_X^*(\theta(\xi_i)) \right] = 0, \ i = 1, \ldots, c-1. \tag{29}$$

With $U(z) := F_Z^*(\theta(z))/F_X^*(\theta(z))$ (29) is equivalent to

$$\sum_{k=1}^c a_k \left[U(\xi_i) - \xi_i^{k-c-1} \right] = 0, \ i = 1, \ldots, c-1, \tag{30}$$

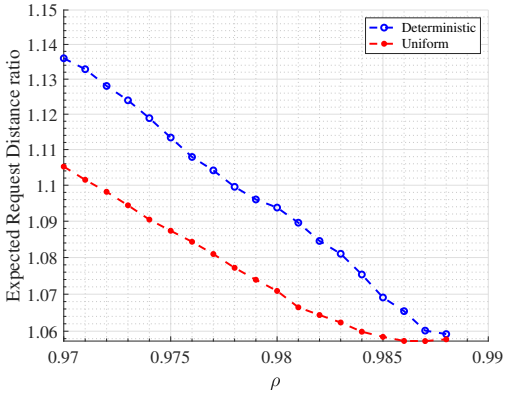


Figure 3: The plot shows the ratio $\mathbb{E}[D]/\overline{D}_s$ for deterministic and uniform inter-server distance distributions.

since $F_X^*(\theta(\xi_i)) \neq 0$ for $i = 1, \ldots, c-1$ ($F_X^*(\theta(\xi_i)) = 0$ implies that ξ_i=0 which contradicts that ξ_i a zero of $z^c - F_X^*(\theta(z))$ since $F_X^*(\theta(0)) = F_X^*(\lambda) > 0$). Eq. (28) can be rewritten as

$$N(z) = \frac{\sum_{k=1}^{c} a_k \left[z^c - z^k + z(1-z^c)F_Z^*(\theta(z)) - (1-z^k)F_X^*(\theta(z))\right]}{\theta(z)(z^c - F_X^*(\theta(z))}. \tag{31}$$

A c-th equation is provided by the normalizing condition $N(z) = 1$. Since the numerator and denominator in (31) have a zero of order 2 at $z = 1$, differentiating twice the numerator and the denominator w.r.t z and letting $z = 1$ gives

$$\sum_{k=1}^{c} a_k(c(1+\rho_z) - \rho k) = \lambda(c-\rho), \tag{32}$$

where $\rho_z = \lambda\alpha_Z$. We consider few special cases of the model in Appendix 13.2.4 and verify with the expressions of queue length distribution available in the literature.

6.3.2. Expected request distance

From (31) the expected queue length is

$$\begin{aligned}\overline{N} &= \frac{d}{dz}N(z)\Big|_{z=1} \\ &= \frac{1}{2\lambda(c-\rho)^2}\sum_{k=1}^{c} a_k\left[\lambda^2\sigma_Z^{(2)}c(c-\rho) + \lambda^2\sigma_X^{(2)}c(1+\rho_z-k) + (ck(c-k) + k(k-1)\rho - c(c-1))\rho + 2c^2\rho_z - c(c+1)\rho_z\rho\right],\end{aligned} \tag{33}$$

where $\sigma_Z^{(2)}$ and $\sigma_X^{(2)}$ are the second order moments of distributions F_Z and F_X respectively. Again by applying Little's law and considering the analogy between our service network with ESABQ we get the following expression for the expected request distance.

$$\mathbb{E}[D] = \overline{N}/\lambda. \tag{34}$$

7. Discussion of Unidirectional Allocation Policies

In this section we describe generalizations of models and results for unidirectional allocation policies. We first consider the case when inter-user and inter-server distances both have general distributions.

7.1. Heavy traffic limit for general request and server spatial distributions

Consider the case when the inter-user and inter-server distances each are described by general distributions. We assume server capacity, $c = 1$. As $\rho \to 1$, we conjecture that the behavior of MTR approaches that of the G/G/1 queue. One argument in favor of our conjecture is the following. As $\rho \to 1$, the busy cycle duration tends to infinity. Consequently, the impact of the exceptional service for the first customer of the busy period on all other customers diminishes to zero as there is an unbounded increasing number of customers served in the busy period.

It is known that in heavy traffic waiting times in a G/G/1 queue are exponential distributed and the mean sojourn time is given by $\alpha_X + [(\sigma_X^2 + \sigma_Y^2)/2\alpha_Y(1-\rho)]$ [20]. We expect the expected request distance to exhibit similar behavior. Thus we have the following conjecture.

Conjecture 1. *At heavy traffic i.e. as $\rho \to 1$, the expected request distance for the G/G/1 spatial system with $c = 1$ is given by*

$$\mathbb{E}[D] = \alpha_X + \frac{\sigma_X^2 + \sigma_Y^2}{2\alpha_Y(1-\rho)}. \tag{35}$$

Denote by $\overline{D}_s$ the average request distance as obtained from simulation. We plot the ratio $\mathbb{E}[D]/\overline{D}_s$ across various inter-request and inter-server distance distributions in Figure 3. It is evident that as $\rho \to 1$, the ratio $\mathbb{E}[D]/\overline{D}_s$ converges to 1 across different inter-server distance distributions.

7.2. Heterogeneous server capacities under PRGS

We now proceed to analyze a setting where server capacity is a random variable. Assume server capacity C takes values from $\{1, 2, \ldots, c\}$ with distribution $\Pr(C = j) = p_j, \forall j \in \{1, 2, \ldots, c\}$, s.t. $\sum_{j=1}^{c} p_j = 1$ and $p_c > 0$. We also assume the stability condition $\rho < \overline{C}$ where $\overline{C}$ is the average server capacity. Denote H as the random variable associated with number of requests that traverse through a point just after a server location[8].

7.2.1. Distribution of H

Let V denote the number of new requests generated during a service period with $k_v = \Pr(V = v), \forall v \geq 0$. According to the law of total probability, it holds that

$$k_v = \int_0^\infty \Pr(V = v | X = \nu) f_X(\nu) = \frac{1}{v!} \int_0^\infty e^{-\lambda\nu} (\lambda\nu)^v dF_X(\nu). \tag{36}$$

Then the corresponding generating function $K(z)$ is denoted by

$$K(z) = \sum_{v=0}^{\infty} k_v z^v = F_X^*(\lambda(1-z)). \tag{37}$$

We now consider an embedded Markov chain generated by H. Denote the corresponding transition matrix as M. Then we have

$$M_{m,l} = \begin{cases} \sum_{i=0}^{c-m} k_i P_{i+m}, & 0 \leq m \leq c, l = 0; \\ \sum_{i=0}^{c} k_{i+l-m} p_i, & 0 \leq m \leq l, l \neq 0; \\ \sum_{i=m-l}^{c} k_{i+l-m} p_i, & l+1 \leq m \leq c+l, l \neq 0; \\ 0, & o.w., \end{cases} \tag{38}$$

where $P_i = \sum_{j=i}^{c} p_j$ and $p_0 = 0$. Let $\pi = [\pi_j, j \geq 0]$ and $N(z) = \sum_{j \geq 0} \pi_j z^j$ denote the steady state distribution and its z-transform respectively. π is obtained out by solving

$$\pi_l = \sum_{m=0}^{\infty} \pi_m M_{m,l}, l = 0, 1, \ldots. \tag{39}$$

Thus we have for $l \in \mathbb{N}$,

$$\pi_0 = \sum_{m=0}^{c} \pi_m \sum_{i=0}^{c-m} k_i P_{i+m}, \ \pi_l = \sum_{m=0}^{l} \pi_m \sum_{i=0}^{c} k_{i+l-m} p_i + \sum_{m=l+1}^{c+l} \pi_m \sum_{i=m-l}^{c} k_{i+l-m} p_i. \tag{40}$$

[8] An analysis for the distribution of number of requests that traverse through any random location would involve the notions of exceptional service and accessible batches.

Multiplying by z^l and summing over l gives

$$N(z) = E_\pi + v_1(z) + v_2(z) \tag{41}$$

$$E_\pi = \pi_0 \sum_{i=0}^{c-1} k_i P_{i+1} + \sum_{m=1}^{c-1} \pi_m \sum_{i=m}^{c-1} k_{i-m} P_{i+1} \tag{42}$$

$$v_1(z) = \sum_{l=0}^{\infty} z^l \sum_{m=0}^{l} \pi_m \sum_{i=0}^{c} k_{i+l-m} p_i \tag{43}$$

$$v_2(z) = \sum_{l=0}^{\infty} z^l \sum_{m=l+1}^{c+l} \pi_m \sum_{i=m-l}^{c} k_{i+l-m} p_i. \tag{44}$$

The expressions for $v_1(z)$ and $v_2(z)$ can be further simplified (see Appendix 13.3) to

$$v_1(z) = N(z)\left\{ \sum_{i=0}^{c} p_i z^{-i} \left[K(z) - \sum_{j=0}^{i} k_j z^j \right] + \sum_{i=0}^{c} k_i z^i \right\} \tag{45}$$

$$v_2(z) = \left[\sum_{m=0}^{c} z^{-m} \sum_{i=m}^{c} k_{i-m} p_i \left\{ N(z) - \sum_{j=0}^{m-1} \pi_j z^j \right\} \right] - N(z) \sum_{i=0}^{c} k_i z^i. \tag{46}$$

Combining (41), (45) and (46) yields

$$N(z) = E_\pi + N(z)\left\{ K(z) \sum_{i=0}^{c} p_i z^{-i} \right\} - \sum_{j=0}^{c-1} \pi_j \sum_{m=1}^{c-j} z^{-m} \sum_{i=m+j}^{c} k_{i-(m+j)} p_i. \tag{47}$$

Thus we obtain

$$N(z) = \frac{E_\pi - \sum_{j=0}^{c-1} \pi_j \sum_{m=1}^{c-j} z^{-m} \sum_{i=m+j}^{c} k_{i-(m+j)} p_i}{1 - K(z) \sum_{i=0}^{c} p_i z^{-i}}. \tag{48}$$

Multipying numerator and denominator by z^c yields

$$N(z) = \frac{z^c E_\pi - \sum_{j=0}^{c-1} \pi_j \sum_{m=1}^{c-j} z^{c-m} \sum_{i=m+j}^{c} k_{i-(m+j)} p_i}{z^c - K(z) \sum_{i=0}^{c} p_{c-i} z^i}. \tag{49}$$

To determine $N(z)$, we need to obtain the probabilities $\pi_i, 0 \le i \le c-1$. It can be shown that the denominator of (49) has $c-1$ zeros inside and one on the unit circle, $|z| = 1$ (See Appendix 13.3.2). As $N(z)$ is analytic within and on the unit circle, the numerator must vanish at these zeros, giving rise to c equations in c unknowns.

Let $\xi_q : 1 \le q \le c$ be the zeros of $z^c - K(z) \sum_{i=0}^{c} p_{c-i} z^i$ in $\{|z| \le 1\}$. W.l.o.g let $\xi_c = 1$. We have the following $c-1$ equations.

$$E_\pi - \sum_{j=0}^{c-1} \pi_j \sum_{m=1}^{c-j} \xi_q^{-m} \sum_{i=m+j}^{c} k_{i-(m+j)} p_i = 0, \ i = 1, \dots, c-1, \tag{50}$$

A c-th equation is provided by the normalizing condition $\lim_{z\to 1} N(z) = 1$. In the particular case where all zeros have multiplicity one, it can be shown that these c equations are linearly independent[9]. Once the parameters $\{\pi_i, 0 \le i \le c-1\}$ are known, $\mathbb{E}[H]$ can be expressed as

$$\mathbb{E}[H] = \overline{H} = \lim_{z\to 1} N'(z). \tag{51}$$

7.2.2. Expected Request Distance

To evaluate the expected request distance we adopt arguments from [6]. Consider any interval of length ν between two consecutive servers. There are on average $\overline{H}$ requests at the beginning of the interval , each of which must travel ν distance. New users are spread randomly over the interval and there are on an average $\lambda\nu$ new users. The request made by each new user must travel on average $\nu/2$. Thus we have

$$\mathbb{E}[D] = \frac{1}{\rho} \int_0^\infty (\overline{H}\nu + \frac{1}{2}\lambda\nu^2) d F_X(\nu) = \frac{1}{\rho}\left[\frac{\overline{H}}{\mu} + \frac{\lambda}{2}\left(\sigma_X^2 + \frac{1}{\mu^2} \right) \right]. \tag{52}$$

[9] For all cases evaluated across uniform, deterministic and hyperexponential distributions we found the set of c equations to be linearly independent.

7.3. Uncapacitated request allocation

An interesting special case of the unidirectional general matching is the uncapacitated scenario. Consider the case where servers do not have any capacity constraints, i.e. $c = \infty$. In such a case, all users are assigned to the nearest server to their right.

GRPS: When $c \to \infty$ and given $0 < r_0 < 1$, $r_0 = F_Y^*(\mu - \mu r_0^c) = F_Y^*(\mu)$. Setting $\omega = 1/F_Y^*(\mu) = 1/r_0$ in (16) and simplifying yields

$$C \to 0, \text{ as } c \to \infty, \implies \mathbb{E}[D] \to \frac{1}{\mu} \text{ as } c \to \infty. \tag{53}$$

PRGS: Under PRGS, when $c \to \infty$ there exists no request allocated to a server other than the nearest server to its right. Again using Bailey's method as in [6] and setting $\overline{H} = 0$ in (52) we get

$$\mathbb{E}[D] \to \frac{\mu}{2}\left(\sigma_X^2 + \frac{1}{\mu^2}\right) \text{ as } c \to \infty. \tag{54}$$

7.4. Cost models

Consider the following generalization of the service network. We define cost of an allocation as the communication cost associated with an allocated request-server pair. Consider communication cost as a function $\mathcal{T}$ of the request distances. Then the expected communication cost across the service network is given as

$$\overline{T} = E[\texttt{cost}] = \int_{d=0}^{\infty} \mathcal{T}(d) dW(d), \tag{55}$$

where W is the request distance distribution. One such cost model widely used in wireless ad hoc networks is [8]

$$\mathcal{T}(d) = t_0 d^{\beta}, \tag{56}$$

where β is the path loss exponent typically $2 \leq \beta \leq 4$ and t_0 is a constant. Below we derive the expected communication cost for the scenario when $c = 1$.

7.4.1. GRPS with $c = 1$

In this case the service network directly maps to a temporal G/M/1 queue. Thus W can be expressed as the sojourn time distribution of the corresponding G/M/1 queue. Hence $W \sim \text{Expo}(\mu(1 - r_0))$ with r_0 as defined in Section 6.2. We have

$$\overline{T} = \int_{d=0}^{\infty} t_0 d^{\beta} dW(d) = \frac{t_0}{\mu^{\beta}(1 - r_0)^{\beta}} \Gamma(\beta + 1), \tag{57}$$

where $\Gamma(x) = \int_0^{\infty} y^{x-1} e^{-y} dy$ is the gamma function.

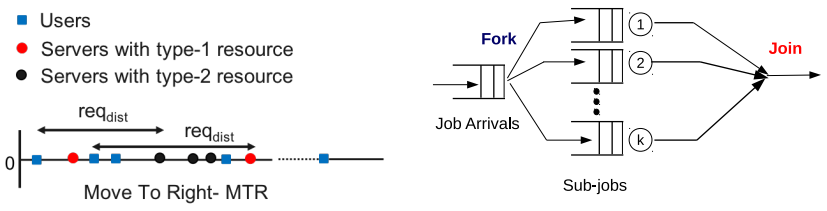
7.4.2. PRGS with $c = 1$

In this case, the service network can be modeled as a temporal M/G/1 queue with first customer having exceptional service [22]. Denote $W^*(s)$ as the LS transform of W. Using results from [22]

$$W^*(s) = \frac{(1 - \rho)\left\{\lambda\left[F_Z^*(s) - F_X^*(s)\right] - sF_Z^*(s)\right\}}{(1 - \rho + \rho_Z)\left[\lambda - s - \lambda F_X^*(s)\right]}. \tag{58}$$

When β is an integer,

$$\overline{T} = t_0(-1)^{\beta} \frac{d^{(\beta)}}{ds} W^*(s)|_{s=0}, \tag{59}$$



(a) (b)

Figure 4: Two resource scenario with $c = 1$ (a) Depiction of request distances and (b) Mapping to Fork-join queues.

7.5. Extension to two resources

Now consider the following scenario where each user requests two resources which reside on different servers as shown in Figure 4(a). Let the corresponding servers be distributed according to a Poisson process with densities μ_1 and μ_2. Let the users be distributed according to a Poisson process. The service network, in this case, can be modeled as a fork-join queueing system as shown in Figure 4(b) [16]. In such a queue, each incoming job is split into two sub-jobs each of which is served on one of the two servers. After service, each sub-job waits until the other sub-job has been processed. They then merge and leave the system. In the service network as well, each request forks two sub-requests one for each resource type. A request is said to be completed only if it has retrieved both the resources, thus mapping it to a fork-join queue. We define the *overall request distance* to be the maximum value among the request distances across all resource types and denote it as the random variable D_{max}.

7.5.1. Identical service rates ($\mu_1 = \mu_2 = \mu$ and $c = 1$)

The approximated expected request distance for this scenario is obtained from the expression for the expected sojourn time of a fork join queue with homogeneous servers as [16]:

$$\mathbb{E}[D_{max}] = \frac{12\mu - \lambda}{8\mu(\mu - \lambda)}, \tag{60}$$

Note that, the corresponding expected request distance in case of single resource is given by Equation (6) $\mathbb{E}[D] = 1/(\mu - \lambda)$. Clearly,

$$\mathbb{E}[D_{max}] = \frac{12\mu - \lambda}{8\mu(\mu - \lambda)} = [1.5 - 0.125\rho]\,\frac{1}{\mu - \lambda} > \frac{1}{\mu - \lambda} = \mathbb{E}[D], \tag{61}$$

Thus we have $\mathbb{E}[D_{max}] > \mathbb{E}[D]$.

8. Bidirectional Allocation Policies

Both UGS and MTR minimize expected request distance among all unidirectional policies. In this section we formulate the bi-directional allocation policy that minimizes expected request distance. Let $\eta : R \to S$ be any mapping of users to servers. Our objective is to find a mapping $\eta^* : R \to S$, that satisfies

$$\begin{aligned} \eta^* &= \arg\min_{\eta} \sum_{i \in R} d_{\mathcal{L}}(r_i, s_{\eta(i)}) \\ s.t. \quad & \sum_{i \in R} \mathbf{1}_{\eta(i)=j} \leq c, \forall j \in S \end{aligned} \tag{62}$$

W.l.o.g, let $r_1 \leq r_2 \leq \cdots \leq r_i \leq \cdots \leq r_{|R|}$ be locations of requests and $s_1 \leq s_2 \leq \cdots \leq s_i \leq \cdots \leq s_{|S|}$ be locations of servers. We first focus on the case when $c = 1$. We consider the following two scenarios.

Case 1: $|R| = |S|$

When $|R| = |S|$, an optimal allocation strategy is given by the following theorem [7].

Theorem 2. *When $|R| = |S|$, an optimal assignment is obtained by the policy: $\eta^*(i) = i,\ \forall i \in \{1, \cdots, |R|\}$ i.e. allocating the i^{th} request to the i^{th} server and the average request distance is given by*

$$\mathbb{E}[D] = \frac{1}{|R|} \sum_{i=1}^{|R|} |s(i) - r(i)|. \tag{63}$$

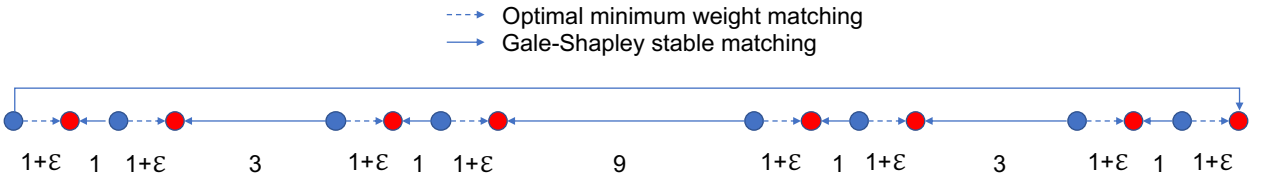


Figure 5: Worst case scenario for Gale-Shapley.

Case 2: $|R| < |S|$ This is the case where there are fewer requesters than servers. In this case, a Dynamic Programming (*DP*) based algorithm (Algorithm 1) obtains the optimal assignment.

Let $C[i, j]$ denote the optimal cost (i.e., sum of distances) of assigning the first i requests (counting from the left) located at $r_1 \leq r_2 \leq \ldots \leq r_i$ to the first j servers (also counting from the left) located at $s_1 \leq s_2 \leq \ldots \leq s_j$. If $j == i$, the optimal assignment is trivial due to Theorem 2 and $C[i, i]$ is computed easily for all $i \leq |R|$ by summing pairwise distances $d[1, 1], d[2, 2], \ldots, d[i, i]$ (Lines 6–7). For the base case, $i = 1, j > 1$, only the first user needs to be assigned to its nearest server (Lines 9–16). For the general dynamic programming step, consider $j > i$. Then $C[i, j]$ can be expressed in terms of the costs of two subproblems, i.e., $C[i - 1, j - 1]$ and $C[i, j - 1]$ (Lines 19–24). In the optimal solution, two cases are possible: either request i is assigned to server j, or the latter is left unallocated. The former case occurs if the first $i - 1$ requests are assigned to the first $j - 1$ servers at cost $C[i - 1, j - 1]$, and the latter case occurs when the first i requests are assigned to the first $j - 1$ servers at cost $C[i, j - 1]$. This is a consequence of the no-crossing lemma (Lemma 1). The optimal $C[i, j]$ is chosen depending on these two costs and the current distance $d[i, j]$.

Lemma 1. *In an optimal solution, η^*, to the problem of matching users at $r_1 \leq r_2 \leq \ldots \leq r_{|R|}$ to servers at $s_1 \leq s_2 \leq \ldots \leq s_{|S|}$, where $|S| \geq |R|$, there do not exist indices i, j such that $\eta^*(i) > \eta^*(i')$ when $i' > i$.*

Proof. See Appendix 13.4. □

The dynamic programming algorithm fills cells in an $|R| \times |S|$ matrix C whose origin is in the north-west corner. The lower triangular portion of this matrix is invalid since $|R| \leq |S|$. The base cases populate the diagonal and the northernmost row, and in the general DP step, the value of a cell depends on the previously computed values in the cells located to its immediate west and diagonally north-west. As an optimization, for a fixed i, the j-th loop index needs to run only from $i + 1$ through $i + |S| - |R|$ (Lines 11 and 18) instead of from $i + 1$ through $|S|$. This is because the first request has to be assigned to a server s_j with $j \leq |S| - |R| + 1$ so that the rest of the $|R| - 1$ requests have a chance of being placed on unique servers[10]. The optimal average request distance is given by $C[|R|, |S|]$.

The time complexity of the main DP step is $O(|R| \times (|S| - |R| + 1))$. Note that this assumes that the pairwise distance matrix d of dimension $|R| \times |S|$ has been precomputed. The optimization applied above can be similarly applied to this computation and hence the overall time complexity of Algorithm 1 is $O(|R| \times (|S| - |R| + 1))$. Therefore, if $|S| = O(|R|)$, the worst case time complexity is quadratic in $|R|$. However, if $|S| - |R|$ grows only sub-linearly with $|R|$, the time complexity is sub-quadratic in $|R|$.

Note that retrieving the optimal assignment requires more book-keeping. An $|R| \times |S|$ matrix A stores key intermediate steps in the assignment as the DP algorithm progresses (Lines 8, 16, 21, 24). The optimal assignment vector π can be retrieved from matrix A using procedure ReadOptAssignment.

Another bidirectional assignment scheme is the Gale-Shapley algorithm [9], which produces stable assignments, though in the worst case it can yield an assignment that is $O(|R|^{\ln 3/2}) \approx O(|R|^{0.58})$ times costlier than the optimal assignment yielded by Algorithm 1, where $|R|$ is the number of users [18]. The worst case scenario is illustrated in Figure 5, with $|R| = 2^{t-1}$, where t is the number of clusters of users and servers; and the largest distance between adjacent points is 3^{t-2}. However at low/moderate loads for the cases evaluated in Section 9, we find its performance to be not much worse than optimal.

[10]Note that in this exposition, we consider server capacity $c = 1$. If $c > 1$, we simply add c servers at each prescribed server location, and requests will still be placed on unique servers.

Algorithm 1 Optimal Assignment by Dynamic Programming

Input: $r_1 \leq \cdots \leq r_{|R|}$; $s_1 \leq \cdots \leq s_{|S|}$
Output: The optimal assignment π
procedure OPTDP(r, s)
    $d_{|R|\times|S|} = \text{COMPUTEPAIRWISEDISTANCES}(r, s)$
    $C = \{\infty\}_{|R|\times|S|}$
    for $i = 1, \cdots, |R|$ **do**
        $C[i, i] = \text{TRIVIALASSIGNMENT}(i, d)$
    $A[|R|, |R|] = |R|$
    $nearest = 0$
    $nearestcost = C[1, 1]$
    for $j = 2, \cdots, |S| - |R| + 1$ **do**
        if $d[1, j] < nearestcost$ **then**
            $nearestcost = d[1, j]$
            $nearest = j$
        $C[1, j] = nearestcost$
        $A[1, j] = nearest$
    for $i = 2, \cdots, |R|$ **do**
        for $j = i + 1, \cdots, i + |S| - |R|$ **do**
            if $C[i, j - 1] < d[i, j] + C[i - 1, j - 1]$ **then**
                $C[i, j] = C[i, j - 1]$
                $A[i, j] = A[i, j - 1]$
            else
                $C[i, j] = d[i, j] + C[i - 1, j - 1]$
                $A[i, j] = j$
    return $\text{READOPTASSIGNMENT}(A)$
procedure TRIVIALASSIGNMENT(n, d)
    $Cost = 0$
    for $i = 1, \cdots, n$ **do**
        $Cost = Cost + d[i, i]$
    return $Cost$
procedure READOPTASSIGNMENT(A)
    $|R|, |S| = \text{DIMENSIONS}(A)$
    $s = |S|$
    for $i = |R|, \cdots, 1$ **do**
        $\pi[i] = A[i, s]$
        $s = A[i, s] - 1$
    return π

9. Numerical Experiments

In this section, we examine the effect of various system parameters on expected request distance under MTR policy. We also compare the performance of various greedy allocation strategies along with the unidirectional policies to the optimal strategy.

9.1. Experimental setup

In our experiments, we consider a mean requester rate $\lambda \in (0, 1)$. We consider various inter-server distance distributions with density one. In particular, (i) for exponential distributions, the density is set to $\mu = 1$; (ii) for deterministic distributions, we assign parameter $d_0 = 1$. (iii) for second order hyper-exponential distribution (H_2), denote p_1 and p_2 as the phase probabilities. Let μ_1 and μ_2 be corresponding phase rates. We assume $p_1/\mu_1 = p_2/\mu_2$. We express H_2 parameters in terms of the squared coefficient of variation, c_v^2, and mean inter-server distance, α_X, i.e. we set

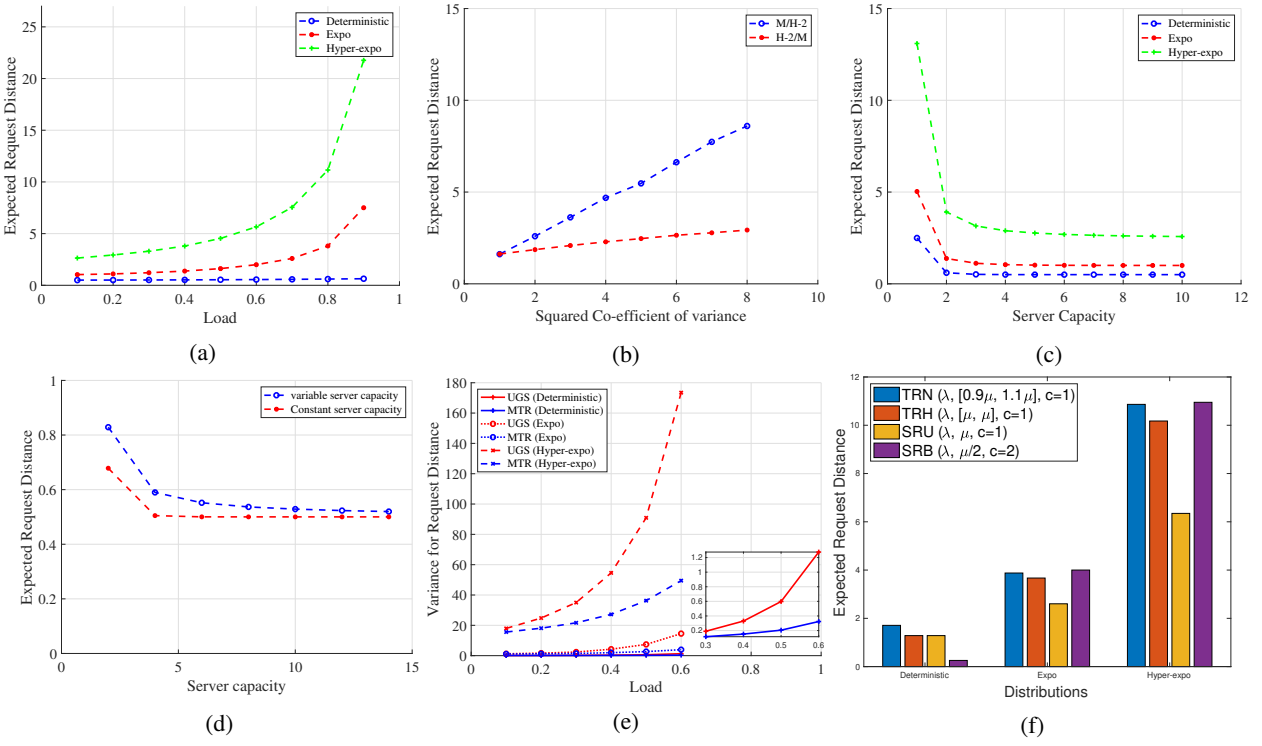


Figure 6: Sensitivity analysis of MTR/UGS policy. (a) Effect of load on expected request distance with $\boldsymbol{c=2}$. (b) Effect of squared coefficient of variation on expected request distance with $\boldsymbol{\lambda=\mu=1}$ and $\boldsymbol{c=2}$. (c) Effect of server capacity on expected request distance with $\boldsymbol{\rho=0.8}$. Effect of variability in server capacity on expected request distance for (d) Deterministic distribution with $\boldsymbol{\rho=0.8}$ (e) Effect of load on variance of request distance with $\boldsymbol{c=2}$ across MTR and UGS. (f) Comparison of expected request distance under Two Resource Non-homogeneous (TRN), Two Resource Homogeneous (TRH), Single Resource Unit-service (SRU) and Single Resource Bulk-service (SRB) scenario across various server distributions with $\boldsymbol{\lambda=0.6, \mu=1}$.

$p_1 = (1/2)\big(1+\sqrt{(c_v^2-1)/(c_v^2+1)}\big), p_2 = 1-p_1, \mu_1 = 2p_1/\alpha_X$ and $\mu_2 = 2p_2/\alpha_X$. Unless specified, for H_2 we take $c_v^2 = 4$ with $c=2$. Also if not specified, users are distributed according to a Poisson process and servers a according to a renewal process.

We consider a collection of 10^5 users and 10^5 servers, i.e. $|R| = |S| = 10^5$. We assign users to servers according to MTR. Let $R_M \subseteq R$ be the set of users allocated under MTR. Clearly $|R_M| \leq |R|$. We then run optimal and other greedy policies on the set R_M and S. For each of the experiments, the expected request distance for the corresponding policy is averaged over 50 trials.

9.2. Sensitivity analysis

9.2.1. Expected request distance vs. load

We first study the effect of load ($=\lambda/c\mu$) on $\mathbb{E}[D]$ as shown in Figure 6(a). Clearly $\mathbb{E}[D]$ increases as a function of load. Note that H_2 distribution exhibits the largest expected request distance and the deterministic distribution, the smallest because the servers are evenly spaced. While for H_2, c_v^2 is larger than for the exponential distribution. Consequently servers are clustered, which increases $\mathbb{E}[D]$.

9.2.2. Expected request distance vs. squared co-efficient of variation

We now examine how c_v^2 affects $\mathbb{E}[D]$ when ρ is fixed. We compare two systems: a general request with Poisson distributed servers (H_2/M) and a Poisson request with general distributed servers (M/H_2) where the general distribution is a H_2 distribution with the same set of parameters, i.e. we fix $\lambda=\mu=1$ with $c=2$. The results are shown in Figure 6(b). Note that, when $c_v^2 = 1$ H_2 is an exponential distribution and both H_2/M and M/H_2 are identical M/M/1 systems.

As discussed in the previous graph, performance of both systems decreases with increase in c_v^2 due to increase in the variability of user and server placements. However, from Figure 6(b) it is clear that performance is more sensitive to server placement as compared to the corresponding user placement.

9.2.3. Expected request distance vs. server capacity

We now focus on how server capacity affects $\mathbb{E}[D]$ as shown in Figure 6(c). We fix $\rho = 0.8$. With an increase in c, while keeping ρ fixed, $\mathbb{E}[D]$ decreases. This is because queuing delay decreases. Note that $\mathbb{E}[D]$ gradually converge to a value with increase in server capacity. Theoretically, this can be explained by our discussion on uncapacitated allocation in Section 7.3. As $c \to \infty$ the contribution of queuing delay to $\mathbb{E}[D]$ vanishes and $\mathbb{E}[D]$ becomes insensitive to c.

9.2.4. Expected request distance vs. capacity moments

We investigate the heterogeneous capacity scenario as discussed in Section 7.2. Consider the plot shown in Figure 6(d). We fix $\rho = 0.8$. For the variable server capacity curve we choose a value for server capacity for each server uniformly at random from the set $\{1, 2, \ldots, 2c\}$. For the constant server capacity curve we deterministically assign server capacity c to each server. We observe better performance for constant server capacity curve at lower values of c under Deterministic distribution. Variability in constant server case is zero, thus explaining its better performance. Both the curves exhibit similar performance under H_2 distribution as well.

9.2.5. Variance vs. load

We now study the effect of load on variance of request distance as shown in Figure 6(e). Clearly variance increases as a function of load. Also note that UGS has a higher variance as compared to MTR across all values of load and across various inter-server distance distributions. Provable results exist (from queueing theory) that among all service disciplines the variance of the request distance (or sojourn time in queueing terminology) is minimized under MTR (a FCFS based policy) for Poisson request arrivals and exponential inter-server distances (or service times) [13]. However, these results do not generalize to other inter-server distance distributions in an exceptional service accessible batch queueing discipline. Our simulation based results in Figure 6(e) thus bolster our observation in Remark 2 mentioned in Section 4. Again, a deterministic equidistant placement of servers produce the least variance for request distance among all other placements.

9.2.6. Comparison of two resource and single resource policies

We compare the performance of MTR under various two resource (TR) and single resource (SR) settings as shown in Figure 6(f). For a two resource setting, denote $[\mu_1, \mu_2]$ as the server densities associated with resource types 1 and 2 respectively as described in Section 7.5. Denote c as the server capacity associated with each resource type. We define a Two Resource Homogeneous (TRH) system to be a two resource setting with $\mu_1 = \mu_2 = \mu$. We define a Two Resource Non-homogeneous (TRN) system to be a two resource setting with $\mu_1 \neq \mu_2$. For simulation purpose, we chose $\mu_1 = \mu + \epsilon$ and $\mu_2 = \mu - \epsilon$ such that the effective server density remains μ. We also choose $c = 1$. A Single Resource Unit-service (SRU) system is a single resource system with server density μ and $c = 1$. A Single Resource Bulk-service (SRB) system is also a single resource system with server density $\mu/2$ and $c = 2$. Note that the request density and effective server densities ($c\mu$) are same in all settings. From Figure 6(f), it is clear that TRH performs better than TRN across all server distributions. This advocates for maintaining similar densities for each resource type in a two resource system. As expected, a deterministic equidistant placement of servers produce the least expected request distance for each system among all other choice of placements. SRB in deterministic server placement scenario performs the best among all other settings. However, it does not perform well with other server distributions. Also, note that, TRH has a higher expected request distance as compared to SRU across all server distributions. Thus Equation (61) in Section 7.5 holds true even under non-markovian setting.

9.3. Comparison of different allocation policies

We consider the case in which both users and servers are distributed according to Poisson processes. From Figure 7 (a), we observe that due to its directional nature MTR has a larger expected request distance compared to other policies while GS provides near optimal performance. At low loads i.e. when $\rho \ll 1$, the Nearest Neighbor policy policy performs similar to the optimal policy. But as $\rho \to 1$, the NN policy perform worse.

In Figure 7 (b), we compare the performance of allocation policies across different server capacities. The expected request distance decreases with increase in server capacities across all policies. NN, GS and the optimal policy converge

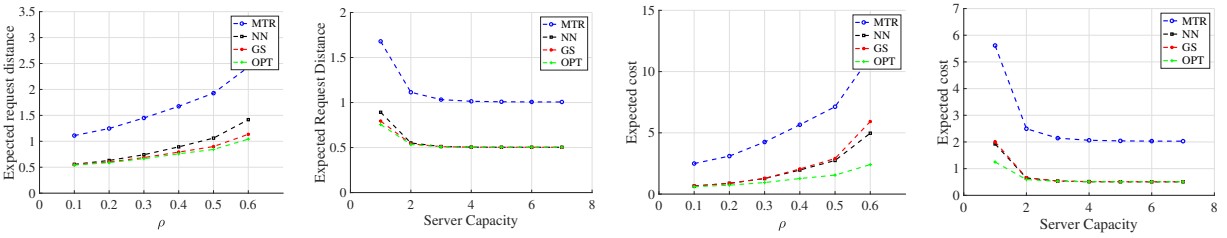


Figure 7: Comparison of different allocation policies: (a) ρ vs $\mathbb{E}[D]$ with $c = 1$, (b) c vs. $\mathbb{E}[D]$ with $\rho = 0.4$, (c) ρ vs $\overline{T}$ with $\beta = 2, t_0 = 1, c = 1$ and (d) c vs. $\overline{T}$ with $\beta = 2, t_0 = 1, \rho = 0.4$.

to the same value as c gets higher.

We now consider the expected communication cost as the performance metric. We use a cost model described in Section 7.4 with the parameter $\beta = 2$ and $t_0 = 1$. From Figure 7 (c), we observe that while at low loads i.e. when $\rho \ll 1$, GS and NN perform similar to the optimal policy, as ρ increases both GS and NN perform worse. Note that, NN has a higher expected request distance than GS at high load as shown in Figure 7 (a). However, the performance is reversed with $\beta = 2$, i.e. NN has a lower expected cost than GS at high load as shown in Figure 7 (c). This depicts the effect of β on the performance of various allocation policies. In Figure 7 (d), we observe that NN, GS and the optimal policy converge to the same value as c gets higher.

We observe similar trends in the case of deterministic inter-server distance distributions. However, under equal densities, all the policies produce smaller expected request distance as compared to their Poisson counterpart. This advocates for placing equidistant servers in a bidirectional system with Poisson distributed requesters to minimize expected request distance.

10. Allocating Resources in 2D

We now consider the case where requesters and servers are located on the two-dimensional plane, $\mathbb{R}^2$. The problem of minimizing the expected request distance can be solved by first constructing a complete $R \times S$ bipartite graph with edge weights $w_{r,s} = \|r, s\|_2, r \in R, s \in S$; followed by executing the Hungarian matching algorithm whose time complexity is $O(n^3)$, where $n = |R| + |S|$[11]. In this section, we present a heuristic algorithm, which leverages the optimal dynamic programming scheme for one-dimensional inputs to solve the two-dimensional problem, has $O(n^2)$ time complexity, and empirically yields request distances within a constant factor of the optimal solution.

The key insight is to *embed* the points denoting $R, S \subset \mathbb{R}^2$ into new locations in $\mathbb{R}$ such that the distances between a requester $r \in R$ and its K nearest neighbors (servers) $s \in Neighbors(r)$ are approximately preserved. We observe that while distance-preserving or even low-distortion embeddings into a very low dimensional space like $\mathbb{R}$ typically do not exist, embeddings that preserve distances to K nearest neighbors of the other node type (for not too large K) may be plausible. This is useful because preserving the nearest servers from $\mathbb{R}^2$ to $\mathbb{R}$ provides a reasonable opportunity for the Dynamic Programming algorithm outlined in Section 8 to find good matchings.

We achieve the aforementioned embedding by adapting a non-linear dimensionality reduction method such as Locally Linear Embedding (LLE) [19], which consists of the steps outlined below.

Estimation of nearest neighbor weights. For each requester r_i, select K nearest servers $s_{i1}, s_{i2}, \ldots, s_{iK}$[12]. Estimate a set of weights $w_{i1}, w_{i2}, \ldots, w_{iK}$ such that the point r_i can be reconstructed from $s_{i1}, s_{i2}, \ldots, s_{iK}$: $r_i = \sum_{j=1}^{K} w_{ij} s_{ij}$. Similarly for each server s_i, select K' nearest requesters $r_{i1}, r_{i2}, \ldots, r_{iK'}$ and estimated weights such that point s_i can be reconstructed from the nearest neighbor requester locations: $s_i = \sum_{j=1}^{K} w_{ij} r_{ij}$.

This can be achieved by minimizing the reconstruction error for each node $i \in R \times S$. Suppose K is fixed for both requesters and servers. W is an $n \times n$ matrix of weights where $n = |R| + |S|$ and the i-th row W_i, which corresponds

[11] While the specific case where the weights are Euclidean distances can be solved by Agrawal's algorithm in $O(n^{2+\epsilon})$ time, for a more general weight function the more expensive Hugarian algorithm is needed.

[12] Note that in general, the nearest servers need not be the ones with the smallest Euclidean distance from r_i; they could be the ones with low *costs* to r_i. However in this section, we equate the costs with the Euclidean distance.

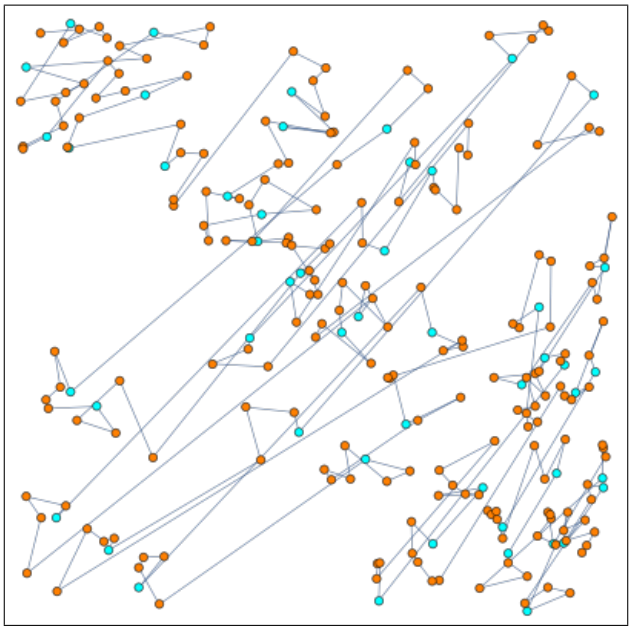


Figure 8: Approximate nearest-K-distance preserving $\mathbb{R}^2 \to \mathbb{R}$ embedding ($K = 25\%$)

to node i, has K non-zero elements. The structure of W is as follows:

$$W = \begin{pmatrix} 0 & W_{RS} \\ W_{SR} & 0 \end{pmatrix},$$

where W_{RS} and W_{SR} are rectangular matrices with dimensions $|R| \times |S|$ and $|S| \times |R|$, respectively. If the two-dimensional coordinates of node i are represented by vector $\mathbf{x_i}$, the reconstruction error can be defined by:

$$\epsilon(W_i) = \|\mathbf{x_i} - \sum_{j=1}^{K} W_{ij}\mathbf{x_j}\|^2. \tag{64}$$

It was shown in [19] that $\epsilon(W_i)$ is minimized when $W_i = (G_i + \lambda I)^{-1}\mathbf{1}$, where $G_i(j,k) = (\mathbf{x_j} - \mathbf{x_i}).(\mathbf{x_k} - \mathbf{x_i})$, $\mathbf{1}$ is the vector of all ones, and λ is chosen such that the elements of W_i add up to 1.

Computing optimal embedding in $\mathbb{R}$. LLE suggests that the relationships between the n points in the higher dimensional space ($\mathbf{R}^2$ in our case) captured by the matrix W should be approximately preserved in the lower dimensional space ($\mathbf{R}$ in our case). Then the optimal embedding $\mathbf{y} = \{y_1, y_2, \ldots, y_n\}, y_i \in \mathbb{R}$ can be found by solving the following quadratic optimization problem:

$$\begin{array}{ll} \min & \sum_{i=1}^{n}(y_i - \sum_j W_{ij}y_j)^2 = \mathbf{y}^T(I-W)^T(I-W)\mathbf{y}, \\ \text{subject to:} & \mathbf{y}^T\mathbf{y} = 1 \end{array} \tag{65}$$

$(I-W)^T(I-W)$ is a positive semi-definite sparse matrix (since $K << n$) and the optimal solution to this "eigenvalue" problem is given by the eigenvector corresponding to the smallest non-zero eigenvalue of $(I-W)^T(I-W)$ [19]. Since we do not need to compute all the eigenvectors, the second smallest eigenvalue of a matrix can be computed efficiently without performing a matrix diagonalization using the Arnoldi algorithm in running time $O(n^2)$.

Using the $\mathbb{R}$*-embedding for matching.* After generating the embedding $\mathbf{y}$, we applied our Dynamic Programming Algorithm to compute the best resource allocation scheme. However, naive application of the algorithm led to high expected request distances. The reason behind this is illustrated in Figure 8, which visualizes an embedding computed for a given set of requesters and servers from $\mathbb{R}^2$ to $\mathbb{R}$; the zigzag lines denote the linear order imposed by the embedding.

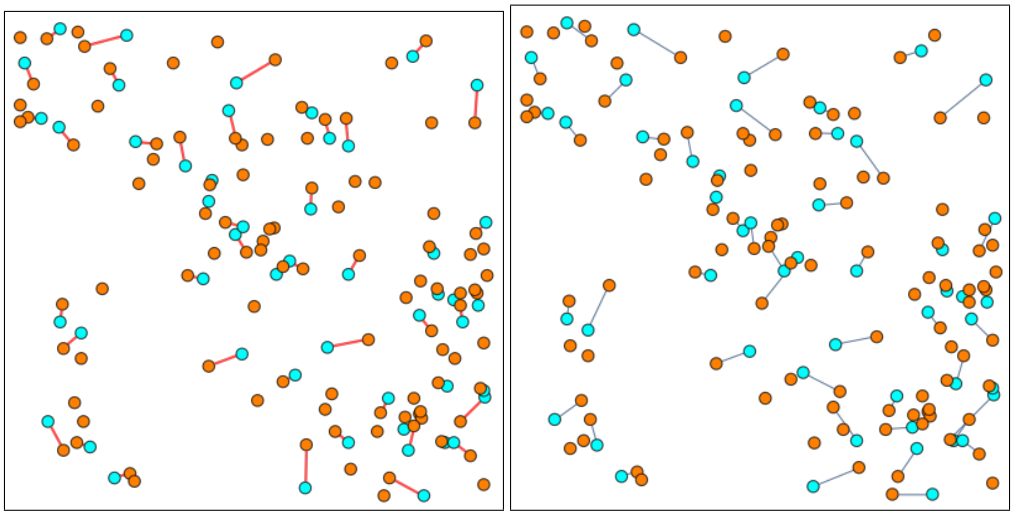


Figure 9: Matching for a clustered distribution: $|R| = 50$ requesters are spread uniformly at random in a square $[0,1]\times[0,1]$ and each of the $|S| = 100$ servers is located uniformly at random in a box of size 0.1 around a randomly selected resource. (a) Optimum weighted bipartite matching in 2D ($\mathbb{E}[D] = 0.0435$); (b) Approximate matching after 1D embedding ($\mathbb{E}[D] = 0.06$) ($K = 25\%$)

$\lvert R\rvert$	$\lvert S\rvert$	$\mathbb{E}[D]$ Optimum	$\mathbb{E}[D]$ $\mathbb{R}^2 \rightarrow \mathbb{R}$
200	400	0.023	0.05
200	500	0.0205	0.041
200	600	0.018	0.037
200	700	0.0166	0.0336
200	800	0.0158	0.0326
200	900	0.0151	0.02989
200	1000	0.014	0.02985

Table 2
Matching in larger networks: requesters are spread uniformly at random in a square $[0,1]\times[0,1]$ and each server is located uniformly at random in a box of size 0.1 around a randomly selected resource.

It is easy to see that it is quite possible that a pair of (requester (blue), server (orange)) nodes which are far away in $\mathbb{R}^2$ may be pretty close to each other in $\mathbb{R}$. Since our LLE-based scheme only tries to preserve close-by neighbors and does not explicitly attempt to repel nodes that are farther away in $\mathbb{R}^2$, such pairs of nodes could end up being embedded close to each other in $\mathbb{R}$. To circumvent this problem, we propose a heuristic scheme to adjust the embedding $\mathbf{y}$ such that whenever for a pair of nodes $\{i, j\}$ we have $\|\mathbf{x_i} - \mathbf{x_j}\| > \Delta$ but $\|y_k - y_{k+1}\| < \epsilon$, where $\mathbf{x_i}$ is mapped to y_k and $\mathbf{x_j}$ is mapped to y_{k+1}, and Δ, ϵ, δ are configurable constants, we increase the distance between y_k and y_{k+1} by adding a large cumulative constant $c_{k+1} = c_k + \delta$ to y_{k+1}. This adjustment of $\mathbf{y}$ sequentially *spreads out* the points in $\mathbb{R}$ toward the right and the Dynamic Programing Algorithm is then able to find good requester-server matchings.

Figure 9 shows a comparison between an optimum matching and an approximate matching constructed by the embedding methods proposed in this section. Table 2 shows results for the case when $|S|$ is varied for a fixed $|R|$. We can observe that the $\mathbb{R}^2 \rightarrow \mathbb{R}$ embedding approach yields a solution which is empirically close to $2 \times OPT$. Given that this procedure has a lower time complexity $O(n^2)$[13] than the usual $O(n^3)$ for Hungarian algorithm, it could be practically useful for large resource allocation problems.

[13] Both the embedding process and the Dynamic Programming algorithm have time complexity $O(n^2)$.

11. Conclusion

We introduced a queuing theoretic model for analyzing the behavior of unidirectional policies to allocate tasks to servers on the real line. We showed the equivalence of UGS and MTR w.r.t the expected request distance and presented results associated with the case when either requesters or servers were Poisson distributed. In this context, we analyzed a new queueing theoretic model: ESABQ, not previously studied in queueing literature. We also proposed a dynamic programming based algorithm to obtain an optimal allocation policy in a bi-directional system. We performed sensitivity analysis for unidirectional system and compared the performance of various greedy allocation strategies along with the unidirectional policies to that of optimal policy. We proposed a heuristic based approximate solution to the optimal assignment problem for the two-dimensional scenario. Going further, we aim to extend our analysis for unidirectional policies to a two-dimensional geographic region.

12. Acknowledgment

This research was sponsored by the U.S. Army Research Laboratory and the U.K. Defence Science and Technology Laboratory under Agreement Number W911NF-16-3-0001 and by the NSF under grant NSF CNS-1617437. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Army Research Laboratory, the U.S. Government, the U.K. Defence Science and Technology Laboratory. This document does not contain technology or technical data controlled under either the U.S. International Traffic in Arms Regulations or the U.S. Export Administration Regulations.

References

[1] Abadi, H.K., Prabhakar, B., 2017. Stable Matchings in Metric Spaces: Modeling Real-World Preferences using Proximity. arXiv:1710.05262 .

[2] Adan, I.J.B.F., Van Leeuwaarden, J.S.H., Winands, E.M.M., 2006. On the Application of Rouché's Theorem in Queueing Theory. Operations Research Letters 34, 355–360.

[3] Agarwal, P., Efrat, A., Sharir, M., 1995. Vertical Decomposition of Shallow Levels in 3-Dimensional Arrangements and Its Applications. SOCG .

[4] Ahuja, R., Magnanti, T., Orlin, J., 1993. Network Flows: Theory, Algorithms, and Applications. Prentice-Hall, Inc.

[5] Atzori, L., Iera, A., Morabito, G., 2010. The Internet of Things: A Survey. Computer Networks 54, 2787–2805.

[6] Bailey, N.T.J., 1954. On Queueing Processes with Bulk Service. J. R. Stat. SOCE. 16, 80–87.

[7] Bukac, J., 2018. Matching On a Line. arXiv:1805.00214 .

[8] Doshi, S., Bhandare, S., 2002. An On-demand Minimum Energy Routing Protocol for a Wireless ad-hoc Network, in: ACM Mobile Computing and Communications Review.

[9] Gale, D., Shapley, L., 1962. College Admissions and Stability of Marriage. Amer. Math. Monthly 69 , 9–15.

[10] Goswami, V., Laxmi, P.V., 2011. Performance Analysis of a Renewal Input Bulk Service Queue with Accessible and non-accessible Batches. Quality Technology and Quantitative Management 8, 87–100.

[11] Ho, I.W.H., Leung, K.K., Polak, J.W., 2011. Stochastic Model and Connectivity Dynamics for vanets in Signalized Road Systems. IEEE/ACM Transactions on Networking 19, 195–208.

[12] Holroyd, A.E., Pemantle, R., Peres, R., Schramm, O., 2009. Poisson Matching. Annales de l IHP Probabilites et Statistiques 45, 266–287.

[13] Kingman, F., 1962. The Effect of Queue Discipline on Waiting Time Variance. Math. Proc. Cambridge Phil. Soc. 58, 163–164.

[14] Leung, K.K., Massey, W.A., Whitt, W., 1994. Traffic Models for Wireless Communication Networks. IEEE Journal on Selected Areas in Communications 12, 1353–1364.

[15] Mezard, M., Parisi, G., 1988. The Euclidean Matching Problem. J. Phys. France 49, 2019–2025.

[16] Nelson, R., Tantawi, A., 1988. Approximate Analysis of Fork/join Synchronization in Parallel Queues. IEEE Transactions on Computers 37, 739–743.

[17] Orlin, J., 1997. A Polynomial Time Primal Network Simplex Algorithm for Minimum Cost Flows. Mathematical Programming 78, 109–129.

[18] Reingold, E.M., Tarjan, R.E., 1981. On a Greedy Heuristic for Complete Matching. SIAM Journal on Computing 10, 676–681.

[19] Roweis, S.T., Saul, L.K., 2000. Nonlinear dimensionality reduction by locally linear embedding. Science 290, 2323–2326. URL: https://science.sciencemag.org/content/290/5500/2323, doi:10.1126/science.290.5500.2323, arXiv:https://science.sciencemag.org/content/290/5500/2323.full.pdf.

[20] Shortle, J.F., Thompson, J.M., Gross, D., Harris, C., 2018. Fundamentals of Queueing Theory. Fifth Edition, Wiley Series in Probability and Statistics .

[21] Stuart, E., 2010. Matching Methods for Causal Inference: a Review and a Look Forward. Stat. Sci. 25, 1–21.

[22] Welch, P., 1964. On a Generalized m/g/1 Queuing Process in Which The First Customer of Each Busy Period Receives Exceptional Service. Operations Research 12, 736–752.

13. Appendix

13.1. Derivation of F_Z for various inter-server distance distrbutions

13.1.1. $F_X(x) \sim$ Exponential(μ)

In this case, both X and Y are exponentially distributed. Thus the difference distribution is given by

$$D_{XY}(x) = 1 - \frac{\lambda}{\lambda+\mu} e^{-\mu x}, \text{when } x \geq 0 \tag{66}$$

Combining (9) and (66), we get

$$F_Z(x) = \frac{1 - \frac{\lambda}{\lambda+\mu} e^{-\mu x} - 1 + \frac{\lambda}{\lambda+\mu}}{\frac{\lambda}{\lambda+\mu}} = 1 - e^{-\mu x}. \tag{67}$$

Thus we obtain $F_X(x) = F_Z(x) \sim$ Exponential(μ).

13.1.2. $F_X(x) \sim$ Uniform$(0, b)$

The c.d.f. for uniform distribution is

$$F_X(x) = \begin{cases} \frac{x}{b}, & 0 \leq x \leq b; \\ 1, & x > b, \end{cases} \tag{68}$$

where b is the uniform parameter. Thus we have

$$D_{XY}(x) = \int_0^\infty F_X(x+y)\lambda e^{-\lambda y} dy = \left[\int_0^{b-x} \frac{x+y}{b} \lambda e^{-\lambda y} dy\right] + \left[\int_{b-x}^\infty 1 \ \lambda e^{-\lambda y} dy\right] = \frac{\lambda x - e^{-\lambda(b-x)} + e^{-\lambda b}}{b\lambda + e^{-\lambda b} - 1} \tag{69}$$

Taking $k_\lambda = 1/(b\lambda + e^{-\lambda b} - 1)$ and using Equation (9) we have

$$F_Z(x) = k_\lambda \left[\lambda x + e^{-\lambda b}(1 - e^{\lambda x})\right] \quad \text{and} \quad f_Z(x) = \lambda k_\lambda \left[1 - e^{-\lambda b} e^{\lambda x})\right]. \tag{70}$$

Taking $\alpha_Z = \int_0^b x f_Z(x) dx$ and $\sigma_Z^2 = [\int_0^b x^2 f_Z(x) dx] - \alpha_Z^2$ we have

$$\alpha_Z = \frac{b^2\lambda}{2} k_\lambda - \frac{1}{\lambda}, \quad \sigma_Z^2 = \frac{b^3\lambda}{3} k_\lambda - \frac{k_\lambda}{\lambda}\left[b(b\lambda - 2) + \frac{2}{\lambda}(1 - e^{-\lambda b})\right] - \alpha_Z^2, \quad \alpha_X = b/2, \quad \sigma_X^2 = b^2/12. \tag{71}$$

13.1.3. $F_X(x) \sim$ Deterministic(d_0)

Another interesting scenario is when servers are equally spaced at a distance d_0 from each other i.e. when $F_X(x) \sim$ Deterministic(d_0). The c.d.f. for deterministic distribution is

$$F_X(x) = \begin{cases} 0, & 0 \leq x < d_0; \\ 1, & x \geq d_0, \end{cases} \tag{72}$$

where d_0 is the deterministic parameter. A similar analysis as that of uniform distribution yields

$$F_Z(x) = c_\lambda \left[e^{-\lambda(d_0 - x)} - e^{\lambda d_0}\right]; \ f_Z(x) = \lambda c_\lambda \left[e^{-\lambda(d_0 - x)}\right], \tag{73}$$

where $c_\lambda = 1/(1 - e^{-\lambda d_0})$. Thus we have

$$\alpha_Z = c_\lambda \frac{d_0\lambda + e^{-\lambda d_0} - 1}{\lambda}, \quad \sigma_Z^2 = \frac{c_\lambda}{\lambda}\left[d_0(d_0\lambda - 2) + \frac{2}{\lambda}(1 - e^{-\lambda d_0})\right] - \alpha_Z^2, \quad \alpha_X = d_0, \quad \sigma_X^2 = 0. \tag{74}$$

13.2. ESABQ under PRGS

13.2.1. Chapman-Kolmorogov equations

Let us write the Chapman-Kolmorogov equations for the Markov chain $\{(L(t), R(t), I(t)),\ t \geq 0\}$ defined in Section 6.3.1.

For $n \geq 2$ and $x > 0$ we get

$$\frac{\partial}{\partial t}p_t(n,x;1) = \frac{\partial}{\partial x}p_t(n,x;1) - \lambda p_t(n,x;1) - \frac{\partial}{\partial x}p_t(n,0;1) + \lambda p_t(n-1,x;1)$$
$$\frac{\partial}{\partial t}p_t(n,x;2) = \frac{\partial}{\partial x}p_t(n,x;2) - \lambda p_t(n,x;2) - \frac{\partial}{\partial x}p_t(n,0;2) + \lambda p_t(n-1,x;2) + F_X(x)\frac{\partial}{\partial x}p_t(n+c,0;1) + F_X(x)\frac{\partial}{\partial x}p_t(n+c,0;2).$$

Letting $t \to \infty$ yields

$$0 = \frac{\partial}{\partial x}p(n,x;1) - \lambda p(n,x;1) - \frac{\partial}{\partial x}p(n,0;1) + \lambda p(n-1,x;1) \tag{75}$$
$$0 = \frac{\partial}{\partial x}p(n,x;2) - \lambda p(n,x;2) - \frac{\partial}{\partial x}p(n,0;2) + \lambda p(n-1,x;2) + F_X(x)\frac{\partial}{\partial x}p(n+c,0;1) + F_X(x)\frac{\partial}{\partial x}p(n+c,0;2). \tag{76}$$

For $n = 1$, $x > 0$

$$\frac{\partial}{\partial t}p_t(1,x;1) = \frac{\partial}{\partial x}p_t(1,x;1) - \lambda p_t(1,x;1) - \frac{\partial}{\partial x}p_t(1,0;1) + \lambda p_t(0)F_Z(x)$$
$$\frac{\partial}{\partial t}p_t(1,x;2) = \frac{\partial}{\partial x}p_t(1,x;2) - \lambda p_t(1,x;2) - \frac{\partial}{\partial x}p_t(1,0;2) + F_X(x)\frac{\partial}{\partial x}p(1+c,0;1) + F_X(x)\frac{\partial}{\partial x}p_t(1+c,0;2).$$

Letting $t \to \infty$ yields

$$0 = \frac{\partial}{\partial x}p(1,x;1) - \lambda p(1,x;1) - \frac{\partial}{\partial x}p(1,0;1) + \lambda p(0)F_Z(x) \tag{77}$$
$$0 = \frac{\partial}{\partial x}p(1,x;2) - \lambda p(1,x;2) - \frac{\partial}{\partial x}p(1,0;2) + F_X(x)\left(\frac{\partial}{\partial x}p(1+c,0;1) + \frac{\partial}{\partial x}p(1+c,0;2)\right), x > 0. \tag{78}$$

We can collect the results in (75)-(78) as follows: for $n \geq 1$, $x > 0$,

$$0 = \frac{\partial}{\partial x}p(n,x;1) - \lambda p(n,x;1) - \frac{\partial}{\partial x}p(n,0;1) + \lambda p(n-1,x;1)\mathbf{1}(n \geq 2) + \lambda p(0)F_Z(x)\mathbf{1}(n = 1) \tag{79}$$
$$0 = \frac{\partial}{\partial x}p(n,x;2) - \lambda p(n,x;2) - \frac{\partial}{\partial x}p(n,0;2) + \lambda p(n-1,x;2)\mathbf{1}(n \geq 2) + F_X(x)\left(\frac{\partial}{\partial x}p(n+c,0;1) + \frac{\partial}{\partial x}p(n+c,0;2)\right). \tag{80}$$

Define $g(n,x) = p(n,x;1) + p(n,x;2)$ for $n \geq 1$, $x > 0$. Summing (79) and (80) gives

$$0 = \frac{\partial}{\partial x}g(n,x) - \lambda g(n,x) - \frac{\partial}{\partial x}g(n,0) + \lambda g(n-1,x)\mathbf{1}(n \geq 2) + \lambda p(0)F_Z(x)\mathbf{1}(n = 1) + F_X(x)\frac{\partial}{\partial x}g(n+c,0),$$
$$\forall n \geq 1, x > 0. \tag{81}$$

13.2.2. Multiplicity of roots of $z^c - F_X^*(\lambda(1-z))$

Assume that $F_X(x) = 1 - e^{-\mu x}$ (regular batch service times are exponentially distributed). Then,

$$z^c - F_X^*(\lambda(1-z)) = \frac{-\rho z^{c+1} + (1+\rho)z^c - 1}{1 + \rho(1-z)}.$$

$z^c - F_X^*(\lambda(1-z)) = 0$ for $|z| \leq 1$ iff $Q(z) := -\rho z^{c+1} + (1+\rho)z^c - 1 = 0$. The derivative of $Q(z)$ is $Q'(z) = z^{c-1}((1+\rho)c - \rho(c+1)z)$. It vanishes at $z = 0$ and at $z = \frac{(1+\rho)c}{\rho(c+1)} > 1$ under the stability condition $\rho < c$. Since $z = 0$ is not a zero of $Q(z)$, we conclude that all zeros of $z^c - F_X^*(\lambda(1-z))$ in $\{|z| \leq 1\}$ have multiplicity one.

More generally, it is shown in [6] that all zeros of $z^c - F_X^*(\lambda(1-z))$ in $\{|z| \leq 1\}$ have multiplicity one if F_X is a χ^2-distribution with an even number $2p$ of degrees of freedom, i.e. $dF_X(x) = \frac{a^p}{\Gamma(p)}x^{p-1}e^{-ax}dx$ so that $1/\mu = p/a$.

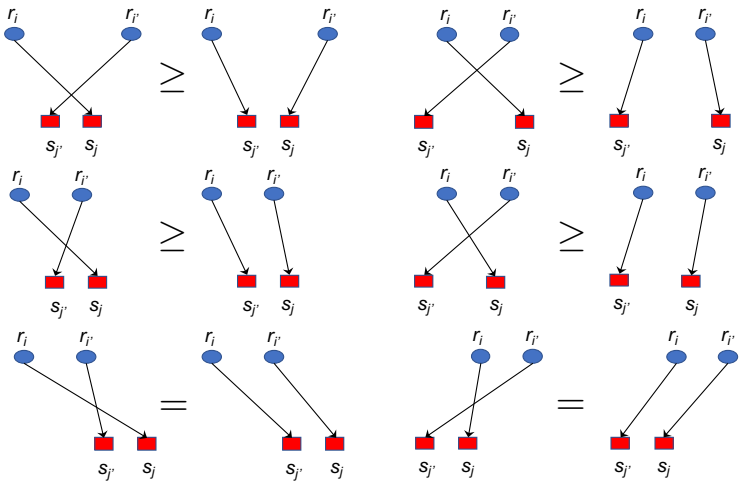


Figure 10: Uncrossing an assignment either reduces request distance or keeps it unchanged.

13.2.3. Roots of $A(z)$

Define $A(z) = F_X^*(\theta(z))$. If $A(z)$ has a radius of convergence larger than one (i.e. $A(z)$ is analytic for $|z| \leq \nu$ with $\nu > 1$) and $A'(1) < c \in \{1, 2, \ldots\}$ a direct application of Rouché's theorem shows that $z^c - A(z)$ has c zeros in the unit disk $\{|z| \leq 1\}$(see e.g. [2]). If the radius of convergence of $A(z)$ is one, $A(z)$ is differentiable at $z = 1$, $A'(1) < c$, and $z^c - A(z)$ has period p, then $z^c - A(z)$ has exactly $p \leq s$ zeros on the unit circle and $s - p$ zeros inside the unit disk $\{|z| < 1\}$ [2, Theorem 3.2]. Assume that the stability condition $\frac{d}{dz}A(z)|_{z=1} = \rho < c$ holds. $A(z)$ has a radius of convergence larger than one when F_X is the exponential/Erlang/Gamma/ etc probability distributions.

13.2.4. Special Cases

One easily checks that (31) gives the classical Pollaczek-Khinchin formula for the M/G/1 queue when $c = 1$ and $F_Z = F_X$.

Let now $c = 1$ in (31) with F_Z and F_X arbitrary. Then,

$$N(z) = \frac{a_1}{\lambda}\left(\frac{F_X^*(\lambda(1-z)) - zF_Z^*(\lambda(1-z))}{F_X^*(\lambda(1-z)) - z}\right)$$

gives the z-transform of the stationary number of customers in a M/G/1 queue with an exceptional first customer in a busy period. The constant a_1/λ is obtained from the identity $N(1) = 1$ by application of L'Hopital's rule, which gives[14] $a_1/\lambda = (1-\rho)/(1-\rho+\rho_Z)$. This gives

$$N(z) = \frac{1-\rho}{1-\rho+\rho_Z}\left(\frac{F_X^*(\lambda(1-z)) - zF_Z^*(\lambda(1-z))}{F_X^*(\lambda(1-z)) - z}\right).$$

The above is a known result [22].

If $F_Z^* = F_X^* := F^*$, then

$$N(z) = \frac{\sum_{k=1}^{c} a_k \left[(z^c - z^k)z^c + ((1-z^c)z - (1-z^k))F^*(\theta(z))\right]}{\theta(z)(z^c - F^*(\theta(z))}.$$

13.3. Results for Section 7.2

13.3.1. Derivation of $v_1(z)$ and $v_2(z)$

$v_1(z)$ in (43) can further be simplified to

$$v_1(z) = \sum_{l=0}^{\infty} z^l \sum_{m=0}^{l} \pi_m \sum_{i=0}^{c} k_{i+l-m}p_i = \sum_{m=0}^{\infty} \pi_m \sum_{l\geq m} z^l \sum_{i=0}^{c} k_{i+l-m}p_i = \sum_{m=0}^{\infty} \pi_m z^m \sum_{l\geq m} z^{l-m} \sum_{i=0}^{c} k_{i+l-m}p_i = \sum_{m=0}^{\infty} \pi_m z^m \sum_{j=0}^{\infty} z^j \sum_{i=0}^{c} k_{i+j}p_i$$

[14] Note that we retrieve this result by letting $c = 1$ in (32).

$$= N(z)\sum_{i=0}^{c} p_i z^{-i} \sum_{j=0}^{\infty} z^{i+j} k_{i+j} = N(z)\sum_{i=0}^{c} p_i z^{-i}\left[K(z) - \sum_{j=0}^{i} k_j z^j + k_i z^i\right] = N(z)\left\{\sum_{i=0}^{c} p_i z^{-i}\left[K(z) - \sum_{j=0}^{i} k_j z^j\right] + \sum_{i=0}^{c} k_i z^i\right\}. \quad (82)$$

Similarly $v_2(z)$ in (44) can further be simplified to

$$v_2(z) = \sum_{l=0}^{\infty} z^l \sum_{m=l+1}^{c+l} \pi_m \sum_{i=m-l}^{c} k_{i+l-m} p_i = \left[\sum_{l=0}^{\infty} z^l \sum_{m=l}^{c+l} \pi_m \sum_{i=m-l}^{c} k_{i+l-m} p_i\right] - N(z)\sum_{i=0}^{c} k_i z^i = \left[\sum_{m=0}^{c} z^{-m} \sum_{i=m}^{c} k_{i-m} p_i \sum_{l=0}^{\infty} z^{m+l} \pi_{m+l}\right] - N(z)\sum_{i=0}^{c} k_i z^i$$
$$= \left[\sum_{m=0}^{c} z^{-m} \sum_{i=m}^{c} k_{i-m} p_i \left\{N(z) - \sum_{j=0}^{m-1} \pi_j z^j\right\}\right] - N(z)\sum_{i=0}^{c} k_i z^i. \quad (83)$$

13.3.2. Roots of $A(z)$

Denote $A(z) = K(z)\sum_{i=0}^{c} p_{c-i} z^i$. Clearly, $A(z)$ is also a probability generating function (*pgf*) for the non-negative random variable $V + \tilde{C}$ where $\tilde{C}$ is a random variable on $\{0, \ldots, c-1\}$ with distribution $\Pr(\tilde{C} = j) = p_{c-j}, \forall j \in \{0, 1, \ldots, c-1\}$. Also we have

$$A'(1) = K'(1) + \sum_{i=0}^{c} p_{c-i} i = \rho + \sum_{i=1}^{c} p_i (c-i) = \rho + \sum_{i=1}^{c} p_i c - \sum_{i=1}^{c} i p_i = \rho + c - \overline{C}$$

From our stability condition we know that $\rho < \overline{C}$. Thus $A'(1) < c$. Since $A(z)$ is a pgf and $A'(1) < c$, by applying the arguments from [2, Theorem 3.2] we conclude that the denominator of equation (49) has $c-1$ zeros inside and one on the unit circle, $|z| = 1$.

13.4. Proof of Lemma 1

Proof. It can be observed that if such a 4-tuple (i, j, i', j') exists, the cost can be reduced by assigning i to j' and i' to j, hence we arrive at a contradiction. To show this, consider the six possible cases of relative ordering between $r_i, r_{i'}, s_j, s_{j'}$ which obey $r_i < r_{i'}$ and $s_j > s_{j'}$. We give a pictorial proof in Figure 10[15]. It is easy to see that in each of the cases, the request distance of the *uncrossed* assignment is either smaller or remains unchanged. □

[15] For ease of exposition, the requesters and servers are shown to be located along two separate horizontal lines, although they are located on the same real-line.