Online Two-sided Bipartite Matching

Van-Anh Truong, Xinshang Wang
Department of Industrial Engineering and Operations Research, Columbia University, New York, NY, USA, vatruong@ieor.columbia.edu, xw2230@columbia.edu

We study a class of bipartite matching problems in which both demand and supply units of various types arrive randomly and sequentially over time. Supply units can wait for a fixed amount of time, but demand units must be matched irrevocably upon arrival to existing supply units if any, or rejected. Each demand unit, when matched to a supply unit, earns a reward that depends on the pair. We study an intermediary whose objective is to match demand and supply units to maximize the total expected reward over a finite horizon.

The above problem is motivated by the rise of two-sided crowdsourcing markets in which demand and supply are dynamically matched by an intermediary firm, which earns a revenue on each transaction. We provide the first efficient algorithms with worst-case performance guarantees for this class of problems. We also prove an upper bound on the best performance guarantee that can be achieved. As part of our analysis, we prove the first Prophet Inequality for arbitrarily correlated non-negative random variables.

1. Introduction
We study a class of dynamic matching problems in which demand and supply units of various types arrive randomly and sequentially over time. Supply units can wait for a fixed amount of time, but demand units must be matched irrevocably upon arrival to existing supply units if any, or rejected. Each demand unit, when matched to a supply unit, earns a reward that depends on the pair. We study an intermediary whose objective is to match demand and supply units to maximize the total expected reward over a finite horizon. We provide the first efficient algorithms with worst-case performance guarantees for this class of problems. We also prove an upper bound on the best performance guarantee that can be achieved.

The problem we study is motivated by the rise of two-sided crowdsourcing markets in which demand and supply are dynamically matched by an intermediary firm, which earns a revenue on each transaction. There are numerous recent examples of two-sided markets. This work is motivated by the marketplaces Upwork, Fiverr, and Freelancer that match providers with customers of professional services. Walmart evaluated a proposal to source its own customers to deliver orders (Barr and Wohl 2013). Enabled by information technology, these crowdsourcing businesses are revolutionizing the traditional marketplace. Freelancers constitute 35% of the U.S. workforce and have generated a trillion dollars in income as of 2015 (Pofeldt 2016). A survey found that 73% of freelancers have found work more easily because of technology (Pofeldt 2016).
The above crowdsourcing economies have been widely studied by economists as two-sided markets (Rysman 2009). In these markets, an intermediary operates a platform to connect the demand side with the supply side, making exchanges possible. For example, Upwork, Fiverr, and Freelancer are all intermediaries. These intermediaries aim to generate revenue from matching demand and supply. In the long term, their revenues depend on the quality of the matches, which depends on characteristics of the two sides and the context. The reward of matching a worker with a project is likely to be higher when the project is more closely related to the worker’s skill set.

In the operation of such economies, a fundamental problem faced by the intermediary is how to match demand with supply when the availability of both may be time-dependent and uncertain. We analyze a basic model that underlies such systems, called the two-sided matching problem. This matching problem is very difficult to solve optimally, due to three main reasons. First, given the many characteristics of both supply and demand types and their importance in determining the quality of a match, the decision problem must keep track of a vast amount of information, including the current state of supply and future demand and supply arrivals. Second, both demand and supply processes may change over time, so that the decision-making environment might be constantly changing. Third, demand units tend to be time-sensitive and unmatched supply units might also leave the system after a time, so that they have a finite period of availability.

In this paper, we take pricing decisions, as well as demand and supply arrivals as exogenous and focus on an intermediary firm’s problem of dynamically matching demand and supply to maximize the total expected reward over a finite horizon. Our contributions are as follow:

- We formulate a new model of dynamic matching with sequential arrivals of both demand and supply units. Supply units can wait a deterministic amount of time, whereas demand units must be matched irrevocably upon arrival. Decisions are not batched and must be made for one demand unit at a time. Arrivals of both demand and supply units may be non-stationary and stochastic.

- We derive the first online algorithm with worst-case performance guarantee for this class of problems. We prove that our algorithm achieves at least a constant fraction of the expected reward of an optimal offline policy, which knows all demand and supply arrivals upfront and makes optimal decisions given this information. We also prove an upper bound on the best constant relative performance guarantee that can be achieved for each step of our algorithm.

- As part of our analysis, we prove a novel variant of a Prophet Inequality. The classical Prophet Inequality arises from a fundamental problem in optimal-stopping theory. In this problem, a gambler sees a finite sequence of independent, non-negative random variables. If he stops the sequence at any time, he collects a reward equal to the most recent observation. The Prophet Inequality states that, knowing the distribution of each random variable, the gambler can achieve at least half as much reward in expectation, as a prophet who knows the entire sample path of random variables.
(Krengel and Sucheston 1978). We prove the first corresponding bound for arbitrarily correlated non-negative random variables and show that this bound is tight. We analyze two methods for proving the bound, a constructive approach, which produces a worst-case instance, and a reductive approach, which characterizes a certain submartingale arising from the reward process of our online algorithm.

2. Literature Review

We review three streams of literature that are most closely related to our problem class.

2.1. Static matching

A variety of matching problems have been studied in static settings, for example, college-admissions problems, marriage problems, and static assignment problems. In these problems, the demand and supply units are known. The reward of matching each demand with each supply unit is also known. The objective is to find a maximum-reward matching. See Abdulkadiroglu and Sonmez (2013) for a recent review. Our setting differs in that demand and supply units arrive randomly over time and decisions must be made before all the units have been fully observed.

2.2. Dynamic assignment

Dynamic-assignment problems are a class of problems in which a set of resources must be dynamically assigned to a stream of tasks that randomly arrive over time. These problems have a long history, beginning with Derman, Lieberman and Ross (1972). See Su and Zenios (2005) for a recent review of this literature. In most of this literature, the resources are known in advance. Only the arrivals of tasks are uncertain. In contrast, in our problem, both the arrivals of demands and supply units are random.

Spivey and Powell (2004) study a version of the dynamic assignment problem in which the resources may arrive randomly over time. They develop approximate-dynamic-programming heuristics for the problem. They do not derive performance bounds for their heuristics.

Anderson et al. (2013) study a specialized model in which supply and demand units are identical, and arrivals are stationary over time. They characterize the performance of the greedy policy under various structures for the demand-supply graph, where the objective is to minimize the total waiting time for all supply units.

Akbarpour et al. (2014) analyze a dynamic matching problem for which they derive several broad insights. They analyze two algorithms for which they derive bounds on the relative performance under various market conditions. Their work differs from ours in four ways. First, they assume that arrivals are stationary whereas we allow non-stationary arrivals. Second, they assume that demand units are identical except for the time of arrivals and supply units are also identical except
for the time of arrival, whereas we allow heterogeneity among the units. Finally, they study an unweighted matching problem in which each match earns a unit reward, whereas we study a more general weighted matching problem. Finally, their results hold in asymptotic regimes, where the market is large and the horizon is long, whereas our results hold in any condition.

More recently Hu and Zhou (2015) study a dynamic assignment problem for two-sided markets similar to ours. They also allow for random, non-stationary arrivals of demand and supply units. They derive structural results for the optimal policy and asymptotic bounds. We depart from both of the above papers in focusing on providing algorithms with theoretical performance guarantees on all problem instances.

Baccara et al. (2015) study a dynamic matching problem in which demand units can wait, and there is a tradeoff between waiting for a higher-quality match, and incurring higher waiting costs. Their setting is limited to just two types of units (demand or supply), whereas we allow arbitrarily many types. They also assume stationary arrivals whereas we allow non-stationary arrivals.

2.3. Online matching

Our work is closely related to works on online matching problems. In all of these problems, the set of available resources is known and corresponds to one set of nodes. Demand requests arrive one by one, and correspond to a second set of nodes. As each demand node arises, its adjacency to the resource nodes is revealed. Each edge has an associated weight. The system must match each demand node irrevocably to an adjacent resource node. The goal is to maximize the total weighted or unweighted size of the matching. In contrast to this literature, we model the uncertainty in arrivals of supply units, which is central to crowdsourcing markets.

The online unweighted bipartite matching problem is originally shown by Karp, Vazirani and Vazirani (1990) to have a worst-case relative reward of 0.5 for deterministic algorithms and \(1 - \frac{1}{e}\) for randomized algorithms. Our work generalizes the online weighted bipartite matching problem. When demands are chosen by an adversary, the worst-case relative reward of this problem cannot be bounded by any constant (Mehta 2012). Many subsequent works have tried to design algorithms with bounded relative reward for this problem for more regulated demand processes.

Three types of demand processes have been studied. The first type of demand processes studied is one in which each demand node is independently and identically chosen with replacement from a known set of nodes. Under this assumption, (Jaillet and Lu 2013, Manshadi et al. 2012, Bahmani and Kapralov 2010, Feldman et al. 2009) propose online algorithms with worst-case relative reward higher than \(1 - \frac{1}{e}\) for the unweighted problem. Haeupler, Mirrokni, Vahab and Zadimoghaddam (2011) study online algorithms with worst-case relative reward higher than \(1 - \frac{1}{e}\) for the weighted bipartite matching problem.
The second type of demand processes studied is one in which the demand nodes are drawn randomly without replacement from an unknown set of nodes. This assumption has been used in the secretary problem (Kleinberg 2005, Babaioff, Immorlica, Kempe, and Kleinberg 2008), ad-words problem (Goel and Mehta 2008) and bipartite matching problem (Mahdian and Yan 2011, Karande, Mehta, and Tripathi 2011).

The third type of demand processes studied is one in which each demand node requests a very small amount of resource. This assumption, called the \textit{small bid} assumption, together with the assumption of randomly drawn demands, lead to polynomial-time approximation schemes (PTAS) for problems such as ad-words (Devanur 2009), stochastic packing (Feldman, Henzinger, Korula, Mirrokni, and Stein 2010), online linear programming (Agrawal, Wang, Zizhuo and Ye 2009), and packing problems (Molinaro and Ravi 2013). Typically, the PTAS proposed in these works use dual prices to make allocation decisions. Devanur, Jain, Sivan, and Wilkens (2011) study a resource-allocation problem in which the distribution of nodes is allowed to change over time, but still needs to follow a requirement that the distribution at any moment induce a small enough offline objective value. They then study the asymptotic performance of their algorithm. In our model, the amount capacity requested by each customer is not necessary small relative to the total amount of capacity available.

Our work builds upon recent results by Wang, Truong and Bank (2015), who propose online algorithms for the bipartite matching problem with heterogeneous demands. They allow demands to arrive according to non-homogenous Poisson processes. They prove that an LP-based policy earns at least half of the expected revenue of an optimal policy that has full hindsight. We study a discrete-time analog of the same demand-arrival processes. In addition, we allow supply units to arrive randomly over time rather than assuming that they are deterministically known.

Methodologically, we depart from Wang, Truong and Bank (2015) in two major ways. The algorithm of Wang, Truong and Bank (2015) consists of two main steps. In the first step, they solve a deterministic assignment LP to find the probabilities of routing each demand to each supply unit. Given this routing, in the second step, they solve a dynamic program to determine whether to match a routed demand unit to a supply unit at any given time. In contrast, in the first step, we find \textit{conditional probabilities} of routing demand to supply units, given the set of supply units that have arrived at any given time. These conditional probabilities are intractable to compute directly and therefore must be approximated. In the second step, after probabilistically routing demands to supply units according to the LP, we cannot simply solve a dynamic program to decide whether to match a given demand unit to a supply unit. The reason is that the stochasticity on the supply side, together with the conditional routing, creates correlation in the demand arrivals. The resulting dynamic program is intractable to solve. Therefore, we propose a novel dynamic threshold-based algorithm for which we prove a bounded performance ratio.
3. Problem Formulation

3.1. Model

There is a finite planning horizon of $T$ periods. There are $I$ types of demand units and $J$ types of supply units. Both demand and supply units arrive randomly over the $T$ periods. The time increments are sufficiently fine that, similar to many standard models in revenue management (van Ryzin and Talluri 2005), we can assume that at most one demand unit arrives in any given period. Similarly, at most one supply unit arrives in any given period.

The demand unit of type $i$, which arrives at time $t$ if any, can be identified using the pair $(i, t)$. Similarly, the supply unit of type $j$, which arrives at time $t$ if any, can be identified with the pair $(j, t)$.

Demand and supply units arrive independently of each other. The probability of seeing a demand arrival $(i, t)$ is $\lambda_i(t)$, $i = 1, \ldots, I$, $t = 1, \ldots, T$. The probability of seeing a supply arrival $(j, t)$ is $\mu_j(t)$, $j = 1, \ldots, J$, $t = 1, \ldots, T$.

Each unit of demand of type $(i, t)$ has a known non-negative reward $r_{ijts}$ when matched with a supply unit of type $(j, s)$. This reward can capture how far apart the units are in time and how compatible their respective types are. If types $i$ and $j$ are incompatible, then the reward $r_{ijts}$ could be very small or 0. If $i$ and $j$ are compatible, then $r_{ijts}$ can decrease with the length of the interval $[s, t]$ to capture the diminishing value of the match when the supply unit must wait for a long time for the demand unit.

We assume that supply units can wait but demand units cannot. At the end of each period $t$, after arrivals of demand and supply units have been observed, the demand unit that arrives in period $t$, if any, must be matched immediately to an existing supply unit or rejected. Thus, $r_{ijts} = 0$ whenever $t < s$. Note that if a supply unit $(j, s)$ can only wait a finite amount of time, then we can require that $r_{ijts} = 0$ for any $t$ that is sufficiently large compared to $s$.

The objective of the problem is to match demand and supply units in an online manner to maximize the expected total reward earned over the horizon. We do not allow fractional matchings. That is, each demand unit must be matched in whole to a supply unit.

3.2. Definition of Competitive Ratios

Define an optimal offline algorithm $OFF$ as an algorithm that knows the entire sample path of demand and supply arrivals at the beginning of period 1 and makes optimal decisions to match demand and supply units given this information. In this paper, we define the competitive ratio of an algorithm as the ratio between the expected reward of an online algorithm and the expected reward of an optimal offline algorithm $OFF$. 
Definition 1. An online algorithm $ON$ is $c$-competitive if its total reward $V_{ON}$ satisfies

$$E[V_{ON}] \geq c E[V_{OFF}],$$

where the expectation is taken over the random arrivals of demand and supply.

Before presenting the two online algorithms, we first characterize an optimal offline algorithm and an upper bound on the optimal offline reward.

3.3. Offline Algorithm and Its Upper Bound

Let $\Lambda_{it} \in \{0, 1\}$ be a random indicator of whether demand unit $(i, t)$ arrives, and $M_{js} \in \{0, 1\}$ a random indicator of whether supply unit $(j, s)$ arrives.

$OFF$ can see the arrivals of demand and supply units $(\Lambda, M)$. Given $(\Lambda, M)$, The maximum offline reward $V_{OFF}(\Lambda, M)$ is equal to the value of the following maximum-weight matching problem.

$$V_{OFF}(\Lambda, M) = \max \sum_{i,j,t,s} x_{ijts}(\Lambda, M) r_{ijts}$$

s.t. $\sum_{i,t} x_{ijts}(\Lambda, M) \leq M_{js}$, for $j = 1, \ldots, J$; $s = 1, \ldots, T$,

$$\sum_{j,s} x_{ijts}(\Lambda, M) \leq \Lambda_{it}$, for $i = 1, \ldots, I$; $t = 1, \ldots, T$,

$$x_{ijts}(\Lambda, M) \leq \Lambda_{it} M_{js}$, for $i = 1, \ldots, I$; $j = 1, \ldots, J$; $t = 1, \ldots, T$; $s = 1, \ldots, T$,

$$x_{ijts}(\Lambda, M) \geq 0$$, for $i = 1, \ldots, I$; $j = 1, \ldots, J$; $t = 1, \ldots, T$; $s = 1, \ldots, T$.

(1)

In the above LP, the variable $x_{ijts}(\Lambda, M)$ indicates whether demand unit $(i, t)$ and supply unit $(j, s)$ both arrive and $(i, t)$ is assigned to $(j, s)$. Although fractional matches are allowed in the LP, it is well known that there exists an optimal solution $x^*(\Lambda, M)$ which is integral because the constraint matrix is totally unimodular and all data are integral. The optimal objective value of the LP is $V_{OFF}(\Lambda, M) = r^T x^*(\Lambda, M)$, where $\tau$ denotes the transpose operator.

Note that (1) cannot be solved without a priori access to the realizations of $(\Lambda, M)$. Thus, we are interested in finding an upper bound on the expected optimal offline reward $E[V_{OFF}(\Lambda, M)]$ when we do not have such a priori access.

For any $S \in \{0, 1\}^{J \times T}$, let $x_{ijts}(S)$ denote the probability that demand unit $(i, t)$ arrives and is matched to supply unit $(j, s)$ conditioned on $M = S$. That is, $x_{ijts}(S) = E[x_{ijts}(\Lambda, M)|M = S]$, where the conditional expectation is taken over $\Lambda$. 
We next show that the following LP (2), in which \( x_{ijts}(S) \)'s are the decision variables, gives an upper bound on LP (1):

\[
\begin{align*}
\text{max} & \quad \sum_{S \in \{0,1\}^{J \times T}} \sum_{i,j,t,s} P(M = S) x_{ijts}(S) r_{ijts} \\
\text{s.t.} & \quad \sum_{i,t} x_{ijts}(S) \leq S_{js}, \quad \text{for } j = 1, \ldots, J; \ s = 1, \ldots, T; \ S \in \{0,1\}^{J \times T}, \\
& \quad \sum_{j,s} x_{ijts}(S) \leq \lambda_{it}, \quad \text{for } i = 1, \ldots, I; \ t = 1, \ldots, T; \ S \in \{0,1\}^{J \times T}, \\
& \quad x_{ijts}(S) \geq 0, \quad \text{for } i = 1, \ldots, I; \ j = 1, \ldots, J; \ t = 1, \ldots, T; \ s = 1, \ldots, T; \ S \in \{0,1\}^{J \times T}, 
\end{align*}
\]

(2)

**Theorem 1.** The optimal objective value of (2) is an upper bound of \( E[V^{OFF}(\Lambda, M)] \).

**Proof.** Since \( \sum_{i,t} x_{ijts}(\Lambda, M) \leq M_{js} \), \( \sum_{j,s} x_{ijts}(\Lambda, M) \leq \Lambda_{it} \), and \( x_{ijts}(\Lambda, M) \leq \Lambda_{it} M_{js} \) are required in (1), we must have

\[
\begin{align*}
\sum_{i,t} x_{ijts}(S) &= \sum_{i,t} E[x_{ijts}(\Lambda, M)|M = S] \leq E[M_{js}|M = S] = S_{js}, \\
\sum_{j,s} x_{ijts}(S) &= \sum_{j,s} E[x_{ijts}(\Lambda, M)|M = S] \leq E[\Lambda_{it}|M = S] = \lambda_{it},
\end{align*}
\]

for every \( S \in \{0,1\}^{J \times T} \).

Thus, \( E[x^*(\Lambda, M)|M = S], \ S \in \{0,1\}^{J \times T} \), is a feasible solution of LP (2). It follows that the optimal value of LP (2) is an upper bound of

\[
\sum_{i,j,t,s} \sum_{S \in \{0,1\}^{J \times T}} P(M = S) E[x^*_{ijts}(\Lambda, M)|M = S] r_{ijts} = r^T E[x^*(\Lambda, M)] = E[V^{OFF}(\Lambda, M)].
\]

\( \square \)

Although the upper bound in (2) is theoretically meaningful, it is impractical because the associated LP has exponentially many variables as a function of the number of supply units. Next, we will consider a cruder LP than (2). This alternative LP provides a looser bound than (2), but it has linearly many variables as a function of the number of supply units.

4. **Upper Bound on LP**

Consider solving for the total probabilities \( x_{ijts} \) of having demand unit \((i, t)\) arrived and assigned to \((j, s)\), rather than the conditional probabilities \( x_{ijts}(S) \) of the same, given the sequence of supply
units $S$ that will arrive over the entire horizon. We express necessary conditions on $x$ in the following LP:

$$\begin{align*}
\text{max} & \quad \sum_{i,j,s,t} x_{ijts} r_{ijts} \\
\text{subject to} & \quad \sum_{i,j,t} x_{ijts} \leq \mu_{js}, \quad \text{for } j = 1, \ldots, J; \quad s = 1, \ldots, T, \\
& \quad \sum_{j,s} x_{ijts} \leq \lambda_{it}, \quad \text{for } i = 1, \ldots, I; \quad t = 1, \ldots, T, \\
& \quad x_{ijts} \leq \lambda_{it} \mu_{js}, \quad \text{for } i = 1, \ldots, I; \quad j = 1, \ldots, J; \quad t = 1, \ldots, T; \quad s = 1, \ldots, T, \\
& \quad x_{ijts} \geq 0, \quad \text{for } i = 1, \ldots, I; \quad j = 1, \ldots, J; \quad t = 1, \ldots, T; \quad s = 1, \ldots, T.
\end{align*}$$

(3)

The constraints above are derived from those of (2).

**Theorem 2.** The optimal objective value of (3) is an upper bound of (2).

**Proof.**

Let $x^*(S)$ be an optimal solution to (2). Then it is easy to verify that

$$E[x_{ijts}^*(M)] = \sum_{S \in \{0,1\}^I \times T} x_{ijts}^*(S) P(M = S)$$

for all $i = 1, \ldots, I$, $j = 1, \ldots, J$, is a feasible solution to (3). \qed

Our algorithms will require a feasible solution to (2). To achieve this, we will construct a solution to (2) from an optimal solution to (3). Note that in general a solution to (2) constructed based on (3) may not be feasible because (3) is a relaxation of (2). In our analysis, we show that our constructed solution is feasible and that this construction does not degrade the quality of the solution by more than a constant factor.

5. Constructing Feasible Solution to the Original LP

Let $x^*$ be an optimal solution to (3). We will show how to construct a feasible solution to (2) from $x^*$. Define a solution to (2) as

$$\hat{x}_{ijts}(S) = \frac{\min(\lambda_{it}, \sum_{j'=1}^J \sum_{t'=1}^t S_{j's'} x_{ij't's'}^*/\mu_{j's'} *)}{\sum_{j'=1}^J \sum_{t'=1}^t S_{j's'} x_{ij't's'}^*/\mu_{j's'} *)} \cdot S_{js} x_{ijts}^*/\mu_{js}.$$  

(4)

**Theorem 3.** $\hat{x}_{ijts}(S)$ is a feasible solution to (2).

**Proof.** By the definition of $\hat{x}_{ijts}(S)$,

$$\sum_{i=1}^I \sum_{t=1}^T \hat{x}_{ijts}(S) = \sum_{i=1}^I \sum_{t=1}^T \frac{\min(\lambda_{it}, \sum_{j'=1}^J \sum_{t'=1}^t S_{j's'} x_{ij't's'}^*/\mu_{j's'} *)}{\sum_{j'=1}^J \sum_{t'=1}^t S_{j's'} x_{ij't's'}^*/\mu_{j's'} *)} \cdot S_{js} x_{ijts}^*/\mu_{js} \leq S_{js}$$

for all $i = 1, \ldots, I$, $j = 1, \ldots, J$, $t = 1, \ldots, T; \quad s = 1, \ldots, T$. 

$\leq S_{js},$
where the last inequality is given by the first constraint of LP (3). Thus, \( \hat{x}_{ijts}(S) \) satisfies the first constraint of (2).

Recall that we require \( r_{ijts} < 0 \) for \( t < s \), so we always have \( x^*_ijts = 0 \) for \( t < s \). We can then derive,

\[
\sum_{j=1}^J \sum_{s=1}^T \hat{x}_{ijts}(S) \\
= \sum_{j=1}^J \sum_{s=1}^T \frac{\min(\lambda_{it}, \sum_{j'=1}^J \sum_{s'=1}^T S_{j's'} \frac{x_{ij't's'}}{\mu_{j's'}})}{\sum_{j'=1}^J \sum_{s'=1}^T S_{j's'} \frac{x_{ij't's'}}{\mu_{j's'}}} \cdot \sum_{j=1}^J \sum_{s=1}^T S_{js} \frac{x_{ijts}}{\mu_{js}} \\
= \frac{\min(\lambda_{it}, \sum_{j'=1}^J \sum_{s'=1}^T S_{j's'} \frac{x_{ij't's'}}{\mu_{j's'}})}{\sum_{j'=1}^J \sum_{s'=1}^T S_{j's'} \frac{x_{ij't's'}}{\mu_{j's'}}} \cdot \sum_{j=1}^J \sum_{s=1}^T S_{js} \frac{x_{ijts}}{\mu_{js}} \\
= \min(\lambda_{it}, \sum_{j'=1}^J \sum_{s'=1}^T S_{j's'} \frac{x_{ij't's'}}{\mu_{j's'}}) \\
\leq \lambda_{it}.
\]

Thus, \( \hat{x}_{ijts}(S) \) satisfies the second constraint of (2). In sum, \( \hat{x}_{ijts}(S) \) is a feasible solution to (2).

\[\square\]

We remark that \( \hat{x}_{ijts}(S) \) does not depend on any arrival information after period \( t \). Therefore, for any \( S \in \{0, 1\}^J \times T \), we can compute \( \hat{x}_{ijts}(S) \) by the end of period \( t \), without observing the arrivals after \( t \).

Next, we show that \( \hat{x} \) is not smaller than a constant factor times \( x^* \). We start with a technical lemma.

**Lemma 1.** For any \( \lambda > 0 \) and \( x > 0 \),

\[\frac{\min(\lambda, x)}{x} \geq 1 - \frac{1}{4\lambda x}.\]

**Proof.** For \( x \leq \lambda \), \( \min(\lambda, x)/x = 1 \geq 1 - \frac{1}{4\lambda x} \).

For \( x > \lambda \),

\[
\frac{\min(\lambda, x)}{x} - (1 - \frac{1}{4\lambda x}) \\
= \frac{\lambda}{x} - 1 + \frac{1}{4\lambda x} \\
= \frac{\lambda}{x} + \frac{1}{4\lambda} \cdot \frac{x}{\lambda} - 1 \\
\geq 2 \cdot \sqrt{\frac{\lambda}{x}} \cdot \frac{1}{2} \cdot \sqrt{\frac{x}{\lambda}} - 1 \\
\geq 0.
\]
Using the above lemma, we are ready to prove the proximity of $\hat{x}$ relative to $x^\ast$.

**Theorem 4.**

$$E[\hat{x}_{ijts}(M)] \geq 0.5x^\ast_{ijts}.$$  

**Proof.** Fix $i,j,t,s$ (but $M$ is random). For any supply unit $(j',s')$, define

$$X_{j's'} = M_{j's'} \cdot x^\ast_{ij'ts}/\mu_{j's'}.$$  

Note that $E[X_{j's'}] = x^\ast_{ij'ts}/\mu_{j's'} \cdot P(M_{j's'} = 1) = x^\ast_{ij'ts}/\mu_{j's'} \cdot \mu_{j's'} = x^\ast_{ij'ts}$.

We can then deduce

$$E[\hat{x}_{ijts}(M)] = E\left[\min(\lambda_{it}, \sum_{j',s'} X_{j's'}) \cdot X_{j's}\right]$$

$$\geq E\left[(1 - \frac{1}{4\lambda_{it}} \sum_{j',s'} X_{j's'}) \cdot X_{j's}\right]$$

(by Lemma 1; if $\sum_{j',s'} X_{j's'} = 0$, we have $X_{j's} = 0$ so the inequality still holds)

$$= E[1 - \frac{1}{4\lambda_{it}} \sum_{j',s'} X_{j's'}] E[X_{j's}] - \frac{1}{4\lambda_{it}} E[X^2_{j's}]$$

$$= E[1 - \frac{1}{4\lambda_{it}} \sum_{j',s'} x^\ast_{ij'ts} x^\ast_{j's}] - \frac{1}{4\lambda_{it}} x^\ast_{ij'ts} \cdot x^\ast_{j's}$$

$$\geq (1 - \frac{1}{4}) x^\ast_{ijts} - \frac{1}{4} x^\ast_{ijts}$$

(because $\sum_{j',s'} x^\ast_{ij'ts} \leq \lambda_{it}$ and $x^\ast_{ijts} \leq \mu_{js} \lambda_{it}$)

$$= \frac{1}{2} x^\ast_{ijts}.$$  

$\square$

6. **Online Algorithm**

In this section, we will describe and analyze a matching algorithm called Online (ON). The algorithm is a composite of two simpler sub-routines, a Separation Subroutine and an Admission Subroutine. The Separation Subroutine produces a feasible solution to LP (2), then routes in a randomized manner all incoming demand units to existing supply units, according to this solution. This routing splits the demand arrivals into separate arrival streams, each coming to a separate supply unit. Subsequently, for each supply unit independently, the Admission Subroutine controls the matching of at most one among all incoming demand units to it.
Let $S_t \equiv \{0,1\}^{J \times t}$. Define $S_t \in S_t$ as the information set that records the arrivals of supply units up to period $t$. That is,

$$S_t = (\{M_{j1}\}_{j=1,2,\ldots,J}, \{M_{j2}\}_{j=1,2,\ldots,J}, \ldots, \{M_{jt}\}_{j=1,2,\ldots,J}).$$

For convenience, let $S_0$ be a dummy constant.

**Online Algorithm (ON):**

- (Initialization) Solve (3) for an optimal solution $x$.
- Upon an arrival of a demand unit $(i,t)$ in period $t$:
  1. (Separation Subroutine) For $j = 1,2,\ldots,J$, $s = 1,2,\ldots,t$, compute $\hat{x}_{itjs}(M)$ using equation (4) (we know $\hat{x}_{itjs}(M)$ by the end of period $t$). Route demand unit $(i,t)$ to supply unit $(j,s)$ independently with probability $\frac{\hat{x}_{itjs}(M)}{\lambda_{it}}$, for $j = 1,2,\ldots,J$, $s = 1,2,\ldots,t$.
  2. (Admission Subroutine) For each supply unit $(j,s)$ outstanding at time $t$, if a demand unit $(i,t)$ is routed to $(j,s)$, then admit $(i,t)$ (i.e., match $(i,t)$ with $(j,s)$) if there is still capacity available and

$$r_{itjs} \geq h_{js}(S_t).$$

Otherwise, reject $(i,t)$. We will specify the function $h_{js}(S_t)$ shortly. (Without loss of generality, we assume that $S_t$, which includes the information of any supply unit that arrives in period $t$, is known at the time of making the admission decision in period $t$.)

Observe that the Admission Subroutine is suboptimal. This is because computing an optimal policy using dynamic programming is in general intractable due to correlation among $\{\hat{x}_{itjs}(M)\}_{t=1}^T$.

The following proposition shows that on every sample path, we do not route more than one unit of demand, on average, to each unit of supply. This property is important in the construction of algorithms using using $\hat{x}$ because it ensures that the routing is balanced. That is, we do not have a large number of demand units being routed to a supply unit in some scenarios, and a very small number of demand units being routed in other scenarios.

**Proposition 1.** Conditioned on $S_T$, the total expected number of demand units routed to each supply unit $(j,s)$ is no more than one.

**Proof.**

$$\sum_{i=1}^I \sum_{t=1}^T \mathbb{E}[\frac{\hat{x}_{itjs}(M)}{\lambda_{it}} | \Lambda_{it} = 1, S_T] \mathbb{P}(\Lambda_{it} = 1)$$

$$= \sum_{i=1}^I \sum_{t=1}^T \mathbb{E}[\frac{\hat{x}_{itjs}(M)}{\lambda_{it}} | S_T] \mathbb{P}(\Lambda_{it} = 1)$$

$$= \sum_{i=1}^I \sum_{t=1}^T \mathbb{E}[\frac{\hat{x}_{itjs}(M)}{\lambda_{it}} | S_T] \lambda_{it}$$
\[ E \left[ \sum_{i=1}^{I} \sum_{t=1}^{T} \hat{x}_{itjs}(M) | S_T \right] \leq E \left[ M_{js} | S_T \right] \]
\[ = M_{js} \leq 1. \]

The first inequality follows from the first constraint of LP (2). □

6.1. Performance of the Admission Subroutine

We will next analyze the performance of the Admission Subroutine. For the rest of the section, we focus on a single supply unit \((j, s)\). For convenience, we will drop \((j, s)\) from notation.

For the rest of the section, we fix a solution \(x^*\) to LP (3), and thus fix the construction of \(\hat{x}\) in equation (4). Let \(X_{it}\) be the random variable indicating whether demand unit \((i, t)\) arrives and is routed to supply unit \((j, s)\). Since the construction of routing probabilities is fixed, we will treat \(X\) in this section as exogenous.

We now specify the threshold function \(h_{js}(S_t)\) that will provide a guaranteed lower bound on the performance of the Admission Subroutine. Let \(p_{it}(S_t) = E[X_{it}|S_t]\) denote the probability that a demand unit \((i, t)\) is routed to the supply unit \((j, s)\) at \(t\), conditioned on \(S_t\). For each scenario \(S_t\), define

\[ h(S_t) = E \left[ \sum_{t'=t+1}^{T} \sum_{i=1}^{I} r_{it'} p_{it'}(S_{t'}) \right] - \sum_{t'=1}^{T} \sum_{i=1}^{I} p_{it'}(S_{t'}) \bigg| S_T = u \]. \tag{5} \]

Note that \(h\) can be computed for each scenario \(S_t\) by simulating the sample paths that potentially arise conditional upon \(S_t\). Another way to write \(h\) that will be useful to our analysis is

\[ h(S_t) = \sum_{u \in S_T} P(S_T = u | S_t) E \left[ \sum_{t'=t+1}^{T} \sum_{i=1}^{I} r_{it'} p_{it'}(S_{t'}) \right] - \sum_{t'=1}^{T} \sum_{i=1}^{I} p_{it'}(S_{t'}) \bigg| S_T = u \]. \tag{6} \]

Let \(h\) denote the Admission Routine above that uses the function \(h\). Conditioned on \(S_t\) and the event that the supply unit \((j, s)\) has not been matched by the beginning of period \(t\), let \(V^h(S_t)\) be the expected reward earned from matching \((j, s)\) with a demand unit during periods from \(t\) to \(T\). The definition holds even if \((j, s)\) has not arrived. We can express \(V^h(S_t)\) explicitly by the following recursion:

\[ V^h(S_t) = \sum_{i=1}^{I} p_{it}(S_t) 1(r_{it} \geq h(S_t)) (r_{it} - E[V^h(S_{t+1}) | S_t]) + E[V^h(S_{t+1}) | S_t], \tag{7} \]

and \(V^h(S_{T+1}) = 0\).

We will prove a constant lower bound on a ratio involving \(V^h(S_t)\), which will later help us to prove a constant bound the performance of \(ON\) with respect to \(OFF\).
In order to determine a worst-case lower bound on (11), we will examine all possible combinations of problem data, including all probability processes \( p_{it}(S_t) \) and reward values \( r_{it} \), for all \( S_t, t = 1, \ldots, T \). We analyze two methods for proving the performance bound, a constructive approach, which produces a worst-case instance, and a reductive approach, which characterizes a certain submartingale arising from the reward process of \( ON \).

6.1.1. Constructive method for proving performance bound. With this method, we fix a starting instance of problem data. We will transform this data progressively, each time making the expected total reward of \( h \) smaller on this instance. Then we will show that at some point, the expected total reward of \( h \) is easily bounded below by a constant. We will work with a tree representation of the stochastic process \( S_1, S_2, \ldots, S_T \). Each node in this tree corresponds to a scenario \( S_t \). The root of the tree is the distinguished node \( S_1 \). We call a node \( S_t \) in the tree a terminal node if \( p_{it}(S_t) > 0 \) for some \( i \) but \( p_{it'}(S_{t'}) = 0 \) for all \( i = 1, 2, \ldots, I \) and all descendants \( S_{t'} \) of \( S_t \).

For the chosen problem instance, we fix

\[
\sum_{t=1}^{T} \sum_{i=1}^{I} E[p_{it}(S_i)|S_1] = c \leq 1, \tag{8}
\]

where the inequality is guaranteed by Proposition 1.

We will arrive at our bound by proving two sets of structural results for the worst-case instance of the problem. The first set of structural results concern the reward process.

**Lemma 2.** Assume that the given data instance achieves the worst-case ratio \( V^h/V^{OFF} \). Then without loss of generality,

1. \( E[\sum_{t'=1}^{T} \sum_{i=1}^{I} p_{it'}(S_{t'})|S_T] = 1 \) for each \( S_T \in S_T \).
2. The reward is scenario dependent. That is, demand unit \((i, t)\) has reward \( r_{it}(S_i) \) at each scenario \( S_t \).
3. Each scenario \( S_i \) has at most one arrival with probability \( p(S_i) \) and reward \( r(S_i) \).
4. \( r(S_i) = h(S_i) \) or \( r(S_i) = (h(S_i))^− \), or is just below \( h(S_i) \).

**Proof.** The first property is easy to see, since by Proposition 1, we have

\[
E[\sum_{t'=1}^{T} \sum_{i=1}^{I} p_{it'}(S_{t'})|S_T] \leq 1
\]

for each \( S_T \in S_T \). We can always add nodes with 0 reward and positive arrival probabilities to paths in the tree to ensure that the property holds. This transformation does not change the outcome of either \( ON \) or \( OFF \).
By adding demand types if necessary, we can assume without loss of generality that the reward is scenario dependent. That is, demand unit \((i, t)\) has reward \(r_{it}(S_t)\) at each scenario \(S_t\).

Assuming that the arrival probabilities in each period are very small, we can split up each period into several periods if necessary, such that each scenario \(S_t\) has at most one arrival with probability \(p(S_t)\) and reward \(r(S_t)\). This change preserves the expected reward of \(OFF\) and decreases the expected reward for \(h\) according to (7), if the rewards are chosen to be increasing with time.

Moving from the root \(S_1\) downward in the tree, if there is some highest-level non-terminal node \(S_t\) for which \(r(S_t) > h(S_t)\) then conditioned on \(S_t\), we can decrease \(r(S_t)\) and scale up \(r(S_{t'}')\) by the same factor for all scenarios \(S_{t'}\) that descend from \(S_t\), such that the value of

\[
\sum_{t'=t}^{T} E[r(S_{t'})p(S_{t'})|S_t]
\]

is unchanged. As a result, the equality in (10) is maintained. Do this until \(r(S_t) = h(S_t)\).

We claim that this change decreases \(V^h(S_t)\), hence \(V^h(S_1)\). To see the claim, note that according to (7), the change reduces the immediate reward given \(S_t\) by some amount \(\Delta\) and increases the future reward given \(S_t\) by no more than \(\Delta\). Thus the net effect is to reduce \(V^h(S_t)\).

Also, since the value of \(h(S_s)\) and the rewards stay the same for every node \(S_s\) preceding \(S_t\), \(V^h(S_1)\), is reduced.

Moving from the root \(S_1\) downward in the tree, if there is some highest-level non-terminal node \(S_t\) for which \(r(S_t) < h(S_t)\) then conditioned on \(S_t\), we can increase \(r(S_t)\) by a small amount and scale down \(r(S_{t'}')\) by the same factor for all scenarios \(S_{t'}\) descending from \(S_t\), such that the equality in (10) is maintained and all rewards remain non-negative. Do this until \(r(S_t) = (h(S_t))^{-}\), where \((h(S_t))^{-}\) denotes a value infinitesimally smaller than \(h(S_t)\). It is easy to see that this change decreases \(V^h(S_t)\). Hence \(V^h(S_1)\) is decreased as we argued just above. Repeat the previous transformations until at all non-terminal nodes \(S_t\), we have \(r(S_t) = h(S_t)\) or \(r(S_t) = (h(S_t))^{-}\). □

We will assume for the rest of the subsection that our worst-case data has the structure imposed by Lemma 2. Therefore, for the rest of this subsection, we write the threshold function \(h\) in the following alternative way

\[
h(S_t) = E \left[ \frac{\sum_{t'=t+1}^{T} \sum_{i=1}^{I} r_{it'}p_{it'}(S_{t'})}{1 + \sum_{t'=t+1}^{T} \sum_{i=1}^{I} p_{it'}(S_{t'})} \mid S_t \right].
\] (9)

Our second set of structural results concern the arrival probabilities.

**Lemma 3.** Assume that the data achieves the worst-case ratio \(V^h/V^{OFF}\). Without loss of generality, for \(T \geq 2\), the followings hold:

1. There is a unique path \(S_1, S_2, \ldots, S_T\) with positive arrival probabilities.
2. At every node $S_t$, $t = 1, \ldots, T - 1$, $r(S_t) = h(S_T) = E[r(S_T)p(S_T)|S_t]$;
3. $p(S_t) = 0$, $t = 2, \ldots, T$;
4. $V^h(S_1) \geq \frac{R}{2}$.

Proof. We will prove the theorem by induction on $T$. First, we prove the result for $T = 2$. By the tree reward simplifications,

$$r(S_1) \approx h(S_1) = \sum_{u \in S_2} P(S_2 = u|S_1) \frac{r(u)p(u)}{1 + p(u)} \leq \sum_{u \in S_2} P(S_2 = u|S_1)r(u)p(u) = E[V^h(S_2)|S_1].$$

Therefore $r(S_1) = h(S_1)$ to make $V^h(S_1)$ as small as possible. Define

$$c_2 = \sum_u P[S_2 = u]p(u),
R_2 = \sum_u P[S_2 = u]p(u)r(u),
R = R_2 + p(S_1)r(S_1),
c = c_2 + p(S_1).$$

Fix $R$, $R_2$, and $c$. Consider what happens when we scale down $c_2$ by a factor $\alpha$, scale up $r(u)$ for leaf nodes $u$ to maintain $R_2$ constant, and scale down $r(S_1)$ to maintain $p(S_1)r(S_1) = R - R_2$ constant. We argue that we will reduce $V^h(S_1)$ while keeping $V^{OFF} = R$ constant. Indeed, for $\alpha = 1$,

$$p(S_1) = c - \alpha c_2.$$  

This implies that

$$V^h(S_1) = p(S_1)r(S_1) + (1 - p(S_1))\sum_u P[S_2 = u]p(u)r(u),
= R - R_2 + (1 - c + \alpha c_2)R_2.$$  

Hence,

$$\frac{\partial V^h(S_1)}{\partial \alpha} = c_2 R_2
\geq 0.$$  

Therefore, $V^h/V^{OFF}$ is minimized when $\alpha = 0$, or $p(S_2) = 0$ for all $S_2 \in S_2$. Therefore, the base case is proved.

Assume the theorem holds for $T - 1$. Fix $c^+ = \sum_{t=2}^{T} P(S_t|S_1)p(S_t)$ and $R^+ = \sum_{t=2}^{T} P(S_t|S_1)p(S_t)r(S_t)$. Let the immediate successors of $S_1$ be $S_2^k$, $k = 1, \ldots, K$. Let

$$R^k = E[\sum_{t=2}^{T} r(S_t)p(S_t)|S_2^k],$$
for $k = 1, \ldots, K$. Since the instance that minimizes $V^h(S_1)$ must minimize $\mathbb{E}[V^h(S_2)|S_1]$ subject to $c^+$ and $R^+$, we have by the induction hypothesis, that

$$\mathbb{E}[V^h(S_2)|S_1] \geq \sum_{1}^{K} P(S_2^k) \frac{R^k}{2} = \frac{R^+}{2}.$$  

This lower bound is attained when $K = 1$, $\sum_{t=2}^{T} P(S_t|S_2^k)p(S_i) = 1$, $p(S_t) = 0$ for all $t = 3, \ldots, T$, and $P(S_2^K|S_1) = c - p(S_1)$.

By the induction hypothesis, $V^h(S_2) = \mathbb{E}[p(S_T)r(S_T)|S_2] = h(S_1)$. We also know that $r(S_1) \approx h(S_1)$ by the tree reward simplifications, we conclude that $r(S_1) = h(S_1)$, since the impact on $V^h(S_1)$ is the same in either case. Thus,

$$V^h(S_1) = \mathbb{E}[p(S_T)r(S_T)|S_1].$$

We know that

$$R = p(S_T)r(S_T)(P(S_T|S_1) + \sum_{s=1}^{T-1} P(S_s|S_1)P(S_T|S_s)p(S_s))$$

$$= p(S_T)r(S_T)(P(S_T|S_1) + P(S_2^K|S_1)P(S_T|S_2^K)p(S_2^K) + P(S_T|S_1)p(S_1))$$

$$= p(S_T)r(S_T)P(S_T|S_1)(1 + p(S_2^K) + p(S_1)).$$

This implies that

$$V^h(S_1) = p(S_T)r(S_T)P(S_T|S_1)$$

$$= \frac{R}{1 + p(S_2^K) + p(S_1)}$$

$$\geq \frac{R}{2},$$

with the lower bound being realizable when $p(S_2^K) = 0$ and $p(S_1) = 1$. By induction, the lemma holds for all $T$. □

Lemmas 2 and 3 combine to give us Theorem 5 directly:

**Theorem 5.** Assume that

$$\sum_{i=1}^{T} \sum_{i=1}^{I} \mathbb{E}[r_{it}p_{it}(S_i)|S_1] = R. \quad (10)$$

Then

$$V^h(S_1) \geq \frac{R}{2}. \quad (11)$$
6.1.2. Reductive martingale method for proving performance bound. Define a stochastic process \( \{Z(S_t)\}_{t \geq 0} \) as

\[
Z(S_t) = h(S_t) + \sum_{t'=1}^{t} \sum_{i=1}^{I} p_{it'}(S_{t'}) (r_{it'} - h(S_{t'}))^+.
\]

Define \( \tau \geq 1 \) as the random period in which a customer is admitted to the resource. If no customer is admitted to the resource during periods \( \{1, 2, ..., T\} \), set \( \tau = T + 1 \). In other words, \( \tau \) is a stopping time bounded by \( T + 1 \). We naturally define \( r_{i,T+1} = 0 \) and \( p_{i,T+1}(S_{T+1}) = 0 \) for all \( i \).

**Proposition 2.** Under algorithm ON, the expected reward of matching a customer to the fixed resource is \( E[Z(S_{\tau})] \).

**Proof.** Recall that in each period, at most one demand unit can arrive. In other words, \( \sum_{i=1}^{I} \Lambda_{it} \leq 1 \) and hence \( \sum_{i=1}^{I} X_{it} \leq 1 \) for all \( t = 1, 2, ..., T \).

For any \( t \in \{1, 2, ..., T\} \), conditioned on \( \tau = t \), exactly one customer is routed to the resource in period \( t \), so we must have \( \sum_{i=1}^{I} X_{it} = 1 \). Moreover, conditioned on \( \tau = t \leq T \), the customer type \( i \) admitted in period \( t \) must satisfy the admission condition \( r_{it} \geq h(S_t) \), or more precisely,

\[
\sum_{i=1}^{I} X_{it} r_{it} \geq h(S_t).
\]

Noting that only the admitted customer type \( i \) satisfies \( X_{it} = 1 \), we can write

\[
\sum_{i=1}^{I} X_{it}(r_{it} - h(S_t)) = \sum_{i=1}^{I} X_{it}(r_{it} - h(S_t))^+.
\]

The expected reward of the algorithm is

\[
E \left[ \sum_{i=1}^{I} \sum_{t=1}^{T} X_{it} r_{it} \cdot 1(\tau = t) \right] = E \left[ \sum_{i=1}^{I} \left( \sum_{t=1}^{T} X_{it}(r_{it} - h(S_t)) + \sum_{i=1}^{I} X_{it} h(S_t) \right) \cdot 1(\tau = t) \right] = E \left[ \sum_{i=1}^{I} \sum_{t=1}^{T} X_{it}(r_{it} - h(S_t)) + h(S_t) \cdot 1(\tau = t) \right] = E \left[ \sum_{i=1}^{I} \sum_{t=1}^{T} X_{it}(r_{it} - h(S_t))^+ + h(S_t) \cdot 1(\tau = t) \right] = E[h(S_{\tau})].
\]

If \( \tau > t \), we must have \( \sum_{i=1}^{I} X_{it}(r_{it} - h(S_t))^+ = 0 \), because any customer routed to the resource in period \( t \) should not satisfy the admission condition. Thus we have

\[
\sum_{i=1}^{I} X_{it}(r_{it} - h(S_t))^+ \cdot 1(\tau > t) = 0
\]
\[ \implies \sum_{i=1}^{l} X_{it}(r_{it} - h(S_i))^+ \cdot 1(\tau = t) = \sum_{i=1}^{l} X_{it}(r_{it} - h(S_i))^+ \cdot 1(\tau \geq t). \]

Then the expected reward can be further written as

\[
\mathbb{E}\left[ \sum_{t=1}^{T} \sum_{i=1}^{l} X_{it}(r_{it} - h(S_i))^+ \cdot 1(\tau = t) \right] + \mathbb{E}[h(S)]
\]

\[
= \mathbb{E}\left[ \sum_{t=1}^{T} \sum_{i=1}^{l} X_{it}(r_{it} - h(S_i))^+ \cdot 1(\tau \geq t) \right] + \mathbb{E}[h(S)]
\]

\[
= \mathbb{E}\left[ \sum_{t=1}^{T} \mathbb{E}\left[ \sum_{i=1}^{l} X_{it}(r_{it} - h(S_i))^+ \cdot 1(\tau \geq t) \mid S_t, \{X_{i'v'}\}_{v'=1,2,...,t',t'=1,2,...,t-1} \right] \right] + \mathbb{E}[h(S)]
\]

\[
= \mathbb{E}\left[ \sum_{t=1}^{T} \sum_{i=1}^{l} X_{it}(r_{it} - h(S_i))^+ | S_t, \{X_{i'v'}\}_{v'=1,2,...,t',t'=1,2,...,t-1} \right] \cdot 1(\tau \geq t) \right] + \mathbb{E}[h(S)]
\]

\[ \text{(the event } \tau \geq t \text{ depends only on the information from periods 1 to } t - 1) \]

\[
= \mathbb{E}\left[ \sum_{t=1}^{T} \mathbb{E}\left[ \sum_{i=1}^{l} X_{it}(r_{it} - h(S_i))^+ | S_t \right] \right] \cdot 1(\tau \geq t) \right] + \mathbb{E}[h(S)]
\]

\[
= \mathbb{E}\left[ \sum_{i=1}^{l} p_i(r_{it} - h(S_i))^+ \right] \cdot 1(\tau \geq t) \right] + \mathbb{E}[h(S)]
\]

\[
= \mathbb{E}\left[ Z(S) \right].
\]

\[ \square \]

**Lemma 4.** For any \( b \geq 1, a \geq 0, r_1, \ldots, r_n \geq 0 \) and \( p_1, \ldots, p_n \geq 0 \),

\[ \frac{a + \sum_{i=1}^{n} p_i r_i}{b + \sum_{i=1}^{n} p_i} \leq \frac{a}{b} + \sum_{i=1}^{n} p_i (r_i - \frac{a}{b})^+. \]

**Proof.** Let \( I \subseteq \{1, 2, \ldots, n\} \) be the set such that \( r_i \geq a/b \) for all \( i \in I \).

\[ \frac{a}{b} + \sum_{i=1}^{n} p_i (r_i - \frac{a}{b})^+ \]

\[ = \frac{a}{b} + \sum_{i \in I} p_i (r_i - \frac{a}{b}) \]

\[ = (\frac{a}{b} + \sum_{i \in I} p_i (r_i - \frac{a}{b})) \left( b + \sum_{i \in I} p_i \right) \]

\[ = a + \sum_{i \in I} p_i r_i + (b - 1 + \sum_{i \in I} p_i)((\sum_{i \in I} p_i)(r_i - a/b)) \]

\[ \geq \frac{a + \sum_{i \in I} p_i r_i}{b + \sum_{i \in I} p_i} \]

\[ \geq \frac{a + \sum_{i=1}^{n} p_i r_i}{b + \sum_{i=1}^{n} p_i}. \]
The last inequality follows from the fact that for any $j \notin I$,

$$r_j < \frac{a}{b} \leq \frac{a + \sum_{i \in I} p_ir_i}{b + \sum_{i \in I} p_i}.$$ 

**Proposition 3.** The process \( \{Z(S_t)\}_{t \geq 0} \) is a sub-martingale with respect to \( S_t \).

**Proof.**

\[
\mathbb{E}[Z(S_t)|S_{t-1}] = \mathbb{E}[h(S_t) + \sum_{t'=1}^t \sum_{i=1}^l p_{it'}(S_{t'}) (r_{it'} - h(S_{t'}))^+] | S_{t-1}]
\]

\[
= \mathbb{E}[h(S_t) - h(S_{t-1}) + \sum_{i=1}^l p_{it}(S_t)(r_{it} - h(S_t))^+] | S_{t-1}] + Z(S_{t-1}).
\]

It suffices to prove that in expectation, \( h(S_{t-1}) \leq h(S_t) + \sum_{i=1}^l p_{it}(S_t)(r_{it} - h(S_t))^+ \). We can derive

\[
\begin{align*}
&\mathbb{E}[h(S_{t-1})] \\
= &\mathbb{E} \left[ \frac{\sum_{t'=1}^t \sum_{i=1}^l r_{it'} p_{it'}(S_{t'})}{2 - \sum_{t'=1}^t \sum_{i=1}^l p_{it'}(S_{t'})} \mid S_{t-1} \right] \\
= &\mathbb{E} \left[ \frac{\sum_{t'=1}^t \sum_{i=1}^l r_{it'} p_{it'}(S_{t'}) + \sum_{i=1}^l r_{it} p_{it}(S_{t})}{2 - \sum_{t'=1}^t \sum_{i=1}^l p_{it'}(S_{t'}) + \sum_{i=1}^l p_{it}(S_{t})} \mid S_{t-1} \right] \\
= &\mathbb{E} \left[ \frac{\sum_{t'=1}^t \sum_{i=1}^l r_{it'} p_{it'}(S_{t'})|S_t| + \sum_{i=1}^l r_{it} p_{it}(S_{t})}{2 - \sum_{t'=1}^t \sum_{i=1}^l p_{it'}(S_{t'}) + \sum_{i=1}^l p_{it}(S_{t})} \mid S_{t-1} \right] \\
\leq &\mathbb{E} \left[ \frac{\sum_{t'=1}^t \sum_{i=1}^l r_{it'} p_{it'}(S_{t'})|S_t|}{2 - \sum_{t'=1}^t \sum_{i=1}^l p_{it'}(S_{t'})} + \sum_{i=1}^l p_{it}(S_{t}) \left( r_{it} - \mathbb{E} \left[ \frac{\sum_{t'=1}^t \sum_{i=1}^l r_{it'} p_{it'}(S_{t'})|S_t|}{2 - \sum_{t'=1}^t \sum_{i=1}^l p_{it'}(S_{t'})} \mid S_{t-1} \right] \right)^+ \mid S_{t-1} \right] \\
= &\mathbb{E} \left[ \frac{\sum_{t'=1}^t \sum_{i=1}^l r_{it'} p_{it'}(S_{t'})|S_t|}{2 - \sum_{t'=1}^t \sum_{i=1}^l p_{it'}(S_{t'})} + \sum_{i=1}^l p_{it}(S_{t}) \left( r_{it} - \mathbb{E} \left[ \frac{\sum_{t'=1}^t \sum_{i=1}^l r_{it'} p_{it'}(S_{t'})|S_t|}{2 - \sum_{t'=1}^t \sum_{i=1}^l p_{it'}(S_{t'})} \mid S_{t-1} \right] \right)^+ \mid S_{t-1} \right] \\
= &\mathbb{E} \left[ h(S_t) + \sum_{i=1}^l p_{it}(S_{t}) (r_{it} - h(S_t))^+ \mid S_{t-1} \right].
\]

The last inequality follows from Lemma 4. \( \square \)

With Proposition 3 established, we apply the optional stopping theorem, to obtain Theorem 5:

\[
\mathbb{E}[Z(S_r)] \geq \mathbb{E}[Z(S_0)] = \mathbb{E} \left[ \frac{\sum_{t'=1}^r \sum_{i=1}^l r_{it'} p_{it'}(S_{t'})}{2} \mid S_0 \right] = R/2.
\]
6.2. Performance of the online algorithm.

Now, we tie together the approximation bounds for both the Separation and Admission subroutines to obtain a competitive ratio for our online algorithm $ON$:

**Theorem 6.** The expected total reward $V^{ON}$ of $ON$ satisfies
\[
V^{ON} \geq \frac{1}{4} V^{OFF}.
\]

*Proof.* Fix any supply unit $(j, s)$. Then the reward of $OFF$ for the supply unit is $R$. Assume that the random routing step results in a total arrival probability of $c'$ over the horizon to $(j, s)$ and a corresponding total expected reward of $R' \geq \frac{R}{2}$. Then by Theorem 5, the total expected reward of $V^{ON}$ for $(j, s)$ is at least $\frac{R'}{2} \geq \frac{R}{4}$. Repeat this argument over all supply units to obtain a competitive ratio of $\frac{1}{4}$ for the online algorithm $ON$.

6.3. Upper Bound on the Competitive Ratios of the Subroutines

In the above analysis we have shown that $\frac{1}{2}$ is a lower bound of the best competitive ratio of our Admission subroutine $h$. Next, we show that $1/2$ is an upper bound on the competitive ratio of any admission-control subroutine. That is,

**Theorem 7.** The competitive ratio of any admission-control algorithm is at most 0.5.

*Proof.* The following example is a discrete-time analogy of the example provided by Wang, Truong and Bank (2015) to show that the competitive ratio for any algorithm is 0.5 in the case that the supply units are known with certainty. Clearly, the known-supply case is a special case of our model.

Consider a situation in which a single supply unit is available to be allocated. There are two classes of customers who want to be matched to that resource.

- Class-1 customers arrive in periods $[1, \ldots, T - 1]$. Their arrival rate is very large in these periods. In particular, $\Lambda_1 = \sum_{t=1}^{T-1} \lambda_{1t} \gg 1$, so that we can ignore the event that no class-1 customer arrives. Their reward is $r_1 = 1$.

- Class-2 customers arrive in period $T$. They have a very small arrival rate. In particular, $\Lambda_2 = \lambda_{2T} \ll 1$. Their reward is $r_2 = 1/\Lambda_2 \gg 1$.

Since $r_2 \gg r_1$, the offline algorithm will allocate the resource to a class-2 customer, if there is one. The probability that at least one class-2 customer arrives is $\Lambda_2$. With probability $1 - \Lambda_2$, no class-2 customer will arrive, in which case the optimal offline algorithm will assign a class-1 customer (there are plenty of class-1 customers) and earn reward $r_1 = 1$. In sum, the expected total offline reward is
\[
r_2 \Lambda_2 + r_1 (1 - \Lambda_2)
\]
\[ \Lambda_2 \cdot \frac{1}{\Lambda_2} + 1 \cdot (1 - \Lambda_2) = 2 - \Lambda_2. \]

The decision of an online algorithm is whether to allocate the resource to a class-1 customer during the first \( T - 1 \) periods. If it does allocate the resource to a class-1 customer, the online algorithm earns reward \( r_1 = 1 \). Otherwise, with probability \( \Lambda_2 \) it earns \( r_2 \), which equals 1 in expectation. In sum, the expected reward obtained by an online algorithm cannot exceed \( 1 + \Lambda_2 \). Thus, an upper bound of the competitive ratio is

\[ \frac{(1 + \Lambda_2)}{(2 - \Lambda_2)}, \]

which tends to 0.5 in the limit as \( \Lambda_2 \to 0 \).

\[ \square \]

7. Numerical Studies

In this section, we conduct numerical experiments to explore the performance of our algorithms. We model our experiments on applications that match employers with freelancers for short-term projects, such as web design, art painting, and data entry.

We assume there are 30 employer (demand) types and 30 worker (supply) types. We set the reward \( r_{ijts} \) according to the formula

\[ r_{ijts} = s_{ij} \cdot f_{ts} \cdot g_{ij}, \]

where we use \( s_{ij}, f_{ts} \) and \( g_{ij} \) to capture three different aspects of a matching:

- **Ability to accomplish tasks.** \( s_{ij} \) represents the ability of workers of type \( j \) to work for employers of type \( i \). We randomly draw \( s_{ij} \) from a normal distribution \( \mathcal{N}(0, 1) \) for each pair \( (i, j) \). In particular, if \( s_{ij} < 0 \), the reward of the matching will be negative, and thus no algorithm will ever match worker type \( j \) to employer type \( i \).

- **Idle time of workers.** It may be wise to limit the total time that a worker is idle in the system before being assigned a job. Thus, we set

\[ f_{ts} = 1 - \alpha + \alpha e^{-(t-s)/\tau} \]

so that the reward of a matching is discounted by \( \alpha \) when the idle time of the worker exceeds \( \tau \).

- **Geographical distance.** Certain freelance jobs may require a short commute distance between workers and employers. For demonstration purpose, we assume that each worker type and employer type is associated with a random zip code in Manhattan, with probability proportional to the total population in the zip code zone. Let \( d(i, j) \) be the Manhattan distance between the centers of zip code zones of worker type \( j \) and employer type \( i \). We assume that

\[ g_{ij} = 1 - \beta + \beta e^{-d(i, j)/\omega}, \]

so the reward is discounted by \( \beta \) when the commute distance exceeds \( \omega \).
We consider a horizon of 60 periods. Depending on the application, one period may correspond to a day or a 10-minute span. In any period, a random number of workers may sign in to be ready to provide service. The type of a worker is uniformly drawn from all the worker types. Let \( \mu(t) \) be the rate at which workers appear in the system. Similarly, when an employer arrives, the type of the employer is uniformly drawn from all the employer types. Let \( \lambda(t) \) be the arrival rate of employers.

We randomly generate multiple test scenarios. In each scenario, we independently draw \( \mu(t) \) and \( \lambda(t) \), for every period \( t \), from a uniform distribution over \([0, 1]\). Given the rates \( \mu(t) \) and \( \lambda(t) \), we further vary other model parameters by first choosing the base case to be \( \alpha = 0.5, \tau = 10 \) (periods), \( \beta = 0.5, \omega = 0.05^\circ \), and then each time varying one of these parameters.

We test the following algorithms:

- \((ON)\) Our online algorithm without resource sharing.
- \((ON_+^1)\) A variant of our online algorithm with resource sharing. When \( ON_+^1 \) rejects a customer in the admission subroutine, \( ON_+^1 \) offers another resource with the largest non-negative margin

\[
E \left[ \frac{\sum_{\tau'=(t+1)}^T \sum_{i=1}^I r_{i\tau'} p_{i\tau'}(S_{\tau'})}{2 - \sum_{\tau'=(t+1)}^T \sum_{i=1}^I p_{i\tau'}(S_{\tau'})} \right] - r_{ij,t}.
\]

- \((ON_+^2)\) A variant of our online algorithm with resource sharing. When \( ON_+^2 \) rejects a customer in the admission subroutine, \( ON_+^2 \) offers another resource with the largest non-negative margin

\[
130\% \times E \left[ \frac{\sum_{\tau'=(t+1)}^T \sum_{i=1}^I r_{i\tau'} p_{i\tau'}(S_{\tau'})}{2 - \sum_{\tau'=(t+1)}^T \sum_{i=1}^I p_{i\tau'}(S_{\tau'})} \right] - r_{ij,t}.
\]

- \((ON_+^3)\) A variant of our online algorithm with resource sharing. When \( ON_+^3 \) rejects a customer in the admission subroutine, \( ON_+^3 \) offers another resource with the largest non-negative margin

\[
160\% \times E \left[ \frac{\sum_{\tau'=(t+1)}^T \sum_{i=1}^I r_{i\tau'} p_{i\tau'}(S_{\tau'})}{2 - \sum_{\tau'=(t+1)}^T \sum_{i=1}^I p_{i\tau'}(S_{\tau'})} \right] - r_{ij,t}.
\]

- \((ON_+^4)\) A variant of our online algorithm with resource sharing. When \( ON_+^4 \) rejects a customer in the admission subroutine, \( ON_+^4 \) offers another resource with the largest non-negative margin

\[
200\% \times E \left[ \frac{\sum_{\tau'=(t+1)}^T \sum_{i=1}^I r_{i\tau'} p_{i\tau'}(S_{\tau'})}{2 - \sum_{\tau'=(t+1)}^T \sum_{i=1}^I p_{i\tau'}(S_{\tau'})} \right] - r_{ij,t}.
\]

- A greedy algorithm that always offers a resource with the highest reward.
- A bid-price heuristic based on the optimal dual prices of LP (1)

We report numerical results in Tables 1 to 4, where the performance of each algorithm is simulated using 1000 replicates. For our online algorithms, in each period we compute the threshold \( h_{js}(S_t) \) by simulating 100 future sample paths. We find that, despite the 1/4 provable ratio, the
algorithm $ON$ captures about half of the offline expected reward, and the improved algorithms $ON_1^+$ and $ON_2^+$ capture 65% to 70% of the offline expected reward. Moreover, the improved algorithms outperform the greedy and the bid-price heuristics in all scenarios. These results demonstrate the advantage of using our online algorithms as they have not only optimized performance in the worst-case scenario, but satisfactory performance on average as well.
Table 3 Scenario 3. Performance of different algorithms relative to LP (1).

<table>
<thead>
<tr>
<th></th>
<th>ON</th>
<th>Greedy</th>
<th>BPH</th>
<th>(ON_+)</th>
<th>(ON_{+}^+)</th>
<th>(ON_{+}^A)</th>
<th>(ON_{+}^A)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Base</td>
<td>51.2%</td>
<td>63.5%</td>
<td>65.2%</td>
<td>67.0%</td>
<td>68.2%</td>
<td>67.7%</td>
<td>67.1%</td>
</tr>
<tr>
<td>(\alpha = 0)</td>
<td>48.6%</td>
<td>58.1%</td>
<td>65.8%</td>
<td>66.7%</td>
<td>68.2%</td>
<td>67.7%</td>
<td>66.8%</td>
</tr>
<tr>
<td>(\alpha = 0.2)</td>
<td>49.7%</td>
<td>60.2%</td>
<td>66.1%</td>
<td>66.8%</td>
<td>68.1%</td>
<td>67.6%</td>
<td>66.9%</td>
</tr>
<tr>
<td>(\alpha = 0.8)</td>
<td>53.2%</td>
<td>65.9%</td>
<td>63.8%</td>
<td>66.3%</td>
<td>67.6%</td>
<td>67.1%</td>
<td>66.6%</td>
</tr>
<tr>
<td>(\alpha = 1)</td>
<td>54.2%</td>
<td>66.2%</td>
<td>64.7%</td>
<td>65.4%</td>
<td>66.5%</td>
<td>66.0%</td>
<td>65.6%</td>
</tr>
<tr>
<td>(\tau = 2)</td>
<td>52.1%</td>
<td>64.9%</td>
<td>68.1%</td>
<td>68.2%</td>
<td>69.5%</td>
<td>69.0%</td>
<td>68.5%</td>
</tr>
<tr>
<td>(\tau = 5)</td>
<td>52.0%</td>
<td>64.8%</td>
<td>66.7%</td>
<td>67.4%</td>
<td>68.7%</td>
<td>68.2%</td>
<td>67.6%</td>
</tr>
<tr>
<td>(\tau = 20)</td>
<td>50.4%</td>
<td>61.9%</td>
<td>64.7%</td>
<td>66.6%</td>
<td>68.1%</td>
<td>67.4%</td>
<td>66.9%</td>
</tr>
<tr>
<td>(\tau = 30)</td>
<td>50.1%</td>
<td>61.0%</td>
<td>64.9%</td>
<td>66.7%</td>
<td>68.0%</td>
<td>67.5%</td>
<td>66.9%</td>
</tr>
<tr>
<td>(\beta = 0)</td>
<td>51.0%</td>
<td>64.4%</td>
<td>65.3%</td>
<td>67.4%</td>
<td>68.9%</td>
<td>68.2%</td>
<td>67.6%</td>
</tr>
<tr>
<td>(\beta = 0.2)</td>
<td>51.4%</td>
<td>64.4%</td>
<td>65.1%</td>
<td>67.4%</td>
<td>68.6%</td>
<td>68.1%</td>
<td>67.5%</td>
</tr>
<tr>
<td>(\beta = 0.8)</td>
<td>50.8%</td>
<td>60.5%</td>
<td>64.8%</td>
<td>64.9%</td>
<td>66.3%</td>
<td>65.8%</td>
<td>65.0%</td>
</tr>
<tr>
<td>(\beta = 1)</td>
<td>50.8%</td>
<td>56.3%</td>
<td>64.0%</td>
<td>63.4%</td>
<td>64.7%</td>
<td>64.2%</td>
<td>63.4%</td>
</tr>
<tr>
<td>(\omega = 0.005)</td>
<td>50.6%</td>
<td>62.8%</td>
<td>65.2%</td>
<td>66.4%</td>
<td>67.7%</td>
<td>67.2%</td>
<td>66.5%</td>
</tr>
<tr>
<td>(\omega = 0.02)</td>
<td>51.0%</td>
<td>62.9%</td>
<td>65.0%</td>
<td>66.4%</td>
<td>67.8%</td>
<td>67.3%</td>
<td>66.5%</td>
</tr>
<tr>
<td>(\omega = 0.08)</td>
<td>51.4%</td>
<td>63.9%</td>
<td>65.2%</td>
<td>67.0%</td>
<td>68.3%</td>
<td>67.8%</td>
<td>67.4%</td>
</tr>
<tr>
<td>(\omega = 0.15)</td>
<td>51.3%</td>
<td>64.2%</td>
<td>65.1%</td>
<td>67.2%</td>
<td>68.6%</td>
<td>68.2%</td>
<td>67.4%</td>
</tr>
</tbody>
</table>

Table 4 Scenario 4. Performance of different algorithms relative to LP (1).

<table>
<thead>
<tr>
<th></th>
<th>ON</th>
<th>Greedy</th>
<th>BPH</th>
<th>(ON_+)</th>
<th>(ON_{+}^+)</th>
<th>(ON_{+}^A)</th>
<th>(ON_{+}^A)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Base</td>
<td>50.4%</td>
<td>64.1%</td>
<td>66.5%</td>
<td>68.6%</td>
<td>69.9%</td>
<td>69.5%</td>
<td>68.8%</td>
</tr>
<tr>
<td>(\alpha = 0)</td>
<td>48.1%</td>
<td>59.5%</td>
<td>66.4%</td>
<td>68.8%</td>
<td>70.0%</td>
<td>69.9%</td>
<td>69.1%</td>
</tr>
<tr>
<td>(\alpha = 0.2)</td>
<td>49.0%</td>
<td>61.2%</td>
<td>67.0%</td>
<td>68.6%</td>
<td>70.1%</td>
<td>69.7%</td>
<td>68.9%</td>
</tr>
<tr>
<td>(\alpha = 0.8)</td>
<td>51.9%</td>
<td>65.9%</td>
<td>65.3%</td>
<td>67.9%</td>
<td>69.1%</td>
<td>68.8%</td>
<td>68.1%</td>
</tr>
<tr>
<td>(\alpha = 1)</td>
<td>53.2%</td>
<td>65.7%</td>
<td>65.3%</td>
<td>67.2%</td>
<td>68.1%</td>
<td>67.8%</td>
<td>67.3%</td>
</tr>
<tr>
<td>(\tau = 2)</td>
<td>50.9%</td>
<td>65.5%</td>
<td>69.2%</td>
<td>70.0%</td>
<td>71.4%</td>
<td>70.8%</td>
<td>70.3%</td>
</tr>
<tr>
<td>(\tau = 5)</td>
<td>50.9%</td>
<td>65.2%</td>
<td>67.5%</td>
<td>69.0%</td>
<td>70.4%</td>
<td>70.0%</td>
<td>69.4%</td>
</tr>
<tr>
<td>(\tau = 20)</td>
<td>49.9%</td>
<td>62.6%</td>
<td>66.0%</td>
<td>68.3%</td>
<td>69.6%</td>
<td>69.2%</td>
<td>68.6%</td>
</tr>
<tr>
<td>(\tau = 30)</td>
<td>49.5%</td>
<td>61.9%</td>
<td>65.9%</td>
<td>68.3%</td>
<td>69.7%</td>
<td>69.4%</td>
<td>68.7%</td>
</tr>
<tr>
<td>(\beta = 0)</td>
<td>50.4%</td>
<td>64.5%</td>
<td>66.4%</td>
<td>68.8%</td>
<td>69.9%</td>
<td>69.6%</td>
<td>68.9%</td>
</tr>
<tr>
<td>(\beta = 0.2)</td>
<td>50.3%</td>
<td>64.4%</td>
<td>66.5%</td>
<td>68.7%</td>
<td>70.0%</td>
<td>69.6%</td>
<td>69.1%</td>
</tr>
<tr>
<td>(\beta = 0.8)</td>
<td>50.2%</td>
<td>62.2%</td>
<td>66.4%</td>
<td>67.7%</td>
<td>68.8%</td>
<td>68.4%</td>
<td>67.8%</td>
</tr>
<tr>
<td>(\beta = 1)</td>
<td>50.3%</td>
<td>59.1%</td>
<td>65.6%</td>
<td>66.4%</td>
<td>67.5%</td>
<td>67.3%</td>
<td>66.5%</td>
</tr>
<tr>
<td>(\omega = 0.005)</td>
<td>50.3%</td>
<td>63.8%</td>
<td>66.4%</td>
<td>68.5%</td>
<td>69.7%</td>
<td>69.2%</td>
<td>68.7%</td>
</tr>
<tr>
<td>(\omega = 0.02)</td>
<td>50.7%</td>
<td>63.9%</td>
<td>66.3%</td>
<td>68.4%</td>
<td>69.7%</td>
<td>69.5%</td>
<td>68.9%</td>
</tr>
<tr>
<td>(\omega = 0.08)</td>
<td>50.3%</td>
<td>64.1%</td>
<td>66.6%</td>
<td>68.6%</td>
<td>69.9%</td>
<td>69.6%</td>
<td>68.9%</td>
</tr>
<tr>
<td>(\omega = 0.15)</td>
<td>50.3%</td>
<td>64.3%</td>
<td>66.6%</td>
<td>68.8%</td>
<td>69.9%</td>
<td>69.6%</td>
<td>69.0%</td>
</tr>
</tbody>
</table>

References


Hu, Ming, Yun Zhou. 2015. Dynamic matching in a two-sided market. *Available at SSRN*.


Pofeldt, Elaine. 2016. Freelancers now make up 35% of u.s. workforce. *Forbes*.


