Dynamic Stochastic Matching Under Limited Time

Ali Aouad
Management Science and Operations, London Business School
Regents Park, London, United Kingdom NW14SA, aaouad@london.edu

Omer Saritac
Management Science and Operations, London Business School
Regents Park, London, United Kingdom NW14SA, osaritac@london.edu

In centralized matching markets such as car-pooling platforms and kidney exchange schemes, new participants constantly enter the market and remain available for potential matches during a limited period of time. To reach an efficient allocation, the "timing" of the matching decisions is a critical aspect of the platform's operations. There is a fundamental trade-off between increasing market thickness and mitigating the risk that participants abandon the market. Nonetheless, the dynamic properties of matching markets have been mostly overlooked in the algorithmic literature.

In this paper, we introduce a general dynamic matching model over edge-weighted graphs, where the agents' arrivals and abandonments are stochastic and heterogeneous. Our main contribution is to design simple matching algorithms that admit strong worst-case performance guarantees for a broad class of graphs. In contrast, we show that the performance of widely used batching algorithms can be arbitrarily bad on certain graph-theoretic structures motivated by car-pooling services. Our approach involves the development of a host of new techniques, including linear programming benchmarks, value function approximations, and proxies for continuous-time Markov chains, which may be of broader interest. In extensive experiments, we simulate the matching operations of a car-pooling platform using real-world taxi demand data. The newly developed algorithms can significantly improve cost efficiency against batching algorithms.

Key words: dynamic matching; approximation algorithms; Markov decision processes; car-pooling.

1. Introduction

The study of centralized matching markets is motivated by a large spectrum of applications such as kidney exchange schemes, ridesharing platforms and public housing programs. In these settings, a central platform controls the matching operations: market participants and resources are matched to maximize social or economic efficiency. To reach an efficient allocation, the "timing" is a critical aspect of the platform's decisions. Indeed, new participants constantly enter the market and remain available for potential matches during a limited period of time. The participants' sojourn time is often uncertain, and they can abandon the market if a satisfying match is not achieved. Making the market participants wait longer often results in more efficient allocations and higher-quality matches, but exposes the system to higher levels of abandonment. Hence, platforms need to adequately time their matching decisions to balance between efficiency and abandonment levels.

One representative example of such market dynamics is given by kidney exchange schemes, which enroll incompatible pairs of donors and recipients seeking mutual exchanges. The arriving market participants are pairs of donors and recipients, which can be classified through various observable attributes, governing their pairwise compatibilities (e.g., blood type, blood markers, time-sensitivity of the transplant). Their lifetime in the system is generally uncertain and can be affected by various exogenous factors, including patient-specific mortality rates and the availability of outside options for transplants. Consequently, the kidney exchange schemes need to strike a good balance between the quality and the latency of their transplant allocation decisions, both of which ultimately influence the survival rates of kidney recipients (Ashlagi et al. 2018). For instance, the living kidney sharing scheme (LKSS) in the United Kingdom conducts matching runs on a quarterly schedule[1]. The rationale is to accumulate, within three months, a sufficiently diverse pool of recipient-donor pairs to perform efficient kidney allocations, while ensuring a relatively high frequency of matches for time-sensitive transplants.

Similar trade-offs are crucial for designing and operating ridesharing platforms, such as Didi, Uber, or Lyft. Their matching mechanism aims to minimize the travel time needed to pick-up riders, to improve riders' satisfaction and increase drivers' hourly earnings. By slightly delaying their matching decisions, these platforms accumulate a *batch* of rider requests before they are assigned to drivers. Due to their substantial benefits, such batching algorithms have been widely implemented in major ridesharing platforms (Lyft 2016). Recent evolutions of car-pooling platforms further illustrate the critical importance of the time dimension. When riders make a request, they might wait for several minutes for the platform to assess potential matches with other riders heading to the same direction, based on their pick-up and drop-off locations (Yan et al. 2020). To inform the design of the matching mechanism, riders' request rate and willingness-to-wait can be estimated from the wealth of historical data.

As illustrated above, matching platforms face a fundamental trade-off between increasing the thickness of the market through longer waiting times and mitigating the risk that participants abandon the system. However, the dynamic properties of matching markets have been mostly overlooked in the classic algorithmic literature, which we review in Section 1.2. The vast majority of previously studied online matching models focus on two extreme types of arrival and abandonment patterns: (i) offline resources stay in the system throughout of the time horizon; (ii) online customers need to be matched immediately upon arrival.

Our work positions in a growing line of research that studies dynamic matching models under nuanced arrival and abandonment patterns (Akbarpour et al. 2020, Hu and Zhou 2021, Ashlagi et al. 2019, Truong and Wang 2019). In view of the current state of affairs, we are the first to study an online stochastic matching problem on a broad class of graph-theoretic structures, where agent's arrivals and abandonments are stochastic and heterogeneous. While the specifics of our modeling approach are presented in Section 2, informally, we formulate the dynamic matching problem as an infinite-horizon continuous-time Markov decision process. Agents of different types arrive according to a Poisson process and abandon the system once their sojourn time elapses. The sojourn times are drawn independently from type-specific exponential distributions, while the compatibility between types and the matching rewards (or costs) are described by an edge-weighted graph.

The main contribution of this paper is to demonstrate that the dynamic properties of matching markets have significant implications on the design of the underlying matching technology. We devise simple matching algorithms with strong performance guarantees in regard to leveraging prior knowledge about the graph structure and market dynamics. We conduct numerical simulations showing that these algorithms can be calibrated to real-world data and implemented at scale.

1.1. Preview of main results

In Section 2, we introduce an online stochastic matching problem on edge-weighted graphs, where the agents' arrivals and abandonments are stochastic and heterogeneous. The problem is formulated as an infinite-horizon continuous-time Markov decision process (MDP) that either minimizes the expected average costs incurred by the matching decisions, or maximizes the expected average rewards. While this problem has no non-trivial competitive ratio (a notion we will formalize shortly that measures the performance of online algorithms against a full-information benchmark), we study the MDP through the lens of approximation algorithms. Our main contributions come in the form of constant-factor approximations in both cost-minimization and reward-maximization settings. It is worth noting that, while there clearly exists an *exact* mapping from cost-minimization instances to reward-maximization instances, the design of *approximation* algorithms is fundamentally different in these settings, owing to the structure of their objective function. Below, we provide a detailed account of our main technical ideas and practical insights.

LP benchmark. We start-off by developing a novel linear programming-based fluid relaxation for our dynamic stochastic matching problem. We show in Section 4.2 that this LP provides a lower bound on the optimal expected average cost in the cost-minimization setting. As explained in Section 5, in the reward-maximization setting, we construct an upper bound on the optimal expected average reward using a quadratically-constrained variant of the LP benchmark.

Constant-factor approximation algorithms. In Section 4, we devise a 3-approximation algorithm for the cost-minimization dynamic stochastic matching problem. This approximation guarantee hinges on the assumptions that (i) the edge-weighted graph satisfies the triangle inequality, (ii) agents have uniform abandonment rates. This setting is motivated by the minimization of time-distance costs of shared rides by car-pooling platforms. Our algorithm builds on a notion of "marginal cost" for each vertex type, which is computed using a simplified variant of our original MDP. By combining the marginal costs, we construct an additive approximation of the cost-to-go at each state of the MDP. Consequently, our algorithm computes a greedy policy with respect to the approximate cost-to-go function. To analyze the resulting matching algorithm, we uncover a relationship between our initial LP benchmark and the notion of marginal costs.

Our main technical contribution is to develop a constant-factor approximation for the reward-maximization MDP in its utmost generality, without any further assumption on the graph structure and the abandonment rates. Specifically, we devise a simple matching algorithm that attains an approximation ratio of $\frac{e-1}{4e}$ on arbitrary edge-weighted graphs; we derive improved performance guarantees in bipartite settings. To keep the paper concise, we provide an outline of our algorithmic approach in Section 5, while the results are formally established in Appendix B. At a high level, our algorithm is based on a variant of the LP benchmark, taking the form of a quadratically-constrained linear program (QCLP). This QCLP relaxation is utilized to specify the distribution of various random edge pruning procedures. Next, our matching policy makes greedy decisions with respect to the pruned subgraphs. The main technical difficulty in analysis is to characterize the stationary distribution of the Markov chain induced by this matching policy. Surprisingly, our approximation guarantees follow by constructing a so-called *virtual Markov chain*, which serves as a simplified proxy for the induced stochastic process.

Simulation case study. In Section 7, we conduct extensive numerical simulations to evaluate our newly developed matching algorithms. We take the perspective of a car-pooling platform that matches pairs of riders to generate cost savings with respect to the utilization of driver time. Using the NY taxi trip data sets, we generate realistic instances that mirror various market conditions faced by car-pooling platforms (commute hours, weekends, etc.). Against batching algorithms, we show that our algorithms achieve significant efficiency gains in certain realistic market conditions.

Negative results. To gain perspective on our algorithmic contributions, we establish two impossibility results, implying a strong separation between our dynamic stochastic matching setting and related online matching problems. First, we argue that there exists no algorithm achieving a positive constant-factor competitive-ratio. Specifically, we compare our LP benchmark to an optimal offline matching algorithm under full knowledge of the arrival and sojourn times on each realization. We show that the optimal offline expected average cost can be arbitrarily smaller than the optimal value of the LP benchmark. Second, we analyze the performance of batching algorithms, which are widely used by ridesharing platforms and kidney exchange schemes. A batching policy periodically computes an optimal matching. Surprisingly, simple counter-examples demonstrate that the performance guarantee of batching policies could be arbitrarily bad in the cost-minimization setting, even if the length of the batching interval is optimally tuned. This family of bad instances is motivated by car-pooling services.

1.2. Related literature

In what follows, we review two long-standing streams of literature on matching optimization, which are both relevant to position our current work. Next, we discuss several recent closely related papers on dynamic matching in greater detail.

Online matching. Our paper is related to the online matching literature. The classic setting, introduced by Karp et al. (1990), describes sequential matching decisions over a bipartite graph. On one side, the vertices are known initially (offline resources), and remain available throughout the horizon. On the other side, the vertices are revealed sequentially (online demand), and should be matched immediately upon their arrival, otherwise they abandon the system. Karp et al. (1990) devise a simple randomized algorithm attaining a $(1-\frac{1}{e})$-*competitive ratio.* Specifically, given $\alpha > 0$, a randomized online algorithm is said to be α-competitive for a reward-maximization matching problem if $\frac{\mathrm{ALG}}{\mathrm{OFF}} \geq \alpha$ for any arbitrary graph instance, where ALG is the expected reward generated by the algorithm, while OFF the *offline* optimal reward, achieved under full knowledge of the underlying graph instance. Consequently, this problem has been studied under various shapes and forms, building on the classic result of Karp et al. (1990). We refer the reader to the survey by Mehta et al. (2013) that covers numerous extensions, including budgeted ad allocation problems (Mehta et al. 2005, Devanur and Hayes 2009). On edge-weighted graphs, Ma and Simchi-Levi (2020) develop nearly-optimal algorithms attaining instance-dependent competitive ratios. In contrast, the stochastic setting, where the arriving online vertices are independently and identically sampled from a known distribution over vertex types, admits constant-factor competitive ratios (Goel and Mehta 2008, Feldman et al. 2009, Manshadi et al. 2012, Jaillet and Lu 2013). Lastly, there is a concurrent stream of literature that studies online metric bipartite matching problems (Kalyanasundaram and Pruhs 1993, Khuller et al. 1994, Bansal et al. 2007, Ashlagi et al. 2017). Even in the presence of i.i.d. stochastic arrivals, the best-known competitive ratio is parametric with respect to the number of requests (Gupta et al. 2019).

Our modeling approach shares similarities with the line of research on stochastic matching problems, which exploit a probabilistic prior on graph instances. Nonetheless, the model studied here differs from the classical setting by capturing a fully-online arrival and abandonment process. Amongst other technical differences, our problem does not admit any positive constant-factor competitive ratio, and our analysis is based on properties of continuous-time stochastic processes. Additionally, we develop specialized results for a cost-minimization formulation, where the underlying graph satisfies a metric-like property. Motivated by car-pooling platforms, our model does not differentiate between "servers" and "customer requests". As such, we are able to devise a constant-factor approximation, in sharp contrast with the best known performance guarantees for related online metric matching problems.

Queuing networks. Matching problems have received a significant amount of attention in the queuing literature that focuses on multi-class multi-server queuing systems. The vast majority of research papers consider a fixed matching policy, described by a service discipline and an exogenous compatibility graph over heterogeneous types of customers and servers. Computing the stationary matching rates over the customer-server types is generally very challenging, and thus, the literature focuses on various approximations. For example, Talreja and Whitt (2008) develop a fluid model to study first-come-first-served (FCFS) discipline in overloaded systems, with further structural restrictions on the underlying compatibility graph. Caldentey et al. (2009) introduced the infinite bipartite matching model. In the basic setting, an infinite stream of arriving customers and servers are matched on a FCFS basis according to a compatibility graph representing the types of customers and servers. The stationary matching rates can be derived on specific compatibility structures, such as 'W'-shaped graphs and almost-complete graphs. Similarly, Adan and Weiss (2012, 2014) study a skill-based parallel queueing system with abandonments under the FCFS-ALIS discipline (first come first served, assign longest idle server). The authors derive the unique stationary distribution of the system, which is in product-form. That said, the state space description is exponentially sized, and the resulting matching rates cannot be easily computed.

Related to our MDP setting, there are a few papers that investigate the *design* of optimal dynamic matching policies under various control objectives. Gurvich and Ward (2014) study a finite-horizon general matching control problem, where the objective is to minimize holding costs. The authors derive a lower bound, emanating from a so-called imbalance process, which is asymptotically matched by a myopic policy. Tsitsiklis and Xu (2017) study a multiserver queueing network under a scaling of the system size. They show that batching policies simultaneously achieve the optimal throughput rate and queuing delays asymptotically, when the underlying network has sufficiently strong connectivity properties. Motivated by ridesharing platforms, Özkan and Ward (2020) study a dynamic matching problem on bipartite graphs where customers need to be matched immediately, while supplies wait in a queue with potential abandonments. The authors show the asymptotic optimality of a myopic randomized policy in a large market regime, based on a continuous linear programming relaxation, which can be simplified into a linear program for time-homogeneous processes. Taking a different perspective, Banerjee et al. (2018) approach the dynamic matching problem in ridesharing through a closed queueing network model, where customers need to be matched immediately upon arrival. They derive state-dependent control policies achieving order optimal decay of the demand loss with respect to the number of supplies. Recently, Cadas et al. (2019) consider the problem of minimizing the long-term average holding cost incurred by waiting times in a bipartite matching system. The authors characterize an optimal threshold-based matching policy under N-shaped graphs with two types of vertices. In a similar vein, Afeche et al. (2021)

study the design of an optimal matching network to balance between waiting time and matching rewards under FCFS-ALIS discipline. They develop a quadratic programming approximation to compute the matching rates in heavy-traffic conditions for a given compatibility graph.

Our paper departs from this line of research on several fronts. First, our model captures heterogeneous patterns of arrivals and abandonments and considers a reward-maximization (or cost-minimization) objective emanating from an edge-weighted compatibility graph. Most importantly, these research papers operate under scaling limits or heavy-traffic conditions. By contrast, we develop matching algorithms achieving *worst-case* performance guarantees with respect to an optimal matching policy. In applications such as ridesharing, the trade-off between market thickness and abandonment risk is due to the scarcity of the demand. Hence, this trade-off is less relevant in scaling limits. This remark will be formalized in Section 3.

Dynamic matching. Closer to our current work, dynamic matching problems have received growing attention in recent literature, motivated by applications to car-pooling platforms (Santi et al. 2014, Alonso-Mora et al. 2017, Bertsimas et al. 2019) and kidney exchanges (Ashlagi et al. 2018). This line of research investigates the design of online matching algorithms on dynamic graphs, where all vertices arrive over time. To our knowledge, the papers by Anderson et al. (2017) and Akbarpour et al. (2020) are the first to study a continuous-time dynamic matching model with stochastic arrivals. Anderson et al. (2017) establish asymptotic optimality results for greedy and batching policies in dynamic barter marketplaces, which further capture cyclic (multi-way) exchanges between market participants. Focusing on bipartite matching settings, Akbarpour et al. (2020) also finds that the greedy algorithm is close to optimal, while leveraging information about when agents' abandon can substantially improve performance. However, the analysis of the preceding papers is based on several stylised assumptions: the underlying network is an Erdös-Rényi random graph, agents are ex-ante identical, and the loss function is the (unweighted) rate of unmatched agents. Subsequent literature has considered several variants of this problem, including unidirectionally differentiated agent types (Hu and Zhou 2021), strategic agents (Baccara et al. 2020), and connections to prophet inequalities (Truong and Wang 2019).

Turning the spotlights on the algorithmic literature, most papers operate under the so-called "criticality" assumption, meaning that the platform knows when a vertex is about to abandon the system; this time period is referred as the vertex's *deadline*. In this setting, Ashlagi et al. (2019) studies an online matching problem on arbitrary edge-weighted graphs, where vertices have a uniform sojourn of d periods. The authors devise a $\frac{1}{4}$-competitive algorithm, where the platform postpones the matching decision for a given vertex until it becomes critical. While this paper comprises a model extension capturing stochastic deadlines, the matching algorithm still utilises the criticality information. Under a similar assumption, Huang et al. (2018, 2019) devise tight competitive ratios for unweighted online matching problems with arbitrary sojourn times. Similarly to the analysis of Huang et al. (2018), we distinguish between so-called *active* and *passive* vertices to construct a novel linear programming relaxation in Section 3. However, our definitions of active and passive vertices slightly differ from those of Huang et al. (2018) since the matching decisions are not "triggered" by the deadlines associated with active vertices. Subsequent to the submission of our paper, Collina et al. (2020) consider a dynamic matching problem with a type-specific Poisson arrival process. Here, the abandonments are both stochastic and unknown to the platform before they occur. In this context, Collina et al. (2020) devise a $\frac{1}{8}$-competitive algorithm for reward maximization, while we obtain an approximation ratio of $\frac{1}{4} \cdot (1 - \frac{1}{e}) \approx 0.158$ in this setting.

2. Modeling Approach

In this section, we introduce the online stochastic matching optimization problem examined in this paper, dubbed *dynamic stochastic matching*. Here, we develop a *cost-minimization* formulation of this problem. An analogous *reward-maximization* formulation is studied in Section 5.

Notation. We begin by defining notation that will be useful throughout the paper. For every integer $\ell \in \mathbb{N}^*$, we use the shorthand $[\ell] = \{1, \dots, \ell\}$. An edge-weighted graph $G = (V, E)$ is endowed with a cost function $c : E \to \mathbb{R}^+ \cup \{+\infty\}$. By extension, for convenience of notation, we may use $c(n, m) = c(\{n, m\})$ for every $\{n, m\} \in E$. Given a matching $M \subseteq E$, we denote by $\phi(M)$ the set of all vertices in V covered by M. Further, let $E(V)$ designate the collection of edges in the complete graph induced by V, and let $S(V)$ be the collection of self-loops. Lastly, given a stochastic process $\{X_t\}_{t \geq 0}$ having right-hand limits everywhere, we denote by $\bar{X}_t = \lim_{u \to t^+} X_u$ the right-hand limit at each $t \geq 0$.

Agents' arrivals and sojourn times. We study a continuous-time model, whereby a stream of random agents arrive and sojourn in the system for a limited amount of time. Each agent, indexed by $n \in \mathbb{N}^* = \{1, 2, \dots\}$, is characterized by an arrival time t_n, a sojourn time δ_n and a type θ_n. Specifically, the agents' arrivals are represented by an independently-marked Poisson point process $\{(t_n, \delta_n, \theta_n)\}_{n \in \mathbb{N}^*}$, where the arrivals t_n are the set of points, while the sojourn times δ_n and the types θ_n are the set of marks.

- The collection of types $\mathcal{T} = \{\theta_n : n \in \mathbb{N}^*\}$ is embedded in a graph $(\mathcal{T}, E)$, where each edge $e \in E$ is assigned with a cost $c(e)$. Without loss of generality, we require that $E = E(\mathcal{T}) \cup S(\mathcal{T})$, provided that the edge costs can be infinite.
- Agents of type $i \in \mathcal{T}$ arrive at a rate $\lambda_i > 0$, and abandon the system at a rate $\mu_i > 0$. Namely, for every agent $n \in \mathbb{N}$, the sojourn time δ_n follows an exponential distribution of rate parameter $\mu_i > 0$ conditional on $\{\theta_n = i\}$.

Consequently, we denote by $N(t)$ the counting process associated with the agents' arrivals, and we let $\mathcal{E}(t)$ be the random subset of times $\zeta \in [0,t]$ at which agent arrive or abandon the system.

Going forward, we define the *sojourn process* as a family of random graphs $\{G_t\}_{t\geq 0}$, whereby the agent's arrivals and departures are represented through graph insertions and deletions. Specifically, upon each agent's arrival, a new vertex is inserted into the current graph, and gets deleted once the sojourn time has elapsed. Formally, $G_t = (V_t, E(V_t))$ is the complete graph over the set of vertices $V_t = \{n \in [N(t)] : t < t_n + \delta_n\}$. For each edge $\{n,m\} \in E(V_t)$, we overload notation by defining the cost $c(n,m) = c(\theta_n, \theta_m)$.

Markov decision process. Let $(\mathcal{F}_t)_{t\geq 0}$ designate the canonical filtration generated by the sojourn process. A matching policy π describes an $\mathcal{F}_t$-adapted stochastic process $\{M_t^\pi\}_{t\geq 0}$, where M_t^π is a matching within the subgraph of G_t induced by the set of remaining vertices $V_t^\pi = V_t \setminus (\bigcup_{e\in[0,t)} \phi(M_e^\pi))$. Hereafter, this subgraph is referred to as the *realization graph* at time t under the matching policy π. The cumulative cost induced by π is represented by a jump process $\{C_t^\pi\}_{t\geq 0}$. Specifically, at each time $t \geq 0$, the cost is accrued by the quantity $c(e)$ for every $e \in M_t^\pi$. In addition, the cost is accrued by a penalty $c_a(i)$ whenever a type-i agent abandons the system. Hence, we denote by $D_t^\pi = \{n \in V_t^\pi : t_n + \delta_n = t\}$ the subset of unmatched agents abandoning the system at time t. Consequently, the cumulative cost C_t^π at time t is given by

$$C_t^\pi = \sum_{\zeta\in\mathcal{E}^\pi(t)} \sum_{e\in M_\zeta^\pi} c(e) + \sum_{\zeta\in\mathcal{E}^\pi(t)} \sum_{n\in D_\zeta^\pi} c_a\left(\theta_n\right) \ , \tag{1}$$

where $\mathcal{E}^\pi(t)$ is the set of jump epochs $\zeta \in [0,t]$, for which $D_\zeta^\pi \neq \emptyset$ or $M_\zeta^\pi \neq \emptyset$.

The objective of the dynamic stochastic matching problem is to devise a matching policy π to minimize the expected average cost $c^\pi = \limsup_{t\to+\infty} \frac{\mathbb{E}[C_t^\pi]}{t}$. Clearly, this problem can be formulated as an average-cost continuous-time Markov decision process (MDP), where, without incurring any loss in optimality, the decision epochs coincide with the times at which agents arrive or abandon the system, i.e., $\mathcal{E}^\pi(t) = \mathcal{E}(t)$. In what follows, we adopt a compact state space representation of this MDP, where each state corresponds to an equivalence class of realization graphs. Specifically, the state space $\mathcal{V}$ is formed by all integral vectors $(\nu_i)_{i\in\mathcal{T}}$, where ν_i is the number of type-i vertices in the current realization graph. For a given matching policy π, we denote by $\vartheta(V_t^\pi) \in \mathcal{V}$ the state of the system at time $t \geq 0$.

It is well-known that there exists an optimal deterministic stationary policy for continuous-time MDPs under mild regularity conditions (e.g., in Appendix A.1, we show that our MDP instances can be reduced to the setting of Guo and Hernández-Lerma (2009, Thm. 5.9)). It is important to note that our dynamic stochastic matching problem cannot be exactly formulated as an average-cost discrete-time MDP using the standard uniformization method (Puterman 2014, Chap. 11.5.3), without slightly altering its probabilistic structure, or imposing further restrictions on the class of admissible policies. Hence, for further generality, we utilize our continuous-time MDP formulation throughout the paper.

Approximation algorithms. While optimal policies are computationally intractable, we will identify simple policies that can be computed in polynomial time, attaining strong performance guarantees with respect to the average-cost criterion. More specifically, we will say that a matching policy π is *α-approximate*, for some constant $\alpha \geq 1$, if $c^{\pi} \leq \alpha \cdot c^{\pi^*}$, where π^* denotes an optimal deterministic stationary policy. Consequently, we will say that an algorithm is an α-approximation if the algorithm in question computes an α-approximate matching policy on every instance of the dynamic stochastic matching problem.

3. Linear Programming Benchmark

In this section, we construct a fluid relaxation of the MDP, which will play a crucial role in our subsequent analysis. To guide our exposition, we introduce the notion of *active* and *passive* vertices. For any given matching policy, a vertex is said to be passive if it is matched with a vertex arriving earlier; otherwise, the vertex is said to be active. The distinction between active and passive vertices is key to capture the probabilistic structure of the agents' abandonments. Intuitively, active vertices have "greater" contributions to the abandonment cost since they wait "longer" in the system.

To formalize this idea, we construct a linear programming formulation, which will be utilized as a benchmark for the cost-minimization dynamic stochastic matching problem:

$$
\begin{array}{llll}
(CB) \quad \min\limits_{x} & \sum\limits_{i \in \mathcal{T}} c_a(i) \cdot x_{i,a} + \sum\limits_{(i,j) \in \mathcal{T}^2} c(i,j) \cdot x_{i,j} & & \\
\text{s.t.} & \sum\limits_{j \in \mathcal{T}} x_{j,i} + \sum\limits_{j \in \mathcal{T}} x_{i,j} + x_{i,a} = \lambda_i \ , & \forall i \in \mathcal{T} & (2) \\
& \dfrac{\mu_i}{\lambda_j} \cdot x_{i,j} \leq x_{i,a} \ , & \forall (i,j) \in \mathcal{T}^2 & (3) \\
& x_{i,j} \geq 0 & \forall (i,j) \in \mathcal{T}^2 &
\end{array}
$$

Let L^* be the minimum cost of (CB). In order to relate the linear program (CB) to a fluid relaxation of our dynamic stochastic matching problem, we view the average match rates induced by a given policy as elementary units of flow. Specifically, for every $(i,j) \in \mathcal{T}^2$, the flow variable $x_{i,j}$ represents the average match rate between active type-i vertices and passive type-j vertices. Similarly, the flow variable $x_{i,a}$ represents the average rate of unmatched type-i vertices abandoning the system. Consequently, constraint (2) can be interpreted as a flow balance equation: every and each arriving vertex is active, passive, or unmatched. Constraint (3) requires that the proportion $x_{i,a}$ of unmatched type-i vertices abandoning the system should be a factor at least $\frac{\mu_i}{\lambda_j}$ relative to the average match rate $x_{i,j}$. Intuitively, this inequality proceeds by observing that an active type-i vertex abandons the system before the next type-j arrival with probability $\frac{\mu_i}{\mu_i+\lambda_j}$; this intuition is formalized in the subsequent analysis.

LEMMA 1. $L^* \le c^{\pi^*}$.

Remark 1: LP benchmark vs. offline benchmark. It is important to highlight that the LP benchmark (CB) does not constitute a lower-bound on the optimal offline matching policy, i.e., the optimal matching under full knowledge of the agents' arrivals and sojourn times on each realization of the stochastic process. Through a simple family of examples constructed in Appendix C, we show that the gap between the optimal LP value and the optimal offline expected average cost can be arbitrarily large. As such, there exists no algorithm attaining a positive constant-factor competitive ratio with respect to the offline benchmark.

Remark 2: Scaling limits. In view of constraint (3), our LP formulation is tighter than the fluid LP relaxation developed by Özkan and Ward (2020) in a related setting. However, in the large market limit studied by the latter paper, where the arrival rates are uniformly scaled and the abandonment rate is held constant, constraint (3) is no longer binding. In fact, Özkan and Ward (2020) show that the myopic LP-based randomized policy is asymptotically optimal. The system controller described by the myopic matching policy does not take advantage of "waiting"; each vertex is either matched immediately upon arrival or irrevocably disposed of, which explains why constraint (3) becomes superfluous in the asymptotic regime.

Proof of Lemma 1. For every $(i,j) \in \mathcal{T}^2$, we define the counting process $\{A_t^*(i,j)\}_{t\ge 0}$ as follows:

$$A_t^*(i,j) = \sum_{\zeta \in \mathcal{E}(t)} \sum_{\substack{\{n,m\} \in M_\zeta^{\pi^*} \\ n<m}} \mathbb{I}\left(\theta_n = i, \theta_m = j\right) \ ,$$

as well as

$$A_t^*(i) = \sum_{\zeta \in \mathcal{E}(t)} \sum_{n \in D_\zeta^{\pi^*}} \mathbb{I}\left(\theta_n = i\right) \ .$$

Namely, $A_t^*(i,j)$ is the number of matches by time t between active vertices of type i and passive vertices of type j, while $A_t^*(i)$ is the number of type-i vertices which have abandoned the system by time t. Consequently, for every $(i,j) \in \mathcal{T}^2$, we define $x_{i,j}^* = \lim_{t\to+\infty} \frac{1}{t} \cdot \mathbb{E}\left[A_t^*(i,j)\right]$; the limit exists in light of the stationarity of π^*. Similarly, we define $x_{i,a}^* = \lim_{t\to+\infty} \frac{1}{t} \cdot \mathbb{E}\left[A_t^*(i)\right]$. We have

$$\begin{aligned} c^{\pi^*} &= \lim_{t\to+\infty} \frac{1}{t} \cdot \mathbb{E}\left[C_t^{\pi^*}\right] \\ &= \sum_{(i,j)\in\mathcal{T}^2} c(i,j) \cdot \left(\lim_{t\to+\infty} \frac{1}{t} \cdot \mathbb{E}\left[A_t^*(i,j)\right] \right) + \sum_{i\in\mathcal{T}} c_a\left(i\right) \cdot \left(\lim_{t\to+\infty} \frac{1}{t} \cdot \mathbb{E}\left[A_t^*(i)\right] \right) \\ &= \sum_{(i,j)\in\mathcal{T}^2} c\left(i,j\right) \cdot x_{i,j}^* + \sum_{i\in\mathcal{T}} c_a(i) \cdot x_{i,a}^* \end{aligned}$$

Hence, in order to establish the inequality $L^* \leq c^{\pi^*}$, it is sufficient to show that the variables $(x^*_{i,j})_{(i,j)\in\mathcal{T}^2}$ and $(x^*_{i,a})_{i\in\mathcal{T}}$ satisfy the constraints of the linear program (CB). Clearly, the vector $(x^*_{i,j})_{i,j}$ satisfies the flow balance equation (2) by noting that each arriving vertex falls into one of the three categories: active, passive, or unmatched.

The remainder of the proof is devoted to establishing constraints (3), by leveraging the associated embedded Markov chain. To this end, let $\{\zeta_q\}_{q\in\mathbb{N}}$ be the sequence of random arrival and abandonment epochs, by increasing order. For every integer $q \geq 0$, we introduce the modified count random variables:

$$\hat{A}_q(i,j) = \sum_{r=0}^{\infty} \sum_{\substack{\{n,m\}\in M^{\pi^*}_{\zeta_r} \\ n<m}} \mathbb{I}\left(\theta_n = i, \theta_m = j, t_m \leq \zeta_q\right) .$$

As such, $\hat{A}_q(i,j)$ is the number of matches executed at *any* time between active type-i vertices and passive type-j vertices, both arriving at or before time ζ_q. Hence, $A^*_{\zeta_q}(i,j)$ counts the matches which occur by time ζ_q, whereas $\hat{A}_q(i,j)$ also counts matches which occur at all subsequent epochs. It is easy to verify that these modified variables asymptotically converge to the same average match rate vector $(x^*_{i,j})_{(i,j)\in\mathcal{T}^2}$, as shown in the next claim.

CLAIM 1. *For every $(i,j)\in\mathcal{T}^2$, $\lim_{q\to+\infty} \frac{1}{\zeta_q}\cdot\mathbb{E}[\hat{A}_q(i,j)] = x^*_{i,j}$ almost surely.*

The proof immediately follows from the stationarity of the matching policy; in order not to deviate from our main argument, the proof is presented in Appendix A.2. Now, fix $(i,j)\in\mathcal{T}^2$, and an integer $q\geq 0$. For every $(i,j)\in\mathcal{T}$, the expected increment of the modified count variable between successive epochs is upper bounded as follows:

$$\begin{aligned}
&\mathbb{E}\left[\hat{A}_{q+1}(i,j) - \hat{A}_q(i,j)\right] \\
&\quad = \sum_{\nu\in\mathcal{V}} \Pr\left[\vartheta\left(V^{\pi^*}_{\zeta_q}\right) = \nu\right] \cdot \mathbb{E}\left[\hat{A}_{q+1}(i,j) - \hat{A}_q(i,j) \middle| \vartheta\left(V^{\pi^*}_{\zeta_q}\right) = \nu\right] \\
&\quad = \sum_{\nu\in\mathcal{V}} \Pr\left[\vartheta\left(V^{\pi^*}_{\zeta_q}\right) = \nu\right] \cdot \mathbb{E}\left[\sum_{r=0}^{\infty} \sum_{\substack{\{n,m\}\in M^{\pi^*}_{\zeta_r} \\ n<m}} \mathbb{I}\left(\theta_n = i, \theta_m = j, t_m = \zeta_{q+1}\right) \middle| \vartheta\left(V^{\pi^*}_{\zeta_q}\right) = \nu\right] \cdot \\
&\quad \leq \sum_{\nu\in\mathcal{V}} \Pr\left[\vartheta\left(V^{\pi^*}_{\zeta_q}\right) = \nu\right] \cdot \mathbb{E}\left[\mathbb{I}\left(\nu_i > 0, \vartheta_j\left(V^{\pi^*}_{\zeta_{q+1}}\right) = \nu_j + 1\right) \middle| \vartheta\left(V^{\pi^*}_{\zeta_q}\right) = \nu\right] \\
&\quad = \sum_{\nu\in\mathcal{V}} \Pr\left[\vartheta\left(V^{\pi^*}_{\zeta_q}\right) = \nu\right] \cdot \mathbb{I}\left(\nu_i > 0\right) \cdot \frac{\lambda_j}{\sum_{k\in\mathcal{T}} \lambda_k + \sum_{k\in\mathcal{T}} \nu_k \cdot \mu_k} \qquad (4)
\end{aligned}$$

On the other hand, for every $i\in\mathcal{T}$, we have:

$$\mathbb{E}\left[A^*_{\zeta_{q+1}}(i) - A^*_{\zeta_q}(i)\right] = \sum_{\nu\in\mathcal{V}} \Pr\left[\vartheta\left(V^{\pi^*}_{\zeta_q}\right) = \nu\right] \cdot \mathbb{E}\left[A^*_{\zeta_{q+1}}(i) - A^*_{\zeta_q}(i) \middle| \vartheta\left(V^{\pi^*}_{\zeta_q}\right) = \nu\right]$$

$$= \sum_{\nu \in \mathcal{V}} \Pr\left[\vartheta\left(V^{\pi^*}_{\zeta_q}\right) = \nu\right] \cdot \Pr\left[\vartheta_i\left(V^{\pi^*}_{\zeta_{q+1}}\right) = \nu_i - 1 \middle| \vartheta\left(V^{\pi^*}_{\zeta_q}\right) = \nu\right]$$
$$= \sum_{\nu \in \mathcal{V}} \Pr\left[\vartheta\left(V^{\pi^*}_{\zeta_q}\right) = \nu\right] \cdot \nu_i \cdot \frac{\mu_i}{\sum_{k \in \mathcal{T}} \lambda_k + \sum_{k \in \mathcal{T}} \nu_k \cdot \mu_k} \ .$$

By combining the latter equality with (4), for every $(i,j) \in \mathcal{T}^2$, we obtain

$$\mathbb{E}\left[\hat{A}_{q+1}(i,j) - \hat{A}_q(i,j)\right] \leq \frac{\lambda_j}{\mu_i} \cdot \mathbb{E}\left[A^*_{\zeta_{q+1}}(i) - A^*_{\zeta_q}(i)\right] \ .$$

By summing over all integers $q \in \mathbb{N}$, and by Claims 1, we conclude that the flow vector formed by $(x^*_{i,j})_{(i,j)\in\mathcal{T}^2}$ and $(x^*_{i,a})_{i\in\mathcal{T}}$ satisfies the ensemble of constraints (3). $\square$

4. Approximation Algorithm for Cost Minimization

In this section, we devise a constant-factor approximation for the cost-minimization dynamic stochastic matching problem, introduced in Section 2. Our main result in the cost-minimization setting relies on two structural assumptions, described in the sequel.

Assumption 1. *The cost function satisfies the triangle inequality. Namely, for every types $i,j,k \in \mathcal{T}$, we have $c(i,j)+c(j,k) \geq c(i,k)$ and $c_a(i)+c(i,j) \geq c_a(j)$.*

In particular, Assumption 1 implies that $c(i,i) \leq 2 \cdot c(i,j)$ for every $i,j \in \mathcal{T}$, and thus, vertices of identical types can be matched. This assumption is motivated by car-pooling platforms, where the goal is to minimize the time-distance cost generated by the shared rides (Santi et al. 2014). In this context, what we consider an "ideal" match is one that pairs riders having identical pick-up and drop-off locations. Moreover, the cost function naturally satisfies the triangle inequality, as further explained in Section 7.

Assumption 2. *The abandonment rates are uniform, namely, $\mu_i = \mu$ for every $i \in \mathcal{T}$.*

Having introduced the structural assumptions utilized in this section, we are ready to state our main theorem in the cost-minimization setting.

Theorem 1. *Under Assumptions 1 and 2, there exists a polynomial-time 3-approximation algorithm for the cost-minimization dynamic stochastic matching problem.*

It is worth highlighting that the matching policy attaining the approximation ratio stated by Theorem 1 has a simple and interpretable structure. At a high-level, each vertex type is associated with a notion of "marginal cost". The matching decisions are guided by an additive approximation of the cost-to-go, constructed by combining these marginal costs. Specifically, our algorithmic approach and analysis proceed in three steps.

Step 1: Active-vertex problem. We begin by considering a simplified problem, dubbed the *type-i active-vertex problem* for every given type $i \in \mathcal{T}$, for which an optimal matching policy can be easily computed. As made formal in Section 4.1, the problem consists in minimizing the expected cost generated by a single active vertex n_i of type i, available at time $t = 0$. Namely, either n_i is matched with some vertex n arriving during her sojourn and incurs the corresponding cost $c(n_i, n)$, or n_i abandons the system unmatched and incurs the penalty $c_a(i)$. By focusing on a single vertex n_i, the matching policy reduces to a stopping rule: clearly, upon matching vertex n_i, it is optimal to pick the vertex of minimal cost within the realization graph. Interestingly, we show in Section 4.1 that there exists a fixed-threshold policy which is optimal. Namely, there exists a threshold $\bar{c}_i \geq 0$ such that it is optimal to match n_i with the first arriving vertex n for which $c(i, \theta_n) \leq \bar{c}_i$, if any.

Step 2: Lower bound. In Section 4.2, we relate the thresholds arising from the active-vertex problems to a lower bound on our original dynamic stochastic matching problem. More concretely, we show that $\frac{1}{3} \cdot \sum_{i \in \mathcal{T}} \lambda_i \bar{c}_i$ is a lower bound on the optimal expected average cost under Assumptions 1 and 2. Intuitively, the active-vertex problem is “optimistic”, in the sense that the “competition” between active vertices is overlooked. That said, a lower bound on our original problem does immediately follow from this observation. The main challenge is that the matching policy ultimately controls the relative rates of active vs. passive vertices. Hence, our lower bound proceeds from jointly bounding the cost contributions of active and passive vertices, using the LP benchmark of Lemma 1 and the triangle inequality property (Assumption 1).

Step 3: Vertex-additive matching policy. Lastly, in Section 4.3, we devise a simple matching policy, referred to as *vertex-additive*, that matches the lower bound established in Step 2, up to a constant multiplicative factor. At a high-level, our policy constructs an additive approximation of the cost-to-go by interpreting $\bar{c}_i$ as the “marginal cost” associated each type-i vertex. Consequently, our matching policy is greedy with respect to this linear approximation of the cost-to-go function.

4.1. Step 1: Active-vertex problem

Given a type $i \in \mathcal{T}$, we proceed with a formal definition of the *type-i active-vertex* problem. In what follows, we assume that the sojourn process is initialized by $V_0 = \{n_i\}$, where $\theta_{n_i} = i$ and $t_{n_i} = 0$. Conditional on the initial state $V_0 = \{n_i\}$, the sojourn process $\{V_t\}_{t \geq 0}$ is identical to our original process, described in Section 2. For ease of notation, we use the shorthand $V_t^- = V_t \setminus \{n_i\}$ to designate the set of vertices at time t at the exclusion of n_i. The objective is to minimize the expected cost generated by n_i, either by matching with a subsequently arriving vertex during its sojourn, or by incurring the abandonment penalty. Specifically, an admissible policy corresponds to a stopping rule τ, generating a cost $\hat{c}_\tau$ defined as follows:

$$\hat{c}_\tau = \begin{cases} c\left(i, \vartheta\left(V_\tau^-\right)\right) \ , & \text{if } \tau < \delta_{n_i} \text{ and } |V_\tau^-| \geq 1 \ , \\ c_a(i) \ , & \text{otherwise} \ , \end{cases} \tag{5}$$

where we use the shorthand $c(i,\nu) = \min\{c(i,j) : j \in \mathcal{T}, \nu_j > 0\}$. The objective is to devise a stopping rule τ that minimizes $\mathbb{E}[\hat{c}_\tau]$. Letting τ^* be an optimal stopping rule, we use $\bar{c}_i = \mathbb{E}[\hat{c}_{\tau^*}]$ to denote the optimal cost of the type-i active-vertex problem.

Threshold policies. Given a constant $\alpha \geq 0$, the α-threshold policy matches n_i with the first arriving vertex n before vertex n_i abandons the system such that $c(n_i, n) \leq \alpha$, if any, i.e., the α-threshold policy corresponds to the stopping rule

$$\tau_\alpha = \min\left\{\delta_{n_i}, \min\left\{t \geq 0 : c\left(i, \vartheta\left(V_t^-\right)\right) \leq \alpha\right\}\right\} \ .$$

Clearly, we have $\tau_\alpha < \infty$ almost surely. In the next lemma, we show that the $\bar{c}_i$-threshold policy is optimal for the type-i active-vertex problem. This result implies that there is no benefit in "accumulating" passive vertices; every arriving vertex is either immediately matched to n_i, or irrevocably disposed of. At the end of this section, we provide algorithmic means to compute $\bar{c}_i$ by characterizing the expected cost generated by threshold policies in closed-form expression.

Lemma 2. *The $\bar{c}_i$-threshold policy is optimal for the type-i active-vertex problem.*

Proof. We formulate our optimal stopping problem through the associated embedded Markov chain. To this end, let $\{\zeta_q\}_{q\in\mathbb{N}^*}$ be the sequence of random arrival and abandonment epochs, by increasing order. It is easy to verify that the stopping times can be restricted to the set $\{\zeta_q : q \in \mathbb{N}^*\}$, without incurring any loss in optimality. Consequently, the optimal stopping problem is expressed through an ensemble of recursive equations. Since the sojourn process satisfies the strong Markov property, the expected cost-to-go $\mathbb{E}[\hat{c}_{\tau^*}|\tau^* \geq \zeta_q, \vartheta(V_{\zeta_q}^-) = \nu, \delta_{n_i} > \zeta_q]$ when reaching state ν at the q-th epoch does not depend on $q \in \mathbb{N}^*$. Hence, we define $F(\nu) = \mathbb{E}[\hat{c}_{\tau^*}|\vartheta(V_0^-) = \nu]$, which is equal to the expected cost-to-go $\mathbb{E}[\hat{c}_{\tau^*}|\tau^* \geq \zeta_q, \vartheta(V_{\zeta_q}^-) = \nu, \delta_{n_i} > \zeta_q]$ for every $q \in \mathbb{N}^*$. The cost-to-go function $F(\cdot)$ satisfies the following Bellman equation:

$$F(\nu) = \min\left\{c(i,\nu), \frac{1}{\mu\cdot|\nu| + \mu + \sum_{k\in\mathcal{T}}\lambda_k}\cdot\left(\mu\cdot c_a(i) + \sum_{j\in\mathcal{T}}\lambda_j\cdot F(\nu + e_j) + \sum_{j\in\mathcal{T}}\nu_j\mu\cdot F(\nu - e_j)\right)\right\}, \tag{6}$$

where we use the shorthand $|\nu| = \sum_{i\in\mathcal{T}}\nu_i$. To instill intuition about this recursive equation, we remark that all possible outcomes are captured by the cost-to-go expression: (1) vertex n_i abandons the system at the next epoch, (2) a type-j vertex arrives at the next epoch, for some $j \in \mathcal{T}$, or (3) a type-j vertex abandons the system at the next epoch, for some $j \in \mathcal{T}$.

Equation (6) describes an undiscounted infinite-horizon MDP, where an absorbing state is reached whenever n_i is matched or abandons the system. Since the sojourn time of n_i is almost surely bounded, any admissible policy of this MDP is proper, and the optimal value function $\{F(\nu)\}_{\nu\in\mathcal{V}}$ corresponds to the unique fixed point of the Bellman operator; see Bertsekas and Tsitsiklis (1996). Now, let $\mathcal{A}_i^*$ be the subset of types that minimizes the function $f_i(\cdot)$, where

$$f_i\left(\mathcal{A}\right) = \frac{\mu}{\mu+\sum_{k\in\mathcal{A}}\lambda_k}\cdot c_a(i) + \sum_{j\in\mathcal{A}}\frac{\lambda_j}{\mu+\sum_{k\in\mathcal{A}}\lambda_k}\cdot c\left(i,j\right)\ ,$$

for every $\mathcal{A}\subseteq\mathcal{T}$. In the next claim, we express $F(\nu)$ as a function of $f_i(\mathcal{A}_i^*)$ for every state ν. In particular, the lemma implies that $\bar{c}_i = f_i(\mathcal{A}_i^*)$ and the $\bar{c}_i$-threshold policy is optimal.

Lemma 3. *$F(\nu) = \min\{f_i(\mathcal{A}_i^*), c(i,\nu)\}$ for every $\nu\in\mathcal{V}$.*

Proof. We begin by highlighting a basic property of the optimal subset of types $\mathcal{A}_i^*$ with respect to the minimization of the function $f_i(\cdot)$.

Claim 2. $\mathcal{A}_i^* = \{j\in\mathcal{T} : c(i,j)\leq f_i(\mathcal{A}_i^*)\}$.

The proof of Claim 2 is presented in Appendix A.3. Now, define $\hat{F}(\nu) = \min\{f_i(\mathcal{A}_i^*), c(i,\nu)\}$ for every $\nu\in\mathcal{V}$. It is sufficient to show that $\hat{F}(\cdot)$ satisfies the Bellman equations (6) to establish the desired claim. To this end, we fix a state $\nu\in\mathcal{V}$. In what follows, we examine the case where $f_i(\mathcal{A}_i^*)\geq c(i,\nu)$. The opposite case $f_i(\mathcal{A}_i^*) < c(i,\nu)$ proceeds from a nearly-identical reasoning (see Appendix A.4).

Case 1: $f_i(\mathcal{A}_i^*)\geq c(i,\nu)$. We begin by bounding the terms $\hat{F}(\nu+e_j)$ and $\hat{F}(\nu-e_j)$ for every $j\in\mathcal{T}$. To this end, we define $\mathcal{A}^+ = \{j\in\mathcal{T} : c(i,j)\leq c(i,\nu)\}$. For every $j\in\mathcal{A}^+$,

$$\hat{F}\left(\nu+e_j\right) = \min\left\{f_i(\mathcal{A}_i^*), c\left(i,\nu\right), c\left(i,j\right)\right\} = c\left(i,j\right)\ . \tag{7}$$

Now, for every $j\in\mathcal{T}\setminus\mathcal{A}^+$, we have

$$\hat{F}\left(\nu+e_j\right) = \min\left\{f_i(\mathcal{A}_i^*), c\left(i,\nu\right), c\left(i,j\right)\right\} = c\left(i,\nu\right)\ , \tag{8}$$

where the last equality proceeds from our case hypothesis. Lastly, for every $j\in\mathcal{T}$, we have

$$\hat{F}\left(\nu-e_j\right) = \min\left\{f_i(\mathcal{A}_i^*), c\left(i,\nu-e_j\right)\right\} \geq \min\left\{f_i(\mathcal{A}^*), c\left(i,\nu\right)\right\} = c\left(i,\nu\right)\ , \tag{9}$$

where the last inequality proceeds from our case hypothesis. Thus, we have

$$\begin{aligned}
\hat{F}\left(\nu\right) &= \min\left\{f_i(\mathcal{A}_i^*), c\left(i,\nu\right)\right\} \\
&= c\left(i,\nu\right) \\
&= \frac{\mu+\sum_{k\in\mathcal{A}^+}\lambda_k}{\mu\cdot|\nu|+\mu+\sum_{k\in\mathcal{T}}\lambda_k}\cdot\ c\left(i,\nu\right) + \frac{\mu\cdot|\nu|+\sum_{k\in\mathcal{T}\setminus\mathcal{A}^+}\lambda_k}{\mu\cdot|\nu|+\mu+\sum_{k\in\mathcal{T}}\lambda_k}\cdot c\left(i,\nu\right) \\
&\leq \frac{\mu+\sum_{k\in\mathcal{A}^+}\lambda_k}{\mu\cdot|\nu|+\mu+\sum_{k\in\mathcal{T}}\lambda_k}\cdot\ f_i\left(\mathcal{A}_i^*\right) + \frac{\mu\cdot|\nu|+\sum_{k\in\mathcal{T}\setminus\mathcal{A}^+}\lambda_k}{\mu\cdot|\nu|+\mu+\sum_{k\in\mathcal{T}}\lambda_k}\cdot c\left(i,\nu\right)
\end{aligned}$$

$$
\begin{aligned}
&\leq \frac{\mu + \sum_{k\in\mathcal{A}^+}\lambda_k}{\mu\cdot|\nu| + \mu + \sum_{k\in\mathcal{T}}\lambda_k}\cdot\ f_i\left(\mathcal{A}^+\right) + \frac{\mu\cdot|\nu| + \sum_{k\in\mathcal{T}\setminus\mathcal{A}^+}\lambda_k}{\mu\cdot|\nu| + \mu + \sum_{k\in\mathcal{T}}\lambda_k}\cdot c\left(i,\nu\right) \\
&= \frac{\mu + \sum_{k\in\mathcal{A}^+}\lambda_k}{\mu\cdot|\nu| + \mu + \sum_{k\in\mathcal{T}}\lambda_k}\left(\frac{\mu}{\mu + \sum_{k\in\mathcal{A}^+}\lambda_k}\cdot c_a(i) + \sum_{j\in\mathcal{A}^+}\frac{\lambda_j}{\mu + \sum_{k\in\mathcal{A}^+}\lambda_k}\cdot c\left(i,j\right)\right) \\
&\quad + \frac{\mu\cdot|\nu| + \sum_{k\in\mathcal{T}\setminus\mathcal{A}^+}\lambda_k}{\mu\cdot|\nu| + \mu + \sum_{k\in\mathcal{T}}\lambda_k}\cdot c\left(i,\nu\right) \\
&\leq \frac{1}{\mu\cdot|\nu| + \mu + \sum_{k\in\mathcal{T}}\lambda_k}\cdot\left(\mu\cdot c_a(i) + \sum_{j\in\mathcal{T}}\lambda_j\cdot\hat{F}\left(\nu+e_j\right) + \sum_{j\in\mathcal{T}}\mu\cdot\nu_j\cdot\hat{F}\left(\nu-e_j\right)\right)\ ,
\end{aligned}
$$

where the first inequality follows from the case hypothesis, and the second inequality proceeds from the optimality of $\mathcal{A}_i^*$. The last inequality holds by combining (7), (8) and (9). Hence, the Bellman equation (6) is satisfied with respect to state ν. $\square$

$\square$

Basic properties of $\bar{c}_i$. In the proof of Lemma 2, we have shown that $\bar{c}_i = f_i(\mathcal{A}_i^*)$, where

$$
f_i\left(\mathcal{A}_i^*\right) = \min_{\mathcal{A}\subseteq\mathcal{T}}\ \frac{\mu}{\mu + \sum_{k\in\mathcal{A}}\lambda_k}\cdot c_a(i) + \sum_{j\in\mathcal{A}}\frac{\lambda_j}{\mu + \sum_{k\in\mathcal{A}}\lambda_k}\cdot c\left(i,j\right)\ , \tag{10}
$$

A close examination of the optimization problem (10) reveals a connection to the unconstrained MNL-assortment optimization problem. In this context, it is well-known that the optimal solution takes the form of a revenue-ordered assortment (Talluri and van Ryzin 2004). The revenue-ordered structure is analogous to the property established in Claim 2, whereby $\mathcal{A}_i^*$ is comprised of all types incurring a cost $c(i,j)$ below a given threshold. By leveraging the connection to the MNL-assortment optimization problem, we can formulate problem (10) as the following linear program:

$$
\begin{array}{llll}
(AN_i) & \min\limits_{y} & c_a(i)\cdot y_a + \sum\limits_{j\in\mathcal{T}} c(i,j)\cdot y_j & \\
& \text{s.t.} & y_a + \sum\limits_{j\in\mathcal{T}} y_j = 1\ , & \\
& & \dfrac{\mu}{\lambda_j}\cdot y_j \leq y_a\ , & \forall j\in\mathcal{T} \\
& & y_j, y_a \geq 0 & \forall j\in\mathcal{T}
\end{array}
$$

In view of the exact analogy between the active-vertex problem and the MNL-assortment optimization problem, the proof of the next lemma is omitted (Gallego et al. 2014).

LEMMA 4. *The minimum cost of the linear program (AN_i) is $\bar{c}_i$. Furthermore, $\mathcal{A}_i^* = \{j\in\mathcal{T} : y_j^* > 0\}$, where $(y_j^*)_{j\in\mathcal{T}}$ describes the optimal solution vector of (AN_i).*

In the next claim, we derive an inequality that will be useful to bound the variations of the optimal expected costs $\bar{c}_i$ across different types $i\in\mathcal{T}$. The proof immediately proceeds from the triangle inequality (Assumption 1), and thus, it is deferred to Appendix A.5.

LEMMA 5. *For every $i,j\in\mathcal{T}$, $\bar{c}_i \leq \bar{c}_j + c(i,j)$.*

4.2. Step 2: Lower bound

In the next lemma, we develop a lower bound on the optimal expected average cost, as a function of the active-vertex expected costs $\{\bar{c}_i\}_{i\in\mathcal{T}}$. The proof is based on on the LP benchmark (CB) developed in Section 4.2 as well as the triangle inequality property (Assumption 1).

LEMMA 6. $\frac{1}{3}\cdot\sum_{i\in\mathcal{T}}\bar{c}_i\cdot\lambda_i\le c^{\pi^*}$

Proof. Let $(x^*_{i,j})_{(i,j)\in\mathcal{T}^2}$ and $(x^*_{i,a})_{i\in\mathcal{T}}$ form an optimal solution vector of problem (CB). For every $i\in\mathcal{T}$, we define $s_i=x^*_{i,a}+\sum_{j\in\mathcal{T}}x^*_{i,j}$. By noting that the vector formed by $(x^*_{i,j}/s_i)_{j\in\mathcal{T}}$ and $x^*_{i,a}/s_i$ satisfies the constraints of the linear program (AN_i), and by Lemma 4, we infer that

$$c_a(i)\cdot x^*_{i,a}+\sum_{j\in\mathcal{T}}c(i,j)\cdot x^*_{i,j}\ge\bar{c}_i\cdot s_i \tag{11}$$

Hence, we obtain

$$\begin{aligned}\sum_{i\in\mathcal{T}}\bar{c}_i\lambda_i&=\sum_{i\in\mathcal{T}}\bar{c}_i\cdot s_i+\sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{T}}\bar{c}_i\cdot x^*_{j,i}\\&\le\sum_{i\in\mathcal{T}}\bar{c}_i\cdot s_i+\sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{T}}c(j,i)\cdot x^*_{j,i}+\sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{T}}\bar{c}_j\cdot x^*_{j,i}\\&=\sum_{i\in\mathcal{T}}\left(1+\frac{s_i-x^*_{i,a}}{s_i}\right)\cdot\bar{c}_i\cdot s_i+\sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{T}}c(j,i)\cdot x^*_{j,i}\\&\le\sum_{i\in\mathcal{T}}\left(\left(1+\frac{s_i-x^*_{i,a}}{s_i}\right)\cdot c_a(i)\cdot x^*_{i,a}+\left(2+\frac{s_i-x^*_{i,a}}{s_i}\right)\cdot\sum_{j\in\mathcal{T}}c(i,j)\cdot x^*_{i,j}\right)\\&\le 3\cdot\left(\sum_{i\in\mathcal{T}}c_a\cdot x^*_{i,a}+\sum_{(i,j)\in\mathcal{T}^2}c(i,j)\cdot x^*_{i,j}\right)\\&=3\cdot L^*\ ,\end{aligned}$$

where the first inequality holds due to Lemma 5 and the second inequality proceeds from (11). □

4.3. Step 3: Vertex-additive matching policy

Finally, in this section, we devise a simple greedy-like matching policy that competes against the lower bound derived in Lemma 6, up to a constant multiplicative factor.

Matching policy. The *vertex-additive* matching policy $\bar{\pi}$ is defined such that $M^{\bar{\pi}}_t$ is a minimum-cost matching within the realization graph at time t with respect to the cost function $M\mapsto\sum_{e\in M}c(e)-\sum_{n\in\phi(M)}\bar{c}_{\theta_n}$; we break ties among matchings of minimum cost by picking $M^{\bar{\pi}}_t$ as an arbitrary matching of maximum cardinality. Policy $\bar{\pi}$ can be interpreted as the greedy policy with respect to the linear approximation of the cost-to-go $\sum_{i\in\mathcal{T}}\nu_i\cdot\bar{c}_i$ at each state $\nu\in\mathcal{V}$; e.g., see Bertsekas and Tsitsiklis (1996) for further background on greedy policies in the context of MDPs.

Algorithm 1 Computing the vertex-additive policy

```
Instance parameters: Collection of types $\mathcal{T}$, arrival rates $(\lambda_i)_{i\in\mathcal{T}}$, abandonment rates $(\mu_i)_{i\in\mathcal{T}}$, matching costs $c(\cdot)$, abandonment costs $\bar{c}_a(\cdot)$.

Offline computation: Solving the active-vertex problems
For every $i \in \mathcal{T}$:
    Compute $\bar{c}_i$ by solving the linear program $(AN_i)$

Online computation: Min-cost matching
Input: Realization graph $G_t = (V_t^{\bar{\pi}}, E(V_t^{\bar{\pi}}))$ at time $t \geq 0$
Compute a min-cost matching $M_t^{\bar{\pi}} \subseteq E(V_t^{\bar{\pi}})$ that minimizes $M \mapsto \sum_{e\in M} c(e) - \sum_{n\in\phi(M)} \bar{c}_{\theta_n}$
Output: Return $M_t^{\bar{\pi}}$
```

Intuitively, this matching policy uses $\bar{c}_i$ as an estimate for the "marginal cost" of each type-i vertex. Our algorithmic approach is summarized by the pseudo-code of Algorithm 1.

Since vertices arrive one at a time, the matching $M_t^{\bar{\pi}}$ is either empty or a singleton, for every $t \geq 0$. Moreover, every match $\{n,m\} \in M_t^{\bar{\pi}}$ picked by the vertex-additive policy necessarily satisfies $c(n,m) \leq \bar{c}_{\theta_n} + \bar{c}_{\theta_m}$, otherwise eliminating the match $\{n,m\}$ is cost-wise beneficial with respect to our approximation of the cost-to-go. Building on these properties, we show that the vertex-additive matching policy is 3-approximate, thereby completing the proof of Theorem 1.

LEMMA 7. $c^{\bar{\pi}} \leq 3 \cdot c^{\pi^*}$.

Proof. The remainder of this section is devoted to proving Lemma 7. To this end, we begin by introducing a notion of allocation cost, which is key to our analysis.

Allocated cost. For every vertex $n \in \mathbb{N}^*$, we define the *allocated cost* $\alpha^{\bar{\pi}}(n)$ as the following random variable:

$$\alpha^{\bar{\pi}}(n) = \begin{cases} \bar{c}_{\theta_n} & \text{if } n \text{ is passive,} \\ c_a(\theta_n) & \text{if } n \text{ is active and abandons the system ,} \\ c(n,m) - \bar{c}_{\theta_m} & \text{if } n \text{ is active and matched with vertex } m \text{ .} \end{cases}$$

Observe that, on every realization, we have $c(n,m) = \alpha^{\bar{\pi}}(n) + \alpha^{\bar{\pi}}(m)$ for every match $\{n,m\} \in \bigcup_{t\geq 0} M_t^{\bar{\pi}}$. Thus, it is not difficult to verify that $c^{\bar{\pi}} = \lim_{t\to+\infty} \frac{1}{t} \cdot \mathbb{E}[\sum_{n\in[N(t)]} \alpha^{\bar{\pi}}(n)]$. Furthermore, in the next claim, we develop an upper-bound on the expected allocated cost conditional on the vertex type.

LEMMA 8. *For every $n \in \mathbb{N}^*$ and $i \in \mathcal{T}$, we have* $\mathbb{E}\left[\alpha^{\bar{\pi}}(n) | \theta_n = i\right] \leq \bar{c}_i$.

The proof of Lemma 8 is deferred to Appendix A.6. Consequently, we obtain:

$$\begin{aligned}
c^{\bar{\pi}} &= \lim_{t\to+\infty} \frac{1}{t} \cdot \mathbb{E}\left[\sum_{n\in[N(t)]} \sum_{i\in\mathcal{T}} \mathbb{E}\left[\alpha^{\bar{\pi}}(n)\cdot \mathbb{I}\left(\theta_n = i\right)\right]\right] \\
&\le \lim_{t\to+\infty} \frac{1}{t} \cdot \mathbb{E}\left[\sum_{n\in[N(t)]} \sum_{i\in\mathcal{T}} \bar{c}_i \cdot \mathbb{E}\left[\mathbb{I}\left(\theta_n = i\right)\right]\right] \\
&= \left(\sum_{i\in\mathcal{T}} \frac{\lambda_i}{\sum_{j\in\mathcal{T}}\lambda_j}\cdot \bar{c}_i\right)\cdot\left(\lim_{t\to+\infty} \mathbb{E}\left[\frac{N(t)}{t}\right]\right) \\
&= \sum_{i\in\mathcal{T}} \lambda_i \cdot \bar{c}_i \\
&\le 3\cdot c^{\pi^*} ,
\end{aligned}$$

where the first inequality follows from Lemma 8, the third equality holds by Wald's equation, and the last inequality proceeds from Lemma 6. □

5. Approximation Algorithm for Reward Maximization

In this section, we study the reward-maximization formulation of the dynamic stochastic matching problem. Here, the collection of types $\mathcal{T} = \{\theta_n : n \in \mathbb{N}^*\}$ is embedded in a graph $(\mathcal{T}, E)$, where each edge $e \in E$ is assigned with an arbitrary reward $r(e)$. Without loss of generality, we require that $E = E(\mathcal{T}) \cup S(\mathcal{T})$, provided that the rewards can be 0. The sojourn process $\{G_t\}_{t\ge 0}$ is identical to the cost-minimization setting, described in Section 2. Namely, at every time $t \ge 0$, $G_t = (V_t, E(V_t))$ is the complete graph over the set of vertices $V_t = \{n \in [N(t)] : t < t_n + \delta_n\}$. Each edge $\{n, m\} \in E(V_t)$ is associated with a reward $r(n,m) = r(\theta_n, \theta_m)$. Consequently, a matching policy π induces a cumulative reward process $\{R^\pi_t\}_{t\ge 0}$. At each decision epoch ζ, the cumulative reward is accrued by the quantity $\sum_{e\in M^\pi_\zeta} r(e)$. Namely, for every $t \ge 0$,

$$R^\pi_t = \sum_{\zeta\in\mathcal{E}(t)} \sum_{e\in M^\pi_\zeta} r(e) \ .$$

The objective is to devise a matching policy π that maximizes the expected average reward $r^\pi = \liminf_{t\to+\infty} \frac{\mathbb{E}[R^\pi_t]}{t}$. We say that π is an α-approximate policy, for some constant $\alpha \in (0,1]$, if $r^\pi \ge \alpha \cdot r^{\pi^*}$, where π^* designates an optimal deterministic stationary policy. In what follows, we also consider two special cases of interest: (i) We say that the underlying graph is *bipartite* when the subgraph formed by the edges $\bar{E} = \{e \in E : r(e) \ne 0\}$ with non-zero rewards is bipartite, i.e., there is a partition $\mathcal{T}_1, \mathcal{T}_2$ of the collection of types $\mathcal{T}$ such that $(\mathcal{T}_1, \mathcal{T}_2, \bar{E})$ is a bipartite graph; (ii) The *bipartite-asymmetric* setting refers to such bipartite graphs $(\mathcal{T}_1, \mathcal{T}_2, \bar{E})$ where vertices on one side of the graph leave immediately after their arrival, i.e., with a slight abuse of notation, we require that $\mu_i = +\infty$ for all $i \in \mathcal{T}_1$ or $\mu_i = +\infty$ for all $i \in \mathcal{T}_2$. This setting can be viewed as a continuous-time formulation of the classical online stochastic matching problem, where "online" demand vertices should be immediately matched upon their arrival lest they abandon, but "offline" resources are replenished according to a stationary stochastic process.

Our main result comes in the form of a constant-factor approximation for the reward-maximization setting for arbitrary graph structures and abandonment rates. As detailed by the next theorem, improved approximation ratios are established under additional structural assumptions on the underlying graph.

Theorem 2. *There exists a polynomial-time constant-factor approximation for the reward-maximization dynamic stochastic matching problem.*

- *On arbitrary graphs, our algorithm computes a* $(\frac{e-1}{4e})$*-approximate matching policy.*
- *On bipartite graphs, our algorithm computes a* $(\frac{e-1}{2e})$*-approximate matching policy.*
- *In the bipartite-asymmetric setting, our algorithm computes a* $(1-\frac{1}{e})$*-approximate policy.*

Since the proof requires significant technical developments, Theorem 2 is established in Appendix B. That said, we provide a technical outline in Section 5.1 and intuitively discuss certain characteristics of our algorithm in Section 5.2.

5.1. Technical outline

Step 1: QCLP benchmark. By adapting the fluid relaxation idea of Section 3, we formulate a linear program, which will serve as a benchmark for the reward-maximization matching problem:

$$\begin{array}{llll}
(RB) & \max\limits_{x} & \sum\limits_{(i,j)\in\mathcal{T}^2} r(i,j)\cdot x_{i,j} & \\
& \text{s.t.} & \sum\limits_{j\in\mathcal{T}} x_{j,i} + \sum\limits_{j\in\mathcal{T}} x_{i,j} + x_{i,a} = \lambda_i\ , & \forall i\in\mathcal{T} \qquad (12)\\
& & \mu_i\cdot x_{i,j} \le \lambda_j\cdot x_{i,a}\ , & \forall (i,j)\in\mathcal{T}^2 \qquad (13)\\
& & x_{i,j}\ge 0 & \forall (i,j)\in\mathcal{T}^2
\end{array}$$

Let U^* designate the optimal value of (RB). As one might expect, the next claim shows that U^* is an upper-bound on the optimal expected average reward. The proof, which is presented in Appendix B.1, proceeds from a direct reduction to the cost-minimization setting of Section 3.

Claim 3. $U^*\ge r^{\pi^*}$.

While (RB) gives an upper bound on the optimal expected average reward, this LP operates a strong relaxation in regard to the arrivals of passive vertices. Indeed, as revealed by the proof of Lemma 1, the coefficient λ_j on the right-hand side of constraint (13) captures the fact that passive type-j vertices arrive at rate λ_j. In reality, however, only a fraction of the arriving type-j vertices are passive. Specifically, in the LP relaxation (RB), the arrival rate of passive type-j vertices is $\sum_{i\in\mathcal{T}} x_{i,j}$, which can be much smaller than λ_j. Motivated by this observation, we will consider an alternative mathematical program that further restricts the feasible region of (RB) by tightening constraint (13). Namely, we define a quadratically-constrained linear program (QCLP) as follows:

$$
\begin{aligned}
(QB) \qquad \max_{x} \quad & \sum_{(i,j)\in\mathcal{T}^2} r(i,j)\cdot x_{i,j} & & \\
\text{s.t.} \quad & \sum_{j\in\mathcal{T}} x_{j,i} + \sum_{j\in\mathcal{T}} x_{i,j} + x_{i,a} + x_i = \lambda_i \ , & \forall i \in \mathcal{T} \qquad & (14)\\
& \mu_i \cdot x_{i,j} \le \left(x_j + \sum_{k\in\mathcal{T}} x_{k,j} \right) \cdot x_{i,a} \ , & \forall (i,j)\in\mathcal{T}^2 \qquad & (15)\\
& x_{i,j} \ge 0 & \forall (i,j)\in\mathcal{T}^2 \qquad & \\
& x_i \ge 0 & \forall i\in\mathcal{T} \qquad &
\end{aligned}
$$

To gain intuition on this formulation, the quadratic constraint (15) can be viewed as an adjustment of constraint (13) to the "actual" arrival rate of passive type-j vertices. Here, the fixed quantity λ_j is replaced by the flow $x_j + \sum_{k\in\mathcal{T}} x_{k,j}$, which is endogenously determined by the QCLP formulation. In this context, the new variable x_j should be understood as a "buffer" of unmatched passive vertices, noting that a larger value of x_j enables us to relax constraint (15). While (QB) no longer provides an upper bound on the optimal expected average reward, the next claim shows a constant-factor loss in optimality with respect to the LP benchmark. The proof appears in Appendix B.2.

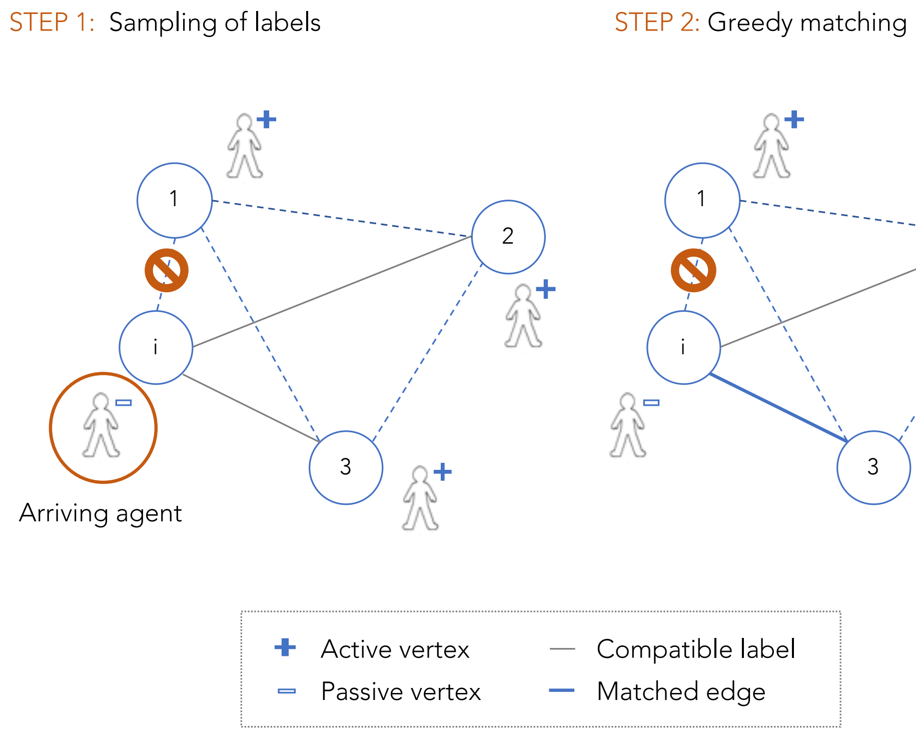
LEMMA 9. *There is a polynomial-time algorithm that computes a feasible flow $\tilde{x}$ for problem (QB) such that $\sum_{(i,j)\in\mathcal{T}^2} r(i,j)\cdot\tilde{x}_{i,j} \ge \kappa\cdot r^{\pi^*}$, where $\kappa = \frac{1}{4}$ on arbitrary graphs, $\kappa = \frac{1}{2}$ on bipartite graphs, and $\kappa = 1$ in the bipartite-asymmetric setting. In particular, we have $r^{\pi^*} \le \kappa\cdot Q^*$ in these settings.*

It is worth noting that the proof of Lemma 9 gives a simple procedure to compute a feasible flow $\tilde{x}$ for the QCLP problem (QB) using only an optimal solution of the LP relaxation (RB). In particular, our algorithm comes at no extra computational cost compared with solving this basic LP. Nonetheless, solving (QB) numerically may result in tighter instance-dependent guarantees, which exploit the structure of each instance. Lemma 9 illustrates this approach in the bipartite and bipartite-asymmetric special cases, for which we obtain better approximation ratios.

Step 2: Randomized compatibility policy (Appendix B.3). Next, we construct an algorithm that computes a randomized matching policy, dubbed *randomized compatibility*, which competes against our QCLP benchmark up to a constant factor. To keep the paper concise, we settle here for a high-level description of this algorithm, while all technical details are deferred to Appendix B.3. The main idea is to approximately simulate the matching flow obtained by solving problem (QB). To this end, a near-optimal flow $\tilde{x}$ is utilized to specify a distribution over *labels*, which are randomly assigned to the arriving vertices. Each label imposes two constraints: (i) The vertex in question is either marked as *passive* or as *active*; (ii) Each active vertex is further assigned with a *compatibility-set*, which is defined as the subset of types with which it can be matched. Next, our policy matches the arriving vertices greedily subject to the restrictions imposed by their labels. That is, the randomized compatibility policy proceeds in two steps, which are illustrated by Figure 1:

1. *Randomized labels:* A label is randomly assigned to each arriving vertex. These labels are independently sampled from carefully constructed distributions based on the flow solution $\tilde{x}$.
2. *Greedy matching:* At each epoch, the policy picks a maximum-weight matching over compatible pairs of vertices in the realization graph. That is, we prune all edges that connect vertices with incompatible labels; a compatible pair corresponds to a vertex marked as active and a vertex marked as passive, whose type resides in the active vertex's compatibility-set.

Figure 1 Illustration of the randomized compatibility matching policy.



Step 3: Analysis via virtual Markov chains (Appendix B.4). The performance guarantees stated by Theorem 2 are established by relating the expected average reward of the randomized compatibility policy to the QCLP-based upper bound of Lemma 9. While the Markov chain induced by this policy is generally difficult to analyze, we construct a modified stochastic process, termed the *virtual Markov chain*, for which the stationary distribution can be easily computed. Intuitively, this

approximation is constructed by assuming that the matching rates of vertices with different labels are mutually independent, which yields a decomposition of the system evolution into independent birth-death processes. Then, we relate the virtual Markov chain to our original stochastic process using a criterion for stochastic dominance.

5.2. Discussion: Capturing the pooling effects

From a conceptual standpoint, our approach has similarities with LP-based and simulation-based algorithms developed for related online matching problems (Manshadi et al. 2012, Jaillet and Lu 2013, Dickerson et al. 2018). These algorithms often solve an offline relaxation of the matching problem and use summary statistics of the offline solution to make randomized online decisions (e.g., an arriving vertex is randomly matched according to a distribution proportional to the flow solution). However, the dynamic properties of our model lead to fundamental differences with previous literature. Due to the risk that agents abandon the market, it is important to minimize the waiting times incurred by the matching decisions. For this purpose, there are benefits in *pooling* agents' arrivals. This notion is illustrated by the following heuristic reasoning. Suppose that we match a currently available type-1 vertex with the first type-2 or-3 vertex who arrives in the future; in other words, the arrivals of type-2 and type-3 vertices are pooled. The probability that the active vertex abandons the system is of $\frac{\mu_1}{\mu_1+\lambda_2+\lambda_3}$. In the absence of pooling, if candidate matches are restricted to type-2 vertices only, the probability of abandonment increases to $\frac{\mu_1}{\mu_1+\lambda_2}$!

As shown by this simple example, pooling effects are critical to minimize waiting times, and thereby, reduce the risk of abandonment. Standard simulation-based algorithms cannot take advantage of pooling effects; such algorithms sample (and commit to) a certain matching decision for each arriving vertex. By contrast, our algorithm does not commit to a specific match upon the arrivals of vertices. The compatibility-sets constructed by our algorithm introduce a degree of flexibility, allowing us to leverage the pooling effects amongst arriving passive vertices.

6. Negative results

In this section, we establish two negative results which foreground the technical challenges surmounted in Sections 4 and 5 to devise provably-good approximation algorithms for the dynamic stochastic matching problem. First, we demonstrate that our cost-minimization MDP has no positive constant-factor competitive ratio. Specifically, we show that the optimal value of our LP benchmark (CB) defined in Section 4.2 can be arbitrarily larger than the optimal cost of an offline benchmark that fully knows the sequence of agents' arrivals and abandonments. This result highlights the value of having access to precise information about the market dynamics. Second, we analyze the worst-case performance of a family of batching algorithms, which are widely implemented by matching platforms in practice, including ridesharing firms and kidney exchange programs (see Section 1). We show that, even if such batching algorithms are optimally tuned, their performance guarantees can be arbitrarily bad. Although our performance analysis focuses on worst-case instances, the "bad" counter-examples that we construct are relatively simple; for example, they satisfy the structural properties considered in Section 4. This suggests that our negative results might hold in certain practically relevant settings.

Offline benchmark. We compare our linear programming benchmark (CB) to offline policies, that have full knowledge of the sojourn process on each realization. Specifically, letting $\mathcal{F}_\infty = \bigcup_{t\geq 0} \mathcal{F}_t$, we denote by $\{M_t^{\rm off}\}_{t\geq 0}$ an $\mathcal{F}_\infty$-measurable family of matchings that minimizes the expected average cost $c^{\rm off} = \limsup_{t\to+\infty} \frac{\mathbb{E}[C_t^{\rm off}]}{t}$. The cumulative cost $\{C_t^{\rm off}\}_{t\geq 0}$ is defined similarly to equation (38); namely, for every $t \geq 0$, we have

$$C_t^{\rm off} = \sum_{\zeta\in\mathcal{E}(t)} c_a \cdot \left|D_\zeta^{\rm off}\right| + \sum_{\zeta\in\mathcal{E}(t)} \sum_{e\in M_\zeta^{\rm off}} c(e) \ ,$$

where $D_\zeta^{\rm off}$ is the subset of vertices that abandon the system at epoch ζ. With these definitions at hand, we show that the expected average cost $c^{\rm off}$ of the offline benchmark is generally incomparable to the optimal value of our LP benchmark.

THEOREM 3. *For every $\epsilon \in (0,1)$, there exists an instance of the dynamic stochastic matching problem, satisfying Assumption 1, such that $c^{\rm off} \leq \epsilon \cdot L_\epsilon^*$, where L_ϵ^* is the optimal value of (CB) for the corresponding instance.*

In combination with Lemma 1, the above result implies that there exists no admissible matching policy that achieves a constant-factor approximation guarantee with respect to the offline benchmark. The proof, presented in Appendix C, is based on constructing instances that have the following structure: (i) there are two distinct types of agents, referred to as *patient* and *impatient* in view of their respective abandonment rates; (ii) patient agents arrive much more frequently than impatient agents; (iii) matches amongst agents of the same type incur very small cost; (iv) unmatched abandonments are very costly. Intuitively, having access to precise information about future arrival and abandonment epochs can dramatically improve the performance of the matching platform by ensuring that a substantial fraction of the impatient agents are matched amongst each other at a very small cost, without increasing their exposure to abandonments.

Performance of batching algorithms. We now turn our attention to analyzing a family of batching algorithms for the cost-minimization formulation of our dynamic stochastic matching problem. A batching policy computes a min-cost maximum-cardinality matching at regular intervals of time. Namely, given a batching window parameter $\eta > 0$, the η-batching policy $b(\eta)$ is defined such that

$M_t^{b(\eta)} = \emptyset$ if $t \notin \{k \cdot \eta : k \in \mathbb{N}\}$, otherwise, $M_t^{b(\eta)}$ is a min-cost maximum cardinality matching in the realization graph at time t, i.e., the subgraph of G_t induced by $V_t^{b(\eta)}$. For any given instance of the dynamic stochastic matching problem, we define the *best batching policy* as a policy $b(\eta)$ that minimizes the expected average cost over all $\eta \geq 0$. In other words, we ensure that the batching parameter η is optimally tuned on each instance. The best batching policy is well-defined since we can restrict η to lie in a compact set without loss of generality.

Theorem 4. *The worst-case performance of the best batching policy is arbitrarily bad for the dynamic stochastic matching problem, even under Assumptions 1 and 2.*

Due to lengthy technical details, the proof of this theorem is deferred in Appendix D. That said, we emphasize that the bad instances that we construct have a simple structure; in particular, the cost function satisfies the triangle inequality (Assumption 1) and the agents have uniform abandonment rates (Assumption 2). We show that batching policies fail to achieve a satisfactory performance on instances for which the optimal "timing" of the matching decisions is highly heterogeneous across types. Since each batching policy utilizes a fixed time interval, it is impossible to optimally balance market thickness and abandonment risks for all agent types simultaneously. This reasoning will be empirically corroborated in Section 7 through our numerical case study.

7. Data-Driven Case Study

In this section, we conduct extensive simulations to evaluate the vertex-additive algorithm developed in Section 4.3. We take the perspective of a car-pooling platform that matches pairs of riders at the beginning of their trips to generate distance cost savings. Using the NY taxi trip data sets, we generate realistic instances that mirror various market conditions faced by car-pooling platforms.

7.1. Data overview

We utilize NY taxi trip data sets that describe the exact date and time of pick-up and drop-off as well as the GPS coordinates and trip lengths for over 28 million trips. Our case study focuses on trips recorded throughout the 8-week period of January and February 2013.[2] Most taxi trips originate from or terminate at locations in Manhattan. Besides Manhattan, the most common pick-up and drop-off locations are LaGuardia and John F. Kennedy airports. To ensure an exhaustive coverage of the taxi network, all regions are incorporated into our case study. We conduct minimal pre-processing of the data by eliminating inaccurate and noisy observations, as further explained in Appendix E.1.

We observe that the demand patterns for taxis are seasonal with significant day-of-week and time-of-day effects (see Figure 7 in Appendix E.4). Consequently, we focus our subsequent analysis

Table 1 Summary statistics of the data sets generated for each time window.

Day of week	Time of day	Number of types $\|\mathcal{T}\|$	Sample size Training set	Test set
Monday	7:30 AM - 8:00 AM	272	50988	9022
	11:00 AM - 11:30 AM	272	48484	7064
Saturday	7:30 AM - 8:00 AM	218	15036	2535
	5:30 PM - 6:00 PM	307	70035	10275

on four time windows that represent various market conditions: *Monday 7:30-8:00AM*, corresponding to commute time; *Saturday 5:30-6:00PM*, corresponding to weekend peak time, as well as *Monday 11:00-11:30AM* and *Saturday 7:30-8:00AM*, describing non-peak times during week days and weekends, respectively. Additionally, we split the data into training sets (comprised of the first 7 weeks of data) and test sets (comprised of the last week of data). The parameters of the problem instances are estimated on the training sets, and the performance of the matching algorithms is measured on the test sets. The statistics of the resulting data sets are summarized in Table 1.

7.2. Simulation set-up

Rider types. Our modeling approach hinges on defining a collection of types $i \in \mathcal{T}$ and specifying their arrival rates λ_i. Intuitively, each type should describe "similar" riders in terms of pick-up and drop-off locations. To formalize this notion, we develop a data-driven clustering method that identifies riders with close-enough pick-up and drop-off locations. The specifics of this method appear in Appendix E.2. At high-level, we build on the approach developed by Buchholz (2021) to decompose the NY area into "homogeneous" regions using census-tract boundaries. The resulting 33 regions are visualized in Figure 5 in Appendix E.2. However, certain combinations of pick-up and drop-off regions exhibit a very small number of rider requests at certain times of day and days of week. For example, for 27% of the pairs of pick-up and drop-off regions, there are fewer than 8 trips recorded throughout our 8-week time period. Hence, we cluster the pairs of pick-up and drop-off regions to deal with data sparsity. As shown by Figure 6 in Appendix E.2, *all* resulting clusters contain more than 8 trips. Consequently, we view each cluster as a distinct rider type $i \in \mathcal{T}$, and the corresponding arrival rate λ_i is estimated on the training set.

Simulation inputs. We define a *simulation trial* as one iteration of the matching process based on the following inputs:

- *A collection of riders* $\{1, \dots, r\}$*:* Riders are formed by all historical trips recorded during the time window forming the test set.
- *Arrival times* $\{t_1, \dots, t_r\}$*:* The arrival times are determined using the timestamp corresponding to the starting time of the trip.
- *Sojourn times* $\{\delta_1, \dots, \delta_r\}$ *:* The sojourn times are i.i.d. samples drawn from an exponential distribution of rate μ, where μ is varied in the set $\{0.5, 1, 2\}$ (arrivals per minute).

- *Trip lengths* $\{\ell_1, \dots \ell_r\}$ *:* The trip length ℓ_n associated with rider $n \in [r]$ is the Euclidean distance between her pick-up and drop-off locations.
- *Matching costs:* The matching cost $c(n_1, n_2)$ for a pair of riders $\{n_1, n_2\}$ is defined as the minimum total Euclidean distance[3] over all possible vehicle routes formed by sequencing the riders' pick-ups and drop-offs (we provide a pictorial illustration of all possible routes in Figure 8, Appendix E.4). In particular, Assumption 1 is satisfied by the specified cost function since $c(n_1, n_2) \le \ell_{n_1} + \ell_{n_2} \le c(n_1, n_3) + c(n_3, n_2)$ for every $n_1, n_2, n_3 \in [r]$.
- *Abandonment costs:* The abandonment cost of rider $n \in [r]$ is defined as $c_a(n) = c_p + \ell_n$, where $c_p > 0$ is a constant penalty. This notion roughly monetizes the opportunity cost for the platform in case rider n abandons the system (see Garg and Nazerzadeh (2021) for further background on affine pricing mechanisms in ridesharing).

Performance metrics. Based on the above-described inputs, we simulate the sequence of events formed by the arrivals of riders, the matching decisions made by the platform, and the abandonments of unmatched riders. Consequently, we compute the total cost C generated by the matching algorithms on the simulation trial, namely $C = \sum_{\zeta \in \mathcal{E}} \sum_{e \in M_\zeta} c(e) + \sum_{\zeta \in \mathcal{E}} \sum_{n \in D_\zeta} c_a(n)$, where $\mathcal{E}$ is formed by all arrival, abandonment and decision epochs, M_ζ is the matching picked at epoch ζ, and D_ζ is the collection of unmatched rider who abandon at epoch ζ. For every match $\{n_1, n_2\}$, we define the *cost saving* $s(n_1, n_2) = \ell_{n_1} + \ell_{n_2} - c(n_1, n_2)$. This notion quantifies how much driver time is saved by "pooling" riders n_1 and n_2, and thus, it is a key indicator of the car-pooling platform's performance. Indeed, the cost C can be expressed as an affine function of the *match rate* and the *saving rate* defined as follows:

$$C = \alpha_1 + \alpha_2 - \alpha_1 \cdot \underbrace{\left(1 - \frac{\sum_{\zeta \in \mathcal{E}} |D_\zeta|}{r}\right)}_{\text{Match rate}} - \alpha_2 \cdot \underbrace{\left(\frac{\sum_{\zeta \in \mathcal{E}} \sum_{e \in M_\zeta} s(e)}{\sum_{n=1}^{r} \ell_n}\right)}_{\text{Saving rate}}, \tag{16}$$

where $\alpha_1 = c_p \cdot r$ and $\alpha_2 = \sum_{n=1}^{r} \ell_n$. By noting that α_1, α_2 do not depend on the platform's decisions, without loss of generality, the algorithms' performance is uniquely determined by the corresponding match rate and saving rate. Intuitively, by making a rider wait longer, the platform can find a more suitable match unless the rider in question abandons the system. As such, we expect that larger savings generally come at the expense of lower match rates (i.e., higher risks of abandonment).

7.3. Tested algorithms

We compare the performance of three matching algorithms: batching, vertex-additive and threshold-based. We utilize a tuning parameter to control the trade-off between saving rates and match rates. By varying this tuning parameter, we generate the *Pareto frontier* attained by each matching algorithm with respect to these criteria. Note that we only implement the vertex-additive algorithm of Section 4.3 since the formulation of a cost-minimization problem and the use of a deterministic matching algorithm are more natural in the context of car-pooling platforms.

Batching algorithm. We refer the reader to Appendix D for a mathematical description of the batching algorithm. Informally speaking, the batching algorithm computes a min-cost maximum-cardinality matching over all available riders, at regular intervals of time. The tuning parameter of the algorithm is the batching window $\eta_b \in \{0, 0.5, 1, 2, 3, \ldots, 29, 45, 60\}$, corresponding to the length of each time interval (in seconds). Longer batching windows might result in higher-saving matches, but riders are more likely to abandon the platform.

Vertex-additive algorithm. In Section 4.3, the vertex-additive policy is defined as the greedy policy with respect to the additive approximation of the cost-to-go $\sum_{i\in\mathcal{T}} \nu_i \bar{c}_i$, where $\nu \in \mathcal{V}$ is the state of the MDP, and $\bar{c}_i$ is the optimal expected cost for the type-i active-vertex problem. We implement an analogous greedy algorithm using a scaled variant of the cost-to-go $\eta_v \cdot (\sum_{i\in\mathcal{T}} \nu_i \bar{c}_i)$, where $\eta_v \in \{0.7, 0.72, \ldots, 1.28, 1.3\}$ is the tuning parameter. Intuitively, we expect that riders' waiting times will decrease as a function of η_v. As explained in Appendix E.3, for every $i \in \mathcal{T}$, we compute $\bar{c}_i$ by employing a binary search algorithm on a normalized variant of the cost function.

Threshold-based algorithm. Lastly, we implement a threshold-based algorithm, motivated by the optimality of the $\bar{c}_i$-threshold policy for the type-i active-vertex problem (see Theorem 2 in Section 4.1). Specifically, upon the arrival of a rider $n \in [r]$, we define the subset of *eligible* riders $V_E = \{m \in V : c(n,m) \leq \eta_t \times \bar{c}_{\theta_m}\}$ out of all available riders V, where $\eta_t \in \{0.7, 0.72, \ldots, 1.98, 2\}$ is the tuning parameter of the threshold-based algorithm. Next, we compute a min-cost maximum-cardinality matching over $V_E \cup \{n\}$ to resolve the potential contention between multiple eligible riders.

7.4. Results

The performance metrics are reported in Figure 2. Specifically, for each algorithm and each tuning parameter value, we compute the average saving rate and match rate over 10 randomly-generated simulation trials. We observe that the vertex-additive algorithm achieves substantial Pareto improvements relative to the batching and threshold-based algorithms. As illustrated by Figure 3, given a match rate target of 90%, the vertex-additive algorithm achieves a 35% saving rate, which represents an improvement of nearly 10% (in absolute terms) over the incumbent matching mechanism. While batching achieves comparable performance when the match rate is high (corresponding to batching windows in the order of a few seconds), the performance devolves for match rates below 95%. In this regime, increasing the batching window has almost no effect on the level of savings, which appears to be capped at approximately 23%. In contrast, there is a clear-cut trade-off between the match rate and the saving rate under the vertex-additive and threshold-based algorithms – almost *all* configurations generated by varying the tuning parameters lie on the Pareto frontier, unlike the batching algorithm. We also compare the performance of the proposed algorithms with our LP lower bound. We find that the vertex-additive algorithm has a gap of 16%-21% with respect to the LP benchmark, which is much tighter than the factor-3 approximation ratio yielded by our analysis (see Table 2 in Appendix E.4).

At first glance, these results are surprising since the vertex-additive algorithm can be implemented in a "decentralized" fashion, using only local information about the first-degree neighbors of each vertex, while batching is a "centralized" algorithm that fully exploits the structure of the realization graph. That said, our newly developed algorithms have the ability to adjust the riders' exposure to waiting times based on their type, while the batching algorithm makes riders wait uniformly, irrespective of their type. This notion of market segmentation is corroborated by Figures 9 and 10 in Appendix E.4. Riders requesting short trips wait relatively less under the vertex-additive algorithm compared to batching, while riders requesting long trips wait relatively more.

Figure 2 Match rates and saving rates achieved by the tested algorithms in various market conditions.

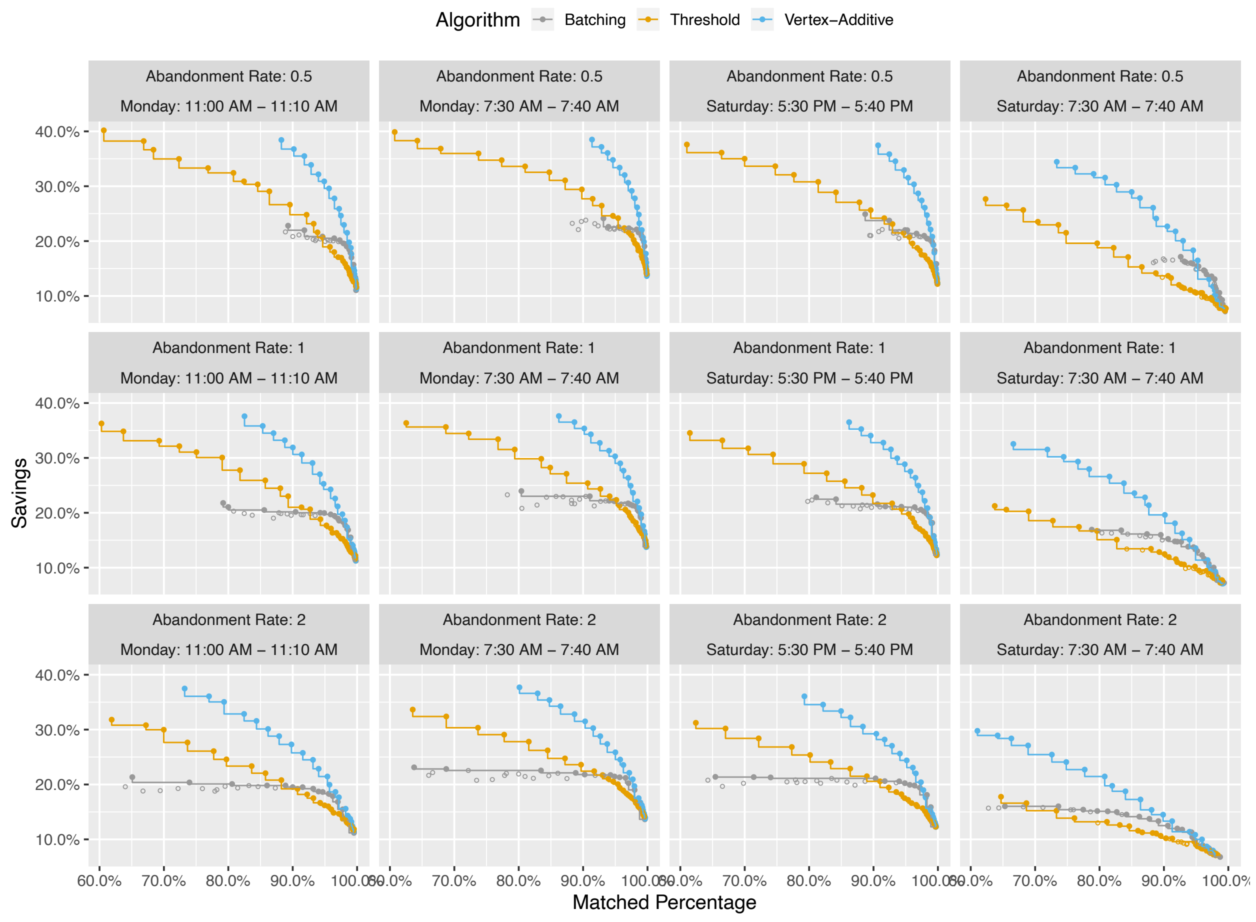
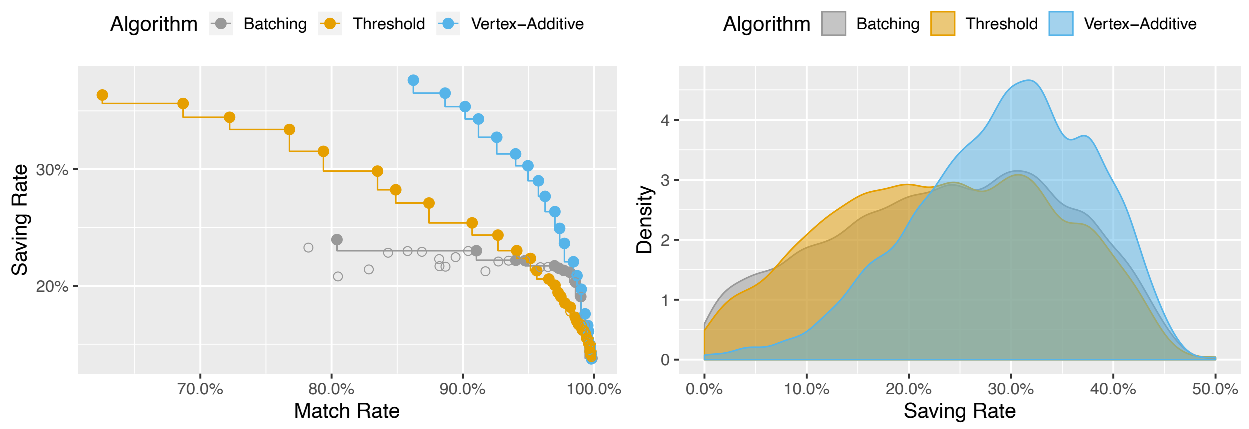


Figure 3 On the left, match rates and saving rates achieved by the tested algorithms (*Mondays 7:30-7:40 AM*, $\mu = 1$). On the right, empirical density function of the saving rates over the population of riders.



Note. Each algorithm is tuned to achieve a match rate of 90%(±1%).

We conduct various robustness checks to verify whether the improvements in performance are detrimental to certain categories of riders. First, we examine the distribution of savings over the population of all riders. As shown by Figure 3, the variance in saving rates is significantly lowered by the vertex-additive algorithm, meaning that our proposed matching mechanism would provide a more reliable experience to riders. Second, we break-down the performance with respect to short vs. long trips. Since long trips often originate from sparse suburban areas, this granular analysis is important to ensure that the newly developed matching algorithms do not generate unfair spatial disparities. As shown by Figure 11 in Appendix E.4, the vertex-additive consistently generates higher levels of savings than the other matching algorithms irrespective of trip length.

8. Concluding Remarks

In summary, this research work advances the development of approximation algorithms for dynamic stochastic matching problems. Our results leave several interesting directions for future research.

Open question 1: Performance of batching algorithms. Theorem 4 shows certain limitations of batching policies. That said, the family of bad instances that we construct has two noteworthy characteristics: (i) The underlying graph is non-bipartite and optimal matches involve agents of identical types. While this structure is very relevant to car-pooling systems, it might not be realistic for other types of matching decisions made by ridesharing platforms, such as matching riders with drivers. (ii) Our negative result crucially relies on a min-cost formulation of the dynamic stochastic matching problem. To our knowledge, it is unknown whether the family of batching policies attains a constant-factor approximation for the reward-maximization problem in its utmost generality. Broadly speaking, understanding the performance of batching policies on specific graph-theoretic structures and objective functions is an interesting direction for future research.

Open question 2: Improved approximation ratios. Important theoretical research questions include whether the performance analyses of Theorems 1 and 2 are tight and whether it is possible to devise algorithms attaining better approximation ratios. One starting point would be to further investigate the bipartite-asymmetric special case of Theorem 2, a setting for which we obtain a $(1-\frac{1}{e})$-approximation. Moreover, we conjecture that our algorithm for the reward-maximization setting might be asymptotically optimal in the large market regime of Özkan and Ward (2020). As discussed in Remark 2 of Section 3, the constraints relative to abandonments are asymptotically relaxed when type-i vertices arrive at a scaled rate $\theta \cdot \lambda_i$ as θ tends to infinity. In this context, it would be interesting to check whether or not our randomized compatibility policy converges to the myopic randomized policy of Özkan and Ward (2020).

Applications to ridesharing. From a modeling perspective, our work captures two characteristics of dynamic matching markets that are relevant to the ridesharing sector. First, we assume that the platform has a stochastic prior on the market dynamics, captured by the arrival and abandonment rates. In practice, ridesharing platforms collect large amounts of historical data from which they can estimate these parameters. Second, the agents' abandonments are unannounced: namely, our work bypasses the criticality assumption used in previous literature. This model seems more realistic for ridesharing platforms, whereby an abandonment corresponds to a rider cancellation. Nonetheless, our approach still presents a number of important limitations. Key to our results is the assumption that the sojourn times are exponentially distributed. While this assumption is realistic in certain regimes, one might expect that riders are relatively more likely to cancel at the beginning and end of the time window quoted by the platform. An equally interesting direction is to study hypergraph matching models, in which more than two agents can be matched together. These issues have received limited attention in the literature so far.

References

Adan I, Weiss G (2012) Exact FCFS matching rates for two infinite multitype sequences. *Operations research* 60(2):475–489.

Adan I, Weiss G (2014) A skill based parallel service system under FCFS-ALIS—steady state, overloads, and abandonments. *Stochastic Systems* 4(1):250–299.

Afeche P, Caldentey R, Gupta V (2021) On the optimal design of a bipartite matching queueing system. *Operations Research (Forthcoming)* .

Akbarpour M, Li S, Gharan SO (2020) Thickness and information in dynamic matching markets. *Journal of Political Economy* 128(3).

Alonso-Mora J, Samaranayake S, Wallar A, Frazzoli E, Rus D (2017) On-demand high-capacity ride-sharing via dynamic trip-vehicle assignment. *Proceedings of the National Academy of Sciences* 114(3):462–467.

Anderson R, Ashlagi I, Gamarnik D, Kanoria Y (2017) Efficient dynamic barter exchange. *Operations Research* 65(6):1446–1459.

Ashlagi I, Azar Y, Charikar M, Chiplunkar A, Geri O, Kaplan H, Makhijani R, Wang Y, Wattenhofer R (2017) Min-cost bipartite perfect matching with delays. *Approximation, Randomization, and Combinatorial Optimization (APPROX/RANDOM 2017).*

Ashlagi I, Bingaman A, Burq M, Manshadi V, Gamarnik D, Murphey C, Roth AE, Melcher ML, Rees MA (2018) Effect of match-run frequencies on the number of transplants and waiting times in kidney exchange. *American Journal of Transplantation* 18(5):1177–1186.

Ashlagi I, Burq M, Dutta C, Jaillet P, Saberi A, Sholley C (2019) Edge weighted online windowed matching. *Proceedings of the 2019 ACM Conference on Economics and Computation*, 729–742 (ACM).

Baccara M, Lee S, Yariv L (2020) Optimal dynamic matching. *Theoretical Economics* 15(3):1221–1278.

Banerjee S, Kanoria Y, Qian P (2018) Dynamic assignment control of a closed queueing network under complete resource pooling. Technical report, Working paper, arXiv:1803.04959.

Bansal N, Buchbinder N, Gupta A, Naor JS (2007) An $O(\log 2k)$-competitive algorithm for metric bipartite matching. *European Symposium on Algorithms*, 522–533 (Springer).

Bertsekas DP, Tsitsiklis JN (1996) *Neuro-dynamic programming*, volume 5 (Athena Scientific Belmont, MA).

Bertsimas D, Jaillet P, Martin S (2019) Online vehicle routing: The edge of optimization in large-scale applications. *Operations Research* 67(1):143–162.

Brandt A, Last G (1994) On the pathwise comparison of jump processes driven by stochastic intensities. *Mathematische Nachrichten* 167(1):21–42.

Buchholz N (2021) Spatial equilibrium, search frictions and efficient regulation in the taxi industry. *Review of Economic Studies (Forthcoming)* .

Cadas A, Bušić A, Doncel J (2019) Optimal control of dynamic bipartite matching models. *Proceedings of the 12th EAI International Conference on Performance Evaluation Methodologies and Tools*, 39–46 (ACM).

Caldentey R, Kaplan EH, Weiss G (2009) FCFS infinite bipartite matching of servers and customers. *Advances in Applied Probability* 41(3):695–730.

Collina N, Immorlica N, Leyton-Brown K, Lucier B, Newman N (2020) Dynamic weighted matching with heterogeneous arrival and departure rates. *Proceedings of the 2020 Web and Internet Economics (WINE)* .

Devanur NR, Hayes TP (2009) The adwords problem: online keyword matching with budgeted bidders under random permutations. *Proceedings of the 10th ACM conference on Electronic commerce*, 71–78 (ACM).

Dickerson J, Sankararaman K, Srinivasan A, Xu P (2018) Allocation problems in ride-sharing platforms: Online matching with offline reusable resources. *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Feldman J, Mehta A, Mirrokni V, Muthukrishnan S (2009) Online stochastic matching: Beating 1-1/e. *2009 50th Annual IEEE Symposium on Foundations of Computer Science*, 117–126 (IEEE).

Gallego G, Ratliff R, Shebalov S (2014) A general attraction model and sales-based linear program for network revenue management under customer choice. *Operations Research* 63(1):212–232.

Garg N, Nazerzadeh H (2021) Driver surge pricing. *Management Science (Forthcoming)* .

Goel G, Mehta A (2008) Online budgeted matching in random input models with applications to Adwords. *Proceedings of the nineteenth annual ACM-SIAM symposium on Discrete algorithms*, 982–991 (Society for Industrial and Applied Mathematics).

Guo X, Hernández-Lerma O (2009) Continuous-time markov decision processes. *Continuous-Time Markov Decision Processes*, 9–18 (Springer).

Gupta A, Guruganesh G, Peng B, Wajc D (2019) Stochastic online metric matching. *arXiv preprint arXiv:1904.09284* .

Gurvich I, Ward A (2014) On the dynamic control of matching queues. *Stochastic Systems* 4(2):479–523.

Hu M, Zhou Y (2021) Dynamic type matching. *Manufacturing & Service Operations Management (Forthcoming)* .

Huang Z, Kang N, Tang ZG, Wu X, Zhang Y, Zhu X (2018) How to match when all vertices arrive online. *Proceedings of 50th Annual ACM SIGACT Symposium on Theory of Computing*, 17–29 (ACM).

Huang Z, Peng B, Tang ZG, Tao R, Wu X, Zhang Y (2019) Tight competitive ratios of classic matching algorithms in the fully online model. *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*, 2875–2886 (SIAM).

Jaillet P, Lu X (2013) Online stochastic matching: New algorithms with better bounds. *Mathematics of Operations Research* 39(3):624–646.

Kalyanasundaram B, Pruhs K (1993) Online weighted matching. *Journal of Algorithms* 14(3):478–488.

Karp RM, Vazirani UV, Vazirani VV (1990) An optimal algorithm for on-line bipartite matching. *Proceedings of the twenty-second annual ACM symposium on Theory of computing*, 352–358 (ACM).

Khuller S, Mitchell SG, Vazirani VV (1994) On-line algorithms for weighted bipartite matching and stable marriages. *Theoretical Computer Science* 127(2):255–267.

López FJ, Martínez S, Sanz G (2000) Stochastic domination and markovian couplings. *Advances in Applied Probability* 32(4):1064–1076.

Lyft (2016) Matchmaking in lyft line: Part 1. URL https://eng.lyft.com/matchmaking-in-lyft-line9c2635fe62c4.

Ma W, Simchi-Levi D (2020) Algorithms for online matching, assortment, and pricing with tight weight-dependent competitive ratios. *Operations Research* 68(6).

Manshadi VH, Gharan SO, Saberi A (2012) Online stochastic matching: Online actions based on offline statistics. *Mathematics of Operations Research* 37(4):559–573.

Mehta A, Saberi A, Vazirani U, Vazirani V (2005) Adwords and generalized on-line matching. *46th Annual IEEE Symposium on Foundations of Computer Science (FOCS'05)*, 264–273 (IEEE).

Mehta A, et al. (2013) Online matching and ad allocation. *Foundations and Trends in Theoretical Computer Science* 8(4):265–368.

Özkan E, Ward AR (2020) Dynamic matching for real-time ride sharing. *Stochastic Systems* 10(1):29–70.

Puterman ML (2014) *Markov Decision Processes.: Discrete Stochastic Dynamic Programming* (John Wiley & Sons).

Santi P, Resta G, Szell M, Sobolevsky S, Strogatz SH, Ratti C (2014) Quantifying the benefits of vehicle pooling with shareability networks. *Proceedings of the National Academy of Sciences* 111(37):13290–13294.

Talluri K, van Ryzin G (2004) Revenue management under a general discrete choice model of consumer behavior. *Management Science* 50(24):15–33.

Talreja R, Whitt W (2008) Fluid models for overloaded multiclass many-server queueing systems with first-come, first-served routing. *Management Science* 54(8):1513–1527.

Topaloglu H (2013) Joint stocking and product offer decisions under the multinomial logit model. *Production and Operations Management* 22(5):1182–1199.

Truong VA, Wang X (2019) Prophet inequality with correlated arrival probabilities, with application to two sided matchings. *arXiv preprint arXiv:1901.02552* .

Tsitsiklis JN, Xu K (2017) Flexible queueing architectures. *Operations Research* 65(5):1398–1413.

Wolff RW (1982) Poisson arrivals see time averages. *Operations Research* 30(2):223–231.

Yan C, Zhu H, Korolko N, Woodard D (2020) Dynamic pricing and matching in ride-hailing platforms. *Naval Research Logistics (NRL)* 67(8):705–724.

Appendix A: Additional Proofs

A.1. Existence of an optimal policy

Guo and Hernández-Lerma (2009, Thm. 5.9) identify optimality conditions for average-cost continuous-time MDPs with infinitesimal cost functions. We argue that our dynamic stochastic matching problem can be reduced to an average-cost continuous-time MDP with a continuous cost function. Specifically, for every state $\nu \in \mathcal{V}$, and for every action $M \in \mathbb{N}^{E(\mathcal{T})\cup S(\mathcal{T})}$ describing a matching over the vertices available at state ν, we define the cost $c(\nu, \cdot)$ as follows:

$$c(\nu, M) = \left(\sum_{i\in\mathcal{T}} \nu_i \cdot \mu_i + \sum_{i\in\mathcal{T}} \lambda_i\right) \cdot \left(\sum_{\{i,j\}\in E(\mathcal{T})\cup S(\mathcal{T})} M_{i,j} \cdot c(i,j)\right) .$$

Picking the action M at state ν means that the vertices matched by M incur cost at rate $c(\nu, M)$ until the vertices leave the system at the next decision epoch. The asymptotic equivalence between our original lump sum cost MDP and the continuous cost MDP follows from Wald's equation. The regularity conditions in Guo and Hernández-Lerma (2009, Thm. 5.9) are met since the cost rate is uniformly bounded by a function that only depends on the current state, namely $c(\nu, M) = O(|\nu|^2)$. $\square$

A.2. Proof of Claim 1

Fix $(i,j) \in \mathcal{T}^2$ and $q \in \mathbb{N}^*$. By definition, the matches counted by $\hat{A}_q(i,j)$ are a superset of those counted by $A^*_{\zeta_q}(i,j)$. Hence, we have $A^*_{\zeta_q}(i,j) \le \hat{A}_q(i,j)$ for each realization of the stochastic process. To establish a reciprocal inequality, observe that $\hat{A}_q(i,j) - A^*_{\zeta_q}(i,j)$ is at most the number of type-i vertices available at time ζ_q, meaning that $\hat{A}_q(i,j) - A^*_{\zeta_q}(i,j) \le \vartheta_i(V^{\pi^*}_{\zeta_q})$. It immediately follows that

$$\mathbb{E}\left[A^*_{\zeta_q}(i,j)\right] \le \mathbb{E}\left[\hat{A}_q(i,j)\right] \le \mathbb{E}\left[A^*_{\zeta_q}(i,j)\right] + \mathbb{E}\left[\vartheta_i(V^{\pi^*}_{\zeta_q})\right] . \tag{17}$$

Clearly, $\{\vartheta(V^{\pi^*}_{\zeta_q})\}_{q\ge 0}$ admits a limiting distribution as $q \to +\infty$ since the matching policy π^* induces an ergodic embedded Markov chain. Hence, the sequence $(\frac{1}{\zeta_q} \cdot \mathbb{E}[\vartheta_i(V^{\pi^*}_{\zeta_q})])_{q\in\mathbb{N}^*}$ converges almost surely to 0. By inequality (17), we conclude that the sequence $(\frac{1}{\zeta_q} \cdot \mathbb{E}[\hat{A}_q(i,j)])_{q\in\mathbb{N}^*}$ converges almost surely to $x^*_{i,j}$. $\square$

A.3. Proof of Claim 2

By the optimality of $\mathcal{A}^*_i$, it is clear that $c(i,j) \ge f_i(\mathcal{A}^*_i)$ for every $j \in \mathcal{T} \setminus \mathcal{A}^*_i$; otherwise, we would obtain $f_i(\mathcal{A}^*_i \cup \{j\}) < f_i(\mathcal{A}^*_i)$ by noting that $f_i(\mathcal{A}^*_i \cup \{j\})$ is a strict convex combination of $f_i(\mathcal{A}^*_i)$ and $c(i,j)$, thereby contradicting the optimality of $\mathcal{A}^*_i$. Similarly, for every $j \in \mathcal{A}^*_i$, we have $c(i,j) \le f_i(\mathcal{A}^*_i)$, otherwise we would obtain $f_i(\mathcal{A}^*_i \setminus \{j\}) < f_i(\mathcal{A}^*_i)$, which contradicts the optimality of $\mathcal{A}^*_i$. $\square$

A.4. Proof of Lemma 3

Case 2: $f_i(\mathcal{A}_i^*) < c(i,\nu)$. Using the same line of reasoning as for Case 1, we begin by bounding the terms $\hat{F}(\nu + e_j)$ and $\hat{F}(\nu - e_j)$ for every $j \in \mathcal{T}$. For every $j \in \mathcal{T} \setminus \mathcal{A}_i^*$, we have

$$\hat{F}(\nu + e_j) = \min\{f_i(\mathcal{A}_i^*), c(i, \nu + e_j)\} = \min\{f_i(\mathcal{A}_i^*), c(i,\nu), c(i,j)\} = f_i(\mathcal{A}_i^*) \ ,$$

where the last inequality proceeds from Claim 2 and our case hypothesis. Similarly, for every $j \in \mathcal{A}_i^*$, we obtain

$$\hat{F}(\nu + e_j) = \min\{f_i(\mathcal{A}_i^*), c(i,\nu), c(i,j)\} = c(i,j) \ .$$

Lastly, for every $j \in \mathcal{T}$,

$$\hat{F}(\nu - e_j) = \min\{f_i(\mathcal{A}_i^*), c(i, \nu - e_j)\} = f_i(\mathcal{A}_i^*) \ ,$$

where the last equality holds since $f_i(\mathcal{A}_i^*) < c(i,\nu) \le c(i, \nu - e_j)$ by our case hypothesis. By combining these equations, we obtain

$$\begin{aligned}
\hat{F}(\nu) &= \min\{f_i(\mathcal{A}_i^*), c(i,\nu)\} \\
&= f_i(\mathcal{A}_i^*) \\
&= \frac{\mu + \sum_{k \in \mathcal{A}_i^*} \lambda_k}{\mu \cdot |\nu| + \mu + \sum_{k \in \mathcal{T}} \lambda_k} \left(\frac{\mu}{\mu + \sum_{k \in \mathcal{A}_i^*} \lambda_k} \cdot c_a(i) + \sum_{j \in \mathcal{A}_i^*} \frac{\lambda_j}{\mu + \sum_{k \in \mathcal{A}_i^*} \lambda_k} \cdot c(i,j) \right) \\
&\quad + \frac{\mu \cdot |\nu| + \sum_{k \in \mathcal{T} \setminus \mathcal{A}^*} \lambda_k}{\mu \cdot |\nu| + \mu + \sum_{k \in \mathcal{T}} \lambda_k} \cdot f_i(\mathcal{A}_i^*) \\
&= \frac{1}{\mu \cdot |\nu| + \mu + \sum_{k \in \mathcal{T}} \lambda_k} \cdot \left(\mu \cdot c_a(i) + \sum_{j \in \mathcal{T}} \lambda_j \cdot \hat{F}(\nu + e_j) + \sum_{j \in \mathcal{T}} \mu \cdot \nu_j \cdot \hat{F}(\nu - e_j) \right) \ .
\end{aligned}$$

It ensues that the Bellman equation (6) is satisfied with respect to the state ν. $\square$

A.5. Proof of Lemma 5

For every $i, j \in \mathcal{T}$, we have:

$$\begin{aligned}
\bar{c}_i &= \frac{\mu}{\mu + \sum_{k \in \mathcal{A}_i^*} \lambda_k} \cdot c_a(i) + \sum_{h \in \mathcal{A}_i^*} \frac{\lambda_h}{\mu + \sum_{k \in \mathcal{A}_i^*} \lambda_k} \cdot c(i,h) \\
&\le \frac{\mu}{\mu + \sum_{k \in \mathcal{A}_j^*} \lambda_k} \cdot c_a(i) + \sum_{h \in \mathcal{A}_j^*} \frac{\lambda_h}{\mu + \sum_{k \in \mathcal{A}_j^*} \lambda_k} \cdot c(i,h) \\
&\le \frac{\mu}{\mu + \sum_{k \in \mathcal{A}_j^*} \lambda_k} \cdot c_a(j) + \frac{\mu}{\mu + \sum_{k \in \mathcal{A}_j^*} \lambda_k} \cdot c(i,j) + \sum_{h \in \mathcal{A}_j^*} \frac{\lambda_h}{\mu + \sum_{k \in \mathcal{A}_j^*} \lambda_k} \cdot c(j,h) \\
&\quad + \sum_{h \in \mathcal{A}_j^*} \frac{\lambda_h}{\mu + \sum_{k \in \mathcal{A}_j^*} \lambda_k} \cdot c(i,j) \\
&= \bar{c}_j + c(i,j) \ ,
\end{aligned}$$

where the first equation proceeds from (10). The first inequality holds since $\mathcal{A}_i^*$ minimizes the function $f_i(\cdot)$ by (10). The second inequality immediately proceeds from Assumption 1. $\square$

A.6. Proof of Lemma 8

Fix a vertex $n \in \mathbb{N}^*$ and a type $i \in \mathcal{T}$. By the formula of conditional expectations, we have:

$$\mathbb{E}\left[\alpha^{\bar{\pi}}(n) \middle| \theta_n = i\right] = \sum_{\nu_0 \in \mathcal{V}} \mathbb{E}\left[\alpha^{\bar{\pi}}(n) \middle| \theta_n = i, \vartheta(V^{\bar{\pi}}_{t_n}) = \nu_0\right] \cdot \Pr\left[\vartheta(V^{\bar{\pi}}_{t_n}) = \nu_0 \middle| \theta_n = i\right] \ .$$

Hence, it suffices to show that $\mathbb{E}[\alpha^{\bar{\pi}}(n)| \theta_n = i, \vartheta(V^{\bar{\pi}}_{t_n}) = \nu_0] \leq \bar{c}_i$ for all $\nu_0 \in \mathcal{V}$. In what follows, we fix $\nu_0 \in \mathcal{V}$ and define the event $E = \{\theta_n = i, \vartheta(V^{\bar{\pi}}_{t_n}) = \nu_0\}$. Since the vertex-additive policy can only match a passive vertex upon its arrival, the event E fully determines whether n is active or passive. If n is passive conditional on E, the definition of the allocated cost immediately implies $\mathbb{E}[\alpha^{\bar{\pi}}(n)| E] = \bar{c}_i$. Hence, the remainder of the proof focuses on the case where n is active conditional on the event E.

Now, let ℓ be the first random vertex arriving after n of type $\theta_\ell \in \mathcal{A}^*_i = \{j \in \mathcal{T} : c(i,j) \leq \bar{c}_i\}$, and let τ be the random time at which n leaves the system, either due to a match or by abandoning the system. It is worth noting that the vertex-additive policy can potentially match n with a vertex of type $j \in \mathcal{T} \setminus \mathcal{A}^*_i$. Nonetheless, the next claim highlights a key property of this policy, showing that if n is still available when ℓ arrives, our policy necessarily picks the match $\{n, \ell\}$ at time t_ℓ.

Claim 4. $\Pr\left[M^{\bar{\pi}}_{t_\ell} = \{n, \ell\} | E, n \in V^{\bar{\pi}}_{t_\ell}\right] = 1$.

The proof is deferred to Appendix A.7. By noting that Claim 4 implies that $\Pr[\tau \leq \min\{t_\ell, t_n + \delta_n\}] = 1$, we further decompose the expected allocated cost as follows:

$$\begin{aligned} \mathbb{E}\left[\alpha^{\bar{\pi}}(n)| E\right] = & \Pr\left[\tau < \min\{t_\ell, t_n + \delta_n\}\right] \cdot \mathbb{E}\left[\alpha^{\bar{\pi}}(n)| E, \tau < \min\{t_\ell, t_n + \delta_n\}\right] \\ & + \Pr\left[\tau = \min\{t_\ell, t_n + \delta_n\}\right] \cdot \mathbb{E}\left[\alpha^{\bar{\pi}}(n)| E, \tau = \min\{t_\ell, t_n + \delta_n\}\right] \ . \end{aligned} \tag{18}$$

The remainder of the proof consists in separately upper bounding the expected allocated costs $\mathbb{E}[\alpha^{\bar{\pi}}(n)|E, \tau < \min\{t_\ell, t_n + \delta_n\}]$ and $\mathbb{E}[\alpha^{\bar{\pi}}(n)|E, \tau = \min\{t_\ell, t_n + \delta_n\}]$. The former quantity corresponds to the case of an *early match*, meaning that n is matched with agent of type $j \in \mathcal{T} \setminus \mathcal{A}^*_i$. The latter quantity corresponds to the case of a *late match attempt*, meaning that either n is matched with ℓ or it abandons before t_ℓ.

Case 1: Upper bound conditional to an early match. Observe that every match $\{n, m\}$ picked by the vertex-additive policy necessarily satisfies $c(n, m) \leq \bar{c}_{\theta_n} + \bar{c}_{\theta_m}$, otherwise eliminating the match $\{n, m\}$ is cost-wise beneficial with respect to the linear approximation of the cost-to-go. Hence, by definition of the allocated cost, it follows that $\alpha^{\bar{\pi}}(n) \leq \bar{c}_i$ almost surely conditional on n being matched. It immediately follows that

$$\mathbb{E}\left[\alpha^{\bar{\pi}}(n)| E, \tau < \min\{t_\ell, t_n + \delta_n\}\right] \leq \bar{c}_i \ .$$

Case 2: Upper bound conditional to a late match attempt. Now, to upper bound the second term on the right-hand side of equation (18), we observe that

$$\begin{aligned}
&\mathbb{E}\left[\alpha^{\bar{\pi}}(n) \middle| E, \tau = \min\{t_\ell, t_n + \delta_n\}\right] \\
&\quad = \sum_{j \in \mathcal{A}_i^*} \Pr\left[\theta_\ell = j, \tau = t_\ell \middle| E, \tau = \min\{t_\ell, t_n + \delta_n\}\right] \cdot \mathbb{E}\left[\alpha^{\bar{\pi}}(n) \middle| E, \theta_\ell = j, \tau = t_\ell\right] \\
&\qquad\qquad + \Pr\left[\tau = t_n + \delta_n \middle| E, \tau = \min\{t_\ell, t_n + \delta_n\}\right] \cdot \mathbb{E}\left[\alpha^{\bar{\pi}}(n) \middle| E, \tau = t_n + \delta_n\right] \\
&\quad = \sum_{j \in \mathcal{A}_i^*} \Pr\left[\theta_\ell = j, \tau = t_\ell \middle| E, \tau = \min\{t_\ell, t_n + \delta_n\}\right] \cdot \left(c(i,j) - \bar{c}_j\right) \\
&\qquad\qquad + \Pr\left[\tau = t_n + \delta_n \middle| E, \tau = \min\{t_\ell, t_n + \delta_n\}\right] \cdot c_a(i) \\
&\quad = \sum_{j \in \mathcal{A}_i^*} \frac{\lambda_j}{\mu + \sum_{k \in \mathcal{A}_i^*} \lambda_k} \cdot \left(c(i,j) - \bar{c}_j\right) + \frac{\mu}{\mu + \sum_{k \in \mathcal{A}_i^*} \lambda_k} \cdot c_a(i) \\
&\quad \leq \sum_{j \in \mathcal{A}_i^*} \frac{\lambda_j}{\mu + \sum_{k \in \mathcal{A}_i^*} \lambda_k} \cdot c(i,j) + \frac{\mu}{\mu + \sum_{k \in \mathcal{A}_i^*} \lambda_k} \cdot c_a(i) \\
&\quad = \bar{c}_i \ ,
\end{aligned}$$

To justify the second equality, observe that $\mathbb{E}[\alpha^{\bar{\pi}}(n) | E, \theta_\ell = j, \tau = t_\ell] = c(i,j) - \bar{c}_i$ by Claim 4 and the definition of the allocated cost, while $\mathbb{E}[\alpha^{\bar{\pi}}(n) | E, \tau = t_n + \delta_n] = c_a(i)$. The third equality proceeds from the distribution of the minimum of independent exponential random variables. $\square$

A.7. Proof of Claim 4

Let $q_1 \neq q_2 \in V_{t_\ell}^{\bar{\pi}}$ be two distinct vertices such that $q_1 < q_2 < \ell$. Suppose that $c(q_1, q_2) \leq \bar{c}_{q_1} + \bar{c}_{q_2}$. Consequently, we define the matching $M^+ = M_{t_{q_2}}^{\bar{\pi}} \cup \{\{q_1, q_2\}\}$, and observe that

$$\sum_{e \in M^+} c(e) - \sum_{n \in \phi(M^+)} \bar{c}_{\theta_n} - \left(\sum_{e \in M_{t_{q_2}}^{\bar{\pi}}} c(e) - \sum_{n \in \phi(M_{t_{q_2}}^{\bar{\pi}})} \bar{c}_{\theta_n} \right) = c(q_1, q_2) - \bar{c}_{q_1} - \bar{c}_{q_2} \leq 0 \ ,$$

where the first equality holds by noting that $q_1, q_2 \notin \phi(M_{t_{q_2}}^{\bar{\pi}})$ since $q_1, q_2 \in V_{t_\ell}^{\bar{\pi}}$ and $t_\ell > t_{q_2}$. The latter inequality contradicts that $M_{t_{q_2}}^{\bar{\pi}}$ is of maximal cardinality, and it follows that $c(q_1, q_2) - \bar{c}_{\theta_{q_1}} - \bar{c}_{\theta_{q_2}} > 0$. In particular, this implies that the matching $M_{t_\ell}^{\bar{\pi}}$ is formed by a single edge that covers vertex ℓ.

Now, observe that, for every $q \in V_{t_\ell}^{\bar{\pi}} \setminus \{n, \ell\}$, we have

$$c(q, \ell) - \bar{c}_{\theta_q} - \bar{c}_{\theta_\ell} \geq c(q, n) - c(n, \ell) - \bar{c}_{\theta_q} - \bar{c}_{\theta_\ell} > \bar{c}_i - c(n, \ell) - \bar{c}_\ell \geq -\bar{c}_\ell \ ,$$

where the second inequality holds since $c(q, n) - \bar{c}_{\theta_q} > \bar{c}_i$ as shown above, while the last inequality holds by the definition of ℓ, whereby $\theta_\ell \in \mathcal{A}_i^*$. On the other hand, we clearly have that $c(n, \ell) - \bar{c}_i - \bar{c}_\ell \leq -\bar{c}_\ell$. It immediately follows that $M_{t_m}^{\bar{\pi}} = \{n, \ell\}$ when $n \in V_{t_\ell}^{\bar{\pi}}$. $\square$

Appendix B: Proof of Theorem 2

B.1. Proof of Claim 3

The inequality is established by constructing a reduction of our reward-maximization instance $\mathcal{I}^r$ into an equivalent instance $\mathcal{I}^c$ of the cost-minimization problem, and by leveraging Lemma 4. Define $c_a = \max_{e \in E} r(e)$, and let $c(e) = 2 \cdot c_a - r(e)$ for every edge $e \in \mathcal{T}^2$. Consequently, the instance $\mathcal{I}^c$ is constructed using the same sojourn process as $\mathcal{I}^r$, while specifying the edge costs $c(\cdot)$ and the uniform abandonment penalty $c_a(i) = c_a$ for all $i \in \mathcal{T}$. Further, we let L^* be the optimal value of the corresponding linear programming benchmark (CB). It easy to verify that any optimal solution vector for (RB) is also an optimal solution vector for (CB), and consequently,

$$U^* + L^* = c_a \cdot \left(\sum_{i \in \mathcal{T}} \lambda_i \right) . \tag{19}$$

On the other hand, the cumulative cost $C_t^{\pi^*}$ at time t associated with the optimal matching policy π^* for instance $\mathcal{I}^c$ is given by:

$$\begin{aligned} C_t^{\pi^*} &= \sum_{\zeta \in \mathcal{E}(t)} c_a \cdot \left| D_\zeta^{\pi^*} \right| + \sum_{\zeta \in \mathcal{E}(t)} \sum_{e \in M_\zeta^{\pi^*}} c(e) \\ &= c_a \cdot \left(N(t) - \left| V_t^{\pi^*} \right| \right) - \sum_{\zeta \in \mathcal{E}(t)} \sum_{e \in M_\zeta^{\pi^*}} r(e) \\ &= c_a \cdot \left(N(t) - \left| V_t^{\pi^*} \right| \right) - R_t^{\pi^*} . \end{aligned}$$

It immediately follows that

$$r^{\pi^*} + c^{\pi^*} = \lim_{t \to +\infty} \frac{1}{t} \cdot \left(\mathbb{E}\left[C_t^{\pi^*} \right] + \mathbb{E}\left[R_t^{\pi^*} \right] \right) = c_a \cdot \lim_{t \to +\infty} \left(\frac{N(t)}{t} - \frac{|V_t^{\pi^*}|}{t} \right) = c_a \cdot \left(\sum_{i \in \mathcal{T}} \lambda_i \right) . \tag{20}$$

Hence, we obtain that

$$U^* = c_a \cdot \left(\sum_{i \in \mathcal{T}} \lambda_i \right) - L^* \geq c_a \cdot \left(\sum_{i \in \mathcal{T}} \lambda_i \right) - c^{\pi^*} = r^{\pi^*} ,$$

where the first equality holds by (19), the inequality proceeds from Lemma 4, and the last equality proceeds from (20). $\square$

B.2. Proof of Lemma 9

Arbitrary graphs. Let $(x^*_{i,j})_{(i,j) \in \mathcal{T}^2}$ and $(x^*_{i,a})_{i \in \mathcal{T}}$ designate an optimal flow for (RB). We begin by establishing the first claim. To this end, we define $\tilde{x}_{i,j} = \frac{1}{4} \cdot x^*_{i,j}$ for every $(i,j) \in \mathcal{T}^2$. In addition, for every $i \in \mathcal{T}$, we define $\tilde{x}_i$ and $\tilde{x}_{i,a}$ as follows:

$$\tilde{x}_i = \frac{\lambda_i}{2} - \left(\sum_{j \in \mathcal{T}} \tilde{x}_{j,i} \right) , \tag{21}$$

and

$$\tilde{x}_{i,a} = \lambda_i - \tilde{x}_i - \left(\sum_{j\in\mathcal{T}} \tilde{x}_{j,i}\right) - \left(\sum_{j\in\mathcal{T}} \tilde{x}_{i,j}\right) \ . \tag{22}$$

Constraint (12) implies that $\tilde{x}_i, \tilde{x}_{i,a} \geq 0$. Further, by the definition of $x_{i,a}$ in equation (22), the flow vector formed by $(\tilde{x}_{i,j})_{(i,j)\in\mathcal{T}^2}$ and $(\tilde{x}_{i,a}, \tilde{x}_i)_{i\in\mathcal{T}}$ satisfies the flow balance constraint (14). On the other hand, for every $(i,j)\in\mathcal{T}^2$, we have:

$$\tilde{x}_{i,a} = \frac{\lambda_i}{2} - \left(\sum_{j\in\mathcal{T}} \tilde{x}_{i,j}\right) \geq \frac{x^*_{i,a}}{2} \geq \frac{2\mu_i}{\lambda_j}\cdot\frac{1}{4}\cdot x^*_{i,j} = \frac{\mu_i}{\tilde{x}_j + \sum_{k\in\mathcal{T}} \tilde{x}_{k,j}}\cdot \tilde{x}_{i,j} \ ,$$

where the first equality is obtained by combining (22) and (21). The first inequality follows from constraint (12). The second inequality holds by constraint (13). The last equality proceeds from the definition of $\tilde{x}_{i,j}$ and equation (21). Hence, we have just shown that the flow vector formed by $(\tilde{x}_{i,j})_{(i,j)\in\mathcal{T}^2}$ and $(\tilde{x}_{i,a}, \tilde{x}_i)_{i\in\mathcal{T}}$ is feasible with respect to problem (QB). Thus, we have

$$Q^* \geq \sum_{(i,j)\in\mathcal{T}^2} r\left(i,j\right)\cdot \tilde{x}_{i,j} = \frac{1}{4}\cdot U^* \geq \frac{1}{4}\cdot r^{\pi^*} \ ,$$

where the last inequality proceeds from Claim 3.

Bipartite graphs. Now, suppose that the reward function $r(\cdot)$ describes a bipartite graph, and let $\mathcal{T}_1, \mathcal{T}_2$ be the partite sets. Without loss of generality, we assume that $\sum_{(i,j)\in\mathcal{T}_1\times\mathcal{T}_2} r(i,j)\cdot x^*_{i,j} \geq \frac{1}{2}\cdot U^*$, otherwise the partite sets can be swapped. Consequently, we define $\tilde{x}_{i,j} = x^*_{i,j}$ for every $(i,j)\in\mathcal{T}_1\times\mathcal{T}_2$, and $\tilde{x}_{i,j}=0$ for every $(i,j)\in\mathcal{T}^2\setminus(\mathcal{T}_1\times\mathcal{T}_2)$. Further, we let $\tilde{x}_{i,a} = x^*_{i,a}$ if $i\in\mathcal{T}_1$ and $\tilde{x}_{i,a}=0$ otherwise. Lastly, for every $i\in\mathcal{T}$, we define $\tilde{x}_i$ as follows:

$$\tilde{x}_i = \lambda_i - \tilde{x}_{i,a} - \left(\sum_{j\in\mathcal{T}} \tilde{x}_{j,i}\right) - \left(\sum_{j\in\mathcal{T}} \tilde{x}_{i,j}\right) \ . \tag{23}$$

Clearly, $\tilde{x}_i \geq 0$ for every $i\in\mathcal{T}$ by constraint (12). Further, the flow vector formed by $(\tilde{x}_{i,j})_{(i,j)\in\mathcal{T}^2}$ and $(\tilde{x}_{i,a}, \tilde{x}_i)_{i\in\mathcal{T}}$ verifies the flow balance constraint (14) by equation (23). On the other hand, constraint (15) is satisfied for every $(i,j)\in\mathcal{T}^2\setminus(\mathcal{T}_1\times\mathcal{T}_2)$ since $\tilde{x}_{i,j}=0$. Now, for every $(i,j)\in\mathcal{T}_1\times\mathcal{T}_2$, we have:

$$\tilde{x}_{i,a} = x^*_{i,a} \geq \frac{\mu_i}{\lambda_j}\cdot x^*_{i,j} = \frac{\mu_i}{\tilde{x}_j + \sum_{k\in\mathcal{T}} \tilde{x}_{k,j}}\cdot \tilde{x}_{i,j} \ ,$$

where the first inequality proceeds from constraint (13) and the last equality holds since $\tilde{x}_{i,j} = x^*_{i,j}$ by construction, noting that equation (23) implies that $\lambda_j = \tilde{x}_j + \sum_{k\in\mathcal{T}} \tilde{x}_{k,j}$ when $j\in\mathcal{T}_2$. We have just shown that the flow vector formed by $(\tilde{x}_{i,j})_{(i,j)\in\mathcal{T}^2}$ and $(\tilde{x}_{i,a}, \tilde{x}_i)_{i\in\mathcal{T}}$ is a feasible solution of (QB). It follows that $Q^* \geq \sum_{(i,j)\in\mathcal{T}_1\times\mathcal{T}_2} r(i,j)\cdot x^*_{i,j} \geq \frac{1}{2}\cdot U^* \geq \frac{1}{2}\cdot r^{\pi^*}$, where the first inequality is due to our initial hypothesis, and the last inequality proceeds from Claim 3.

Bipartite-asymmetric setting. Here, our analysis is nearly identical to that of the bipartite case, except that the premise $\sum_{(i,j)\in\mathcal{T}_1\times\mathcal{T}_2} x^*_{i,j} \geq \frac{1}{2}\cdot U^*$ is replaced by $\sum_{(i,j)\in\mathcal{T}_1\times\mathcal{T}_2} x^*_{i,j} = U^*$. To establish this equality, let $(\mathcal{T}_1,\mathcal{T}_2,\bar{E})$ be the bipartite graph associated with the reward function $r(\cdot)$. Without loss of generality, we assume that $\mu_i = +\infty$ for all $i\in\mathcal{T}_2$, otherwise the partite sets can be swapped. Hence, for all $i\in\mathcal{T}_2$, constraint (13) implies that $x_{i,j}=0$ in light of the assumption $\mu_i=+\infty$. (Since the vertices of type $i\in\mathcal{T}_2$ leave immediately after their arrival, they are necessarily passive if they are matched, which naturally implies that $x_{i,j}=0$ for all $j\in\mathcal{T}$.) It follows that

$$U^* = \sum_{(i,j)\in\mathcal{T}_1\times\mathcal{T}_2} r(i,j)\cdot x^*_{i,j} + \sum_{(i,j)\in\mathcal{T}_2\times\mathcal{T}_1} r(i,j)\cdot x^*_{i,j} = \sum_{(i,j)\in\mathcal{T}_1\times\mathcal{T}_2} r(i,j)\cdot x^*_{i,j} \ .$$

□

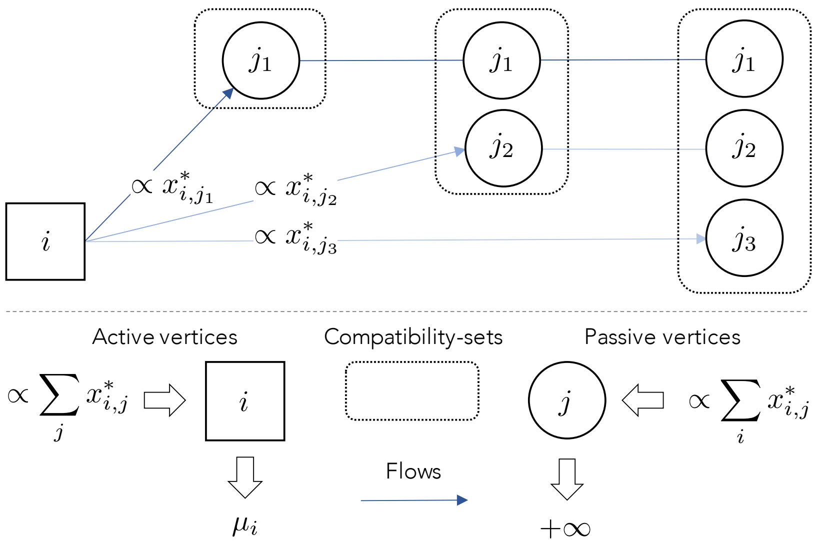
B.3. Randomized compatibility matching policy

Here, we describe our matching algorithm in the reward-maximization setting. The resulting matching policy is referred to as the *randomized compatibility* policy. At a high-level, this approach approximately simulates the matching flows induced by the fluid relaxation benchmark. Hereinafter, we denote by $\tilde{x}$ a feasible flow for problem (QB) satisfying the performance bounds stated by Lemma 9. Our algorithm assigns each arriving vertex with a *random label*; the distribution over labels is constructed using a careful decomposition of the flow $\tilde{x}$. Specifically, each label indicates whether the vertex in question is *active* or *passive*. If the vertex is active, the label further specifies a restricted subset of types, named the *compatibility-set*, that can be matched with this vertex. Consequently, our matching policy is greedy with respect to the restrictions imposed by the labels. The remainder of this section formally describes the notions of vertex label, compatibility-set and greedy matching.

Flow decomposition and compatibility-sets. In what follows, we fix a type $i\in\mathcal{T}$, and we define S_i as the support of the flow vector $(\tilde{x}_{i,j})_{j\in\mathcal{T}}$, i.e., $S_i=\{j\in\mathcal{T}:\tilde{x}_{i,j}>0\}$. Our objective is to linearly decompose the flow outgoing from type i, formed by $(\tilde{x}_{i,j})_{j\in\mathcal{T}}$ and $\tilde{x}_{i,a}$, into a collection of flow components that satisfy two key properties: (i) the supports of the flow components are nested subsets of S_i, (ii) each flow component saturates constraint (15). The compatibility-sets are defined as the collection of subsets that support each of the flow component appearing in this decomposition. The construction of the compatibility-sets is illustrated in Figure 4. We mention in passing that our approach has similarities with randomized assortment policies informed by fluid relaxations in MNL-based revenue management problems (Topaloglu 2013, Gallego et al. 2014). However, due to the two-sided nature of the market dynamics, our construction is more intricate.

To this end, we let $\sigma_i(1),\sigma_i(2),\ldots,\sigma_i(|S_i|)$ be the sequence of types $j\in S_i$ by decreasing order of $\frac{\tilde{x}_{i,j}}{\lambda^p_j}$, where $\lambda^p_j=\tilde{x}_j+\sum_{k\in\mathcal{T}}\tilde{x}_{k,j}$, breaking ties arbitrarily. Consequently, for every $\ell\in[1,|S_i|]$, the

Figure 4 Illustration of our construction of the compatibility-sets.



ℓ-th compatibility-set associated with type i is defined as the set $S_i^\ell = \{\sigma_i(q) : q \in [1,\ell]\}$. Clearly, the compatibility-sets are nested, i.e., $S_i^1 \subseteq S_i^2 \subseteq \ldots \subseteq S_i^\ell$. Moreover, each compatibility-set S_i^ℓ is associated with a flow quantity $\hat{\lambda}_{i,\ell}$ constructed using the following inductive procedure:

- *Base case, $\ell = |S_i|$:* Initially, we have:

$$\hat{\lambda}_{i,\ell} = \left(\frac{\mu_i + \sum_{j \in S_i} \lambda_j^p}{\lambda_{\sigma_i(\ell)}^p} \right) \cdot \tilde{x}_{i,\sigma_i(\ell)} \ . \tag{24}$$

- *Induction step, $\ell \leq |S_i| - 1$:* Here, we have:

$$\hat{\lambda}_{i,\ell} = \left(\frac{\mu_i + \sum_{j \in S_i^\ell} \lambda_j^p}{\lambda_{\sigma_i(\ell)}^p} \right) \cdot \left(\tilde{x}_{i,\sigma_i(\ell)} - \sum_{q=\ell+1}^{|S_i|} \left(\frac{\lambda_{\sigma_i(\ell)}^p}{\mu_i + \sum_{j \in S_i^q} \lambda_j^p} \right) \cdot \hat{\lambda}_{i,q} \right) \ . \tag{25}$$

The next claim, we highlight certain key properties of the compatibility-sets and their associated flow quantities.

Claim 5. *For every $\ell \in [1, |S_i|]$, we have $\hat{\lambda}_{i,\ell} \geq 0$ and $\tilde{x}_{i,\sigma_i(\ell)} = \sum_{q=\ell}^{|S_i|} \frac{\lambda_{\sigma_i(\ell)}^p}{\mu_i + \sum_{j \in S_i^q} \lambda_j^p} \cdot \hat{\lambda}_{i,q}$. Furthermore, we have $\tilde{x}_{i,a} = \sum_{\ell=1}^{|S_i|} \frac{\mu_i}{\mu_i + \sum_{j \in S_i^\ell} \lambda_j^p} \cdot \hat{\lambda}_{i,\ell}$.*

The proof is deferred to the end of this section (Appendix B.5). Based on Claim 5, our construction of the compatibility-sets and their associated flow quantities yields a decomposition of the flow outgoing from type i, described by $(\tilde{x}_{i,j})_{j \in \mathcal{T}}$ and $\tilde{x}_{i,a}$. To this end, for every $\ell \in [1, |S_i|]$, we define the vector $(\hat{x}_{i,j}^\ell)_{j \in \mathcal{T}}$ supported by S_i^ℓ, where, for every $j \in S_i^\ell$,

$$\hat{x}_{i,j}^\ell = \frac{\lambda_j^p}{\mu_i + \sum_{k \in S_i^\ell} \lambda_k^p} \cdot \hat{\lambda}_{i,\ell} \ .$$

Similarly, we define the quantity $\hat{x}^{\ell}_{i,a}$ as follows:

$$\hat{x}^{\ell}_{i,a} = \frac{\mu_i}{\mu_i + \sum_{j \in S_i^{\ell}} \lambda_j^p} \cdot \hat{\lambda}_{i,\ell} \ .$$

By definition, each flow vector $(\hat{x}^{\ell}_{i,j})_{j \in \mathcal{T}}, \hat{x}^{\ell}_{i,a}$ saturates constraint (15), meaning that $\frac{\hat{x}^{\ell}_{i,j}}{\hat{x}^{\ell}_{i,a}} = \frac{\lambda_j^p}{\mu_i}$. Moreover, Claim 5 directly implies that $\tilde{x}_{i,j} = \sum_{\ell=1}^{|S_i|} \hat{x}^{\ell}_{i,j}$ for every $j \in \mathcal{T}$, and $\tilde{x}_{i,a} = \sum_{\ell=1}^{|S_i|} \hat{x}^{\ell}_{i,a}$.

Having constructed our decomposition of the flow $\tilde{x}$ and specified the compatibility-sets $(S_i^{\ell})_{i \in [n], \ell \in [|S_i|]}$, we proceed by describing the randomized compatibility matching policy, which we denote by $\hat{\pi}$. All the steps of the algorithm are summarized by the pseudo-code of Algorithm 2.

Step 1: Assigning vertices with random labels. Each arriving vertex $n \in \mathbb{N}^*$ is assigned with a random label (θ_n, ℓ_n) upon its arrival. Here, θ_n is the type of vertex n, while ℓ_n is a random integer generated by our policy, independently from the sojourn process. While the precise role of the label ℓ_n will be revealed later on, the event $\{\ell_n = 0\}$ indicates that n is passive, while $\{\ell_n > 0\}$ indicates that n is active. Furthermore, conditional on $\{\ell_n > 0\}$, ℓ_n is a random experiment over the compatibility-set indices $[1, |S_{\theta_n}|]$. To complete the probabilistic description of our labelling operation, we specify the distribution of ℓ_n conditional on $\{\theta_n = i\}$, for each type $i \in \mathcal{T}$.

- *Active vs. passive:* The probability of assigning the value 0 to ℓ_n is given by:

$$\Pr[\ell_n = 0 | \, \theta_n = i] = \frac{\lambda_i^p}{\lambda_i} \ . \tag{26}$$

- *Compatibility-set:* Conditional on $\{\ell_n > 0\}$, ℓ_n follows a multinomial distribution over the compatibility-set indices $[1, |S_i|]$; specifically, for every $\ell \in [1, |S_i|]$, we have:

$$\Pr[\ell_n = \ell | \, \ell_n > 0, \theta_n = i] = \frac{\hat{\lambda}_{i,\ell}}{\sum_{\ell=1}^{|S_i|} \hat{\lambda}_{i,\ell}} \ . \tag{27}$$

Step 2: Greedy matching. Having assigned each incoming vertex with a random label, our policy greedily selects the matching that maximizes the immediate reward subject to the restrictions imposed by the labels of the vertices. Specifically, the decision epochs are the arrival times of vertices. Upon the arrival of each vertex $n \in \mathbb{N}^*$ at time t_n, we distinguish two cases:

- *Case (1): n is active.* Conditional on $\ell_n > 0$, we have $M^{\hat{\pi}}_{t_n} = \emptyset$.
- *Case (2): n is passive.* Conditional on $\ell_n = 0$, we define the set of vertices $\mathcal{C} = \{m \in V^{\hat{\pi}}_{t_n} : \ell_m > 0, \theta_n \in S^{\ell_m}_{\theta_m}\}$ formed by all active vertices that have a compatibility-set containing θ_n. In the sequel, the labels of the vertices in $\mathcal{C}$ are said to be *compatible* with type θ_n.
  - — *Case (2a): $\mathcal{C} = \emptyset$.* Here, we have $M^{\hat{\pi}}_{t_n} = \emptyset$. Moreover, n is never matched subsequently. Without loss of generality, n abandons the system at time t_n.
  - — *Case (2b): $\mathcal{C} \neq \emptyset$.* In this case, the matching $M^{\hat{\pi}}_{t_n}$ is a singleton formed by an edge $\{n, m\} \in E(V^{\hat{\pi}}_t)$ that maximizes $r(\theta_n, \theta_m)$ out of all $m \in \mathcal{C}$.

Algorithm 2 Computing the randomized compatibility policy

```
**Instance parameters:** Collection of types $\mathcal{T}$, arrival rates $(\lambda_i)_{i\in\mathcal{T}}$, abandonment rates $(\mu_i)_{i\in\mathcal{T}}$, matching rewards $r(\cdot)$.

**Preliminary step: Flow decomposition and compatibility-sets**
Construct a feasible flow $\tilde{x}$ for problem $(QB)$ such that $\sum_{(i,j)\in\mathcal{T}^2}\tilde{x}_{i,j} \geq \kappa \cdot r^{\pi^*}$ (Lemma 9)
For every $i \in \mathcal{T}$:
    Define $S_i = \{j \in \mathcal{T} : \tilde{x}_{i,j} > 0\}$ and $\lambda^p_j = \tilde{x}_j + \sum_{k\in\mathcal{T}} \tilde{x}_{k,j}$
    Compute the permutation $\sigma_i(\cdot)$ by sorting the types $j \in S_i$ in decreasing order of $\frac{\tilde{x}_{i,j}}{\lambda^p_j}$
    For every $\ell \in [|S_i|]$:
        Compute $\hat{\lambda}_{i,\ell}$ according to equations (24) and (25)

**Step 1 (online computation): Assigning arriving vertices with random labels**
*Input:* An arriving vertex $n \in \mathbb{N}$ and his type $\theta_n$
Sample $\ell_n$ independently according to the conditional probability distribution (26)-(27)
*Output:* Return $\ell_n$

**Step 2 (online computation): Greedy matching**
*Input:* Realization graph $G_t = (V^{\hat{\pi}}_t, E(V^{\hat{\pi}}_t))$ at time $t \geq 0$
Set $M^{\hat{\pi}}_t = \emptyset$
If $t$ is an epoch in which a passive vertex $n \in \mathbb{N}^*$ arrives, i.e., $t = t_n$ and $\ell_n = 0$:
    Define $\mathcal{C} = \{m \in V^{\hat{\pi}}_{t_n} : \ell_m > 0, \theta_n \in S^{\ell_m}_{\theta_m}\}$
    If $\mathcal{C} \neq \emptyset$:
        Set $M^{\hat{\pi}}_t = \{n, m\}$ where $\{n, m\} \in E(V^{\hat{\pi}}_t)$ maximizes $r(\theta_n, \theta_m)$ out of all $m \in \mathcal{C}$
*Output:* Return $M^{\hat{\pi}}_t$
```

B.4. Analysis

In this section, we derive a constant-factor approximation guarantee for the randomized compatibility policy $\hat{\pi}$ described in Section B.3, thereby completing the proof of Theorem 2.

Lemma 10. *For all instances of the dynamic stochastic matching problem, we have $r^{\hat{\pi}} \geq (\frac{e-1}{4e}) \cdot r^{\pi^*}$. On bipartite graphs, we have $r^{\hat{\pi}} \geq (\frac{e-1}{2e}) \cdot r^{\pi^*}$, while $r^{\hat{\pi}} \geq (\frac{e-1}{e}) \cdot r^{\pi^*}$ in the bipartite-impatient setting.*

Since our matching policy $\hat{\pi}$ is stationary, the state of the system $\{\vartheta(V^{\hat{\pi}}_t)\}_{t\geq 0}$ describes a positive recurrent Markov chain, which converges to a unique stationary distribution. However, this distribution is hard to compute analytically. Hence, the proof of Lemma 10 proceeds in two steps. First, we construct a modified stochastic process, termed the *virtual Markov chain*, for which it is easy to compute the resulting limiting distribution. Second, we relate the virtual Markov chain to our original stochastic process, by exploiting a property of stochastic dominance. Consequently, we derive a lower bound on $r^{\hat{\pi}}$ based on the stationary distribution of the virtual Markov chain, which ensues the desired performance guarantee.

Augmented state space. Recall from Section 2 that the state of system at time t is described by a vector $\vartheta(V_t^{\hat{\pi}}) \in \mathcal{V}$, where $\vartheta_i(V_t^{\hat{\pi}})$ is the number of type-i vertices in $V_t^{\hat{\pi}}$. In what follows, we augment our state space to account for the labelling actions performed by our matching policy. To this end, we let $\mathcal{L}$ denote the collection of all possible labels. For every $(i,\ell) \in \mathcal{L}$, and $t \geq 0$, we define $\bar{X}_t^{i,\ell}$ as the number of vertices in $\bar{V}_t^{\hat{\pi}}$ assigned with the label (i,ℓ). In vector form, each state is denoted by $\bar{X}_t = (\bar{X}_t^{i,\ell})_{(i,\ell)\in\mathcal{L}}$. We let $\mathcal{X}$ denote the augmented state space, formed by all the states reached by $\{\bar{X}_t\}_{t\geq 0}$. By the definition of the randomized compatibility policy, all passive vertices leave the system immediately after their arrival (see cases (2a) and (2b) of the matching decisions in Appendix B.3). Hence, the support of all vectors $x \in \mathcal{X}$ is contained in the set of labels $\mathcal{L}^{>0} = \{(i,\ell) \in \mathcal{L} : \ell > 0\}$.

Given a state $x \in \mathcal{X}$, we describe all the possible transitions from state x. To this end, let $\mathcal{L}_x^{>0} = \{(i,\ell) \in \mathcal{L}^{>0} : x_{i,\ell} > 0\}$ be the subset of active labels having at least one vertex in state x. Upon a new arrival to the system, the state transitions are deterministic conditional on the current state and the label of the arriving vertex. In particular, for every $(i,\ell) \in \mathcal{L}_x^{>0}$, we define $\mathcal{T}_x^{i,\ell}$ as the subset of types $j \in \mathcal{T}$ such that the arrival of a passive type-j vertex triggers a transition from state x to state $x - e_{i,\ell}$; this state transition corresponds to case (2b) of our matching decisions, where an arriving vertex, labelled $(j,0)$, is matched with an active vertex having a compatible label (i,ℓ). With these definitions at hand, the transitions of the continuous-time Markov chain $\{\bar{X}_t\}_{t\geq 0}$ are governed by the intensity matrix $(\bar{\mathcal{Q}}_{x,y})_{(x,y)\in\mathcal{X}^2}$, where, for every distinct states $x \neq y$, we have

$$\bar{\mathcal{Q}}_{x,y} = \begin{cases} \hat{\lambda}_{i,\ell} & \text{if } y = x + e_{i,\ell}, \text{ where } (i,\ell) \in \mathcal{L}^{>0} \ , & (28) \\ |x^{i,\ell}| \cdot \mu_i + \displaystyle\sum_{j\in\mathcal{T}_x^{i,\ell}} \hat{\lambda}_j^p & \text{if } y = x - e_{i,\ell}, \text{ where } (i,\ell) \in \mathcal{L}_x^{>0}, & (29) \\ 0 & \text{otherwise} \ , \end{cases}$$

and $\bar{\mathcal{Q}}_{x,x} = -\sum_{\substack{y\in\mathcal{X}\\ y\neq x}} \bar{\mathcal{Q}}_{x,y}$ for every $x \in \mathcal{X}$. Equation (28) corresponds to case (1) of the randomized compatibility policy, where an active vertex labelled (i,ℓ) arrives, and thus, the state variable $x^{i,\ell}$ is incremented by one unit. Equation (29) subsumes the state transitions whereby a vertex labelled (i,ℓ) either abandons the system unmatched, or gets matched with an arriving vertex, labelled $(j,0)$ for some $j \in \mathcal{T}_x^{i,\ell}$. In both cases, the state variable $x^{i,\ell}$ is decremented by one unit.

Virtual Markov chain. The main challenge in analyzing the Markov chain $\{\bar{X}_t\}_{t\geq 0}$ is the probabilistic dependence between the state variables $\bar{X}_t^{i,\ell}$ over $(i,\ell)\in\mathcal{L}$. For example, we are not aware of any tractable characterization of the stationary distribution of $\{\bar{X}_t\}_{t\geq 0}$. In what follows, our goal is to introduce a simplified stochastic process $\{\tilde{X}_t\}_{t\geq 0}$, termed the *virtual Markov chain*, as a proxy for $\{\bar{X}_t\}_{t\geq 0}$. Intuitively, the probabilistic dependence between state variables stems from step (2b) of the randomized compatibility policy $\hat{\pi}$, whereby a single vertex is matched out of all compatible labels. The virtual Markov chain will "eliminate" these dependencies by allowing *all* compatible labels to be depleted at a uniform rate. Specifically, we construct the intensity matrix $(\tilde{\mathcal{Q}}_{x,y})_{(x,y)\in\mathcal{X}^2}$ that governs $\{\tilde{X}_t\}_{t\geq 0}$ by specifying, for every distinct states $x\neq y$,

$$\tilde{\mathcal{Q}}_{x,y}=\begin{cases}\hat{\lambda}_{i,\ell} & \text{if } y=x+e_{i,\ell},\text{ where } (i,\ell)\in\mathcal{L}^{>0}, \qquad (30)\\ |x^{i,\ell}|\cdot\mu_i+\sum_{j\in S_i^{\ell}}\hat{\lambda}_j^p & \text{if } y=x-e_{i,\ell},\text{ where } (i,\ell)\in\mathcal{L}_x^{>0}, \qquad (31)\\ 0 & \text{otherwise ,}\end{cases}$$

and $\tilde{\mathcal{Q}}_{x,x}=-\sum_{y\in\mathcal{X}}\tilde{\mathcal{Q}}_{x,y}$ for every $x\in\mathcal{X}$. We highlight that the transition rates of the virtual Markov chain do not capture feasible matching decisions; in other words, there exists no matching policy that induces the stochastic process $\{\tilde{X}_t\}_{t\geq 0}$. Indeed, equation (31) would imply that the "combined match rate" of passive type-j vertices at state x is $|\{(i,\ell)\in\mathcal{L}_x^{>0}: j\in S_i^{\ell}\}|\cdot\hat{\lambda}_j^p$. This should be contrasted with equation (29), where the match rate is at most $\hat{\lambda}_j^p$ (namely, a single active vertex is picked out of all compatible labels).

LEMMA 11. *For every $\mathcal{I}\subseteq\mathcal{L}^{>0}$, we have:*

$$\lim_{t\to+\infty}\Pr\left[\sum_{(i,\ell)\in\mathcal{I}}\tilde{X}_t^{i,\ell}=0\right]\leq\prod_{(i,\ell)\in\mathcal{I}}e^{-\frac{\hat{\lambda}_{i,\ell}}{\mu_i+\sum_{j\in S_i^{\ell}}\lambda_j^p}} .$$

Proof. The key observation is that the stochastic processes $\{\tilde{X}_t^{i,\ell}\}_{t\geq 0}$ indexed by $(i,\ell)\in\mathcal{I}$ are mutually independent birth-death processes. Indeed, for every $(i,\ell)\in L$, the transitions of $\{\tilde{X}_t^{i,\ell}\}_{t\geq 0}$ are governed by the intensity matrix $(\tilde{\mathcal{Q}}_{u,v}^{i,\ell})_{(u,v)\in\mathbb{N}^2}$, defined as follows:

$$\tilde{\mathcal{Q}}_{u,v}^{i,\ell}=\begin{cases}\hat{\lambda}_{i,\ell} & \text{if } v=u+1\ , \qquad (32)\\ u\cdot\mu_i+\sum_{j\in S_i^{\ell}}\hat{\lambda}_j^p & \text{if } v=u-1, \qquad (33)\\ 0 & \text{otherwise ,}\end{cases}$$

and $\tilde{\mathcal{Q}}_{u,u}^{i,\ell}=-\sum_{\substack{v\in\mathbb{N}\\ v\neq u}}\tilde{\mathcal{Q}}_{u,v}^{i,\ell}$ for every $u\in\mathbb{N}$. Equations (32) and (33) immediately proceed from (30) and (31), respectively. Hence, we obtain

$$\lim_{t\to+\infty}\Pr\left[\sum_{(i,\ell)\in\mathcal{I}}\tilde{X}_t^{i,\ell}=0\right]=\prod_{(i,\ell)\in\mathcal{I}}\left(\lim_{t\to+\infty}\Pr\left[\tilde{X}_t^{i,\ell}=0\right]\right)\leq\prod_{(i,\ell)\in\mathcal{I}}e^{-\frac{\hat{\lambda}_{i,\ell}}{\mu_i+\sum_{j\in S_i^{\ell}}\lambda_j^p}} ,$$

where the equality proceeds from the mutual independence of the birth-death processes $\{\tilde{X}_t^{i,\ell}\}_{t\geq 0}$ over $(i,\ell)\in\mathcal{I}$, while the inequality immediately follows from equations (32) and (33). $\square$

Bounding the expected average reward. To conclude our analysis, we relate the original Markov chain $\{\bar{X}_t\}_{t\geq 0}$ to the virtual Markov chain $\{\tilde{X}_t\}_{t\geq 0}$, through a stochastic dominance relationship. Namely, given two random variables $Y^{(1)}, Y^{(2)}$, taking values in a countable partially-ordered set $\mathcal{Y}$, the stochastic order $Y^{(1)} \preceq Y^{(2)}$ indicates that $\Pr[Y^{(1)} \leq y] \geq \Pr[Y^{(2)} \leq y]$ for every $y\in\mathcal{Y}$.

LEMMA 12. *For every $t\geq 0$, $\tilde{X}_t \preceq \bar{X}_t$.*

The proof, presented at the end of this section (Appendix B.6), proceeds from straightforward comparisons of the intensity matrices $(\bar{\mathcal{Q}}_{x,y})_{(x,y)\in\mathcal{X}^2}$ and $(\tilde{\mathcal{Q}}_{x,y})_{(x,y)\in\mathcal{X}^2}$. Now, for every $j\in\mathcal{T}$, let $\mathcal{L}_j$ be the set of all labels (i,ℓ) compatible with type j, i.e., $j\in S_i^\ell$. We construct the sequence $(i_j^1,\ell_j^1),(i_j^2,\ell_j^2),\ldots$, by ranking the labels $(i,\ell)\in\mathcal{L}_j$ by decreasing order of $r(i,j)$, and breaking ties arbitrarily. By exploiting these ranked sequences, the expected average reward of the randomized compatibility policy $\hat{\pi}$ can be expressed as follows:

$$\begin{aligned}
r^{\hat{\pi}} &= \sum_{j\in\mathcal{T}} \hat{\lambda}_j^p \cdot \sum_{q=1}^{|\mathcal{L}_j|} r(i_j^q,j)\cdot\left(\lim_{t\to+\infty}\Pr\left[\sum_{r=1}^{q-1}\bar{X}_t^{i_j^r,\ell_j^r}=0\ ,\ \bar{X}_t^{i_j^q,\ell_j^q}>0\right]\right)\\
&= \sum_{j\in\mathcal{T}} \hat{\lambda}_j^p \cdot \sum_{q=1}^{|\mathcal{L}_j|} \left(r(i_j^q,j)-r(i_j^{q+1},j)\right)\cdot\left(\lim_{t\to+\infty}\Pr\left[\sum_{r=1}^{q}\bar{X}_t^{i_j^r,\ell_j^r}>0\right]\right)\\
&\geq \sum_{j\in\mathcal{T}} \hat{\lambda}_j^p \cdot \sum_{q=1}^{|\mathcal{L}_j|} \left(r(i_j^q,j)-r(i_j^{q+1},j)\right)\cdot\left(\lim_{t\to+\infty}\Pr\left[\sum_{r=1}^{q}\tilde{X}_t^{i_j^r,\ell_j^r}>0\right]\right)\\
&\geq \sum_{j\in\mathcal{T}} \hat{\lambda}_j^p \cdot \sum_{q=1}^{|\mathcal{L}_j|} \left(r(i_j^q,j)-r(i_j^{q+1},j)\right)\cdot\left(1-\prod_{r=1}^{q} e^{-\frac{\hat{\lambda}_{i^r,\ell^r}}{\mu_{i^r}+\sum_{k\in S_{i^r}^{\ell^r}}\hat{\lambda}_k^p}}\right)\\
&\geq \sum_{j\in\mathcal{T}} \hat{\lambda}_j^p \cdot \sum_{q=1}^{|\mathcal{L}_j|} \left(r(i_j^q,j)-r(i_j^{q+1},j)\right)\cdot\left(1-\frac{1}{e}\right)\cdot\left(\sum_{r=1}^{q}\frac{\hat{\lambda}_{i^r,\ell^r}}{\mu_{i^r}+\sum_{k\in S_{i^r}^{\ell^r}}\hat{\lambda}_k^p}\right)\\
&= \left(1-\frac{1}{e}\right)\cdot\sum_{j\in\mathcal{T}}\sum_{q=1}^{|\mathcal{L}_j|} r(i_j^q,j)\cdot\frac{\hat{\lambda}_j^p}{\mu_{i^q}+\sum_{k\in S_{i^q}^{\ell^q}}\hat{\lambda}_k^p}\cdot\hat{\lambda}_{i^q,\ell^q}\\
&= \left(1-\frac{1}{e}\right)\cdot\sum_{(i,j)\in\mathcal{T}^2} r(i,j)\cdot\tilde{x}_{i,j}\\
&\geq \kappa\left(1-\frac{1}{e}\right)\cdot r^{\pi^*}
\end{aligned}$$

where, for convenience of notation, we have $r(i_j^{q+1},j)=0$ for every $j\in\mathcal{T}$ and $q=|\mathcal{L}_j|$. The first equation proceeds from the PASTA property (Wolff 1982), with respect the Markov chain $\{\bar{X}_t\}_{t\geq 0}$ and the arrivals of vertices labelled $(j,0)$. Indeed, by step (2b) of the matching decisions, an arriving passive vertex labelled $(j,0)$ is matched with an available vertex of compatible type $(i,\ell)\in\mathcal{L}_j$ that maximizes $r(i,j)$ (see Appendix B.3). The first inequality holds since $\tilde{X}_t \preceq \bar{X}_t$ for every $t \geq 0$, by Lemma 12. The second inequality proceeds from Lemma 10. The third inequality holds since, for every $j \in \mathcal{T}$, we have

$$\sum_{r=1}^{|\mathcal{L}_j|} \frac{\hat{\lambda}_{i^r,\ell^r}}{\mu_{i^r} + \sum_{k \in S_{i^r}^{\ell^r}} \hat{\lambda}_k^p} = \frac{1}{\hat{\lambda}_j^p} \cdot \left(\sum_{r=1}^{|\mathcal{L}_j|} \frac{\hat{\lambda}_{i^r,\ell^r} \cdot \hat{\lambda}_j^p}{\mu_{i^r} + \sum_{k \in S_{i^r}^{\ell^r}} \hat{\lambda}_k^p} \right) = \frac{1}{\tilde{x}_j + \sum_{k \in \mathcal{T}} \tilde{x}_{k,j}} \cdot \left(\sum_{k \in \mathcal{T}} \tilde{x}_{k,j} \right) \leq 1 \ .$$

Finally, the fourth equality follows from the flow decomposition established by Claim 5, and the last inequality proceeds from Lemma 9. $\square$

B.5. Proof of Claim 5

Without loss of generality, we may assume that constraint (15) is saturated for the pair of types $(i, \sigma_i(1))$, meaning that $x_{i,a}^* = \frac{\mu_i}{\lambda_{\sigma_i(1)}^p} \cdot x_{i,\sigma_i(1)}^*$; otherwise, we could decrease the variable $x_{i,a}^*$ and increase the variable x_i^*, while preserving the optimality and feasibility of the flow. Suppose temporarily that the first part of Claim 5 is established. It follows that

$$x_{i,a}^* = \frac{\mu_i}{\lambda_{\sigma_i(1)}^p} \cdot x_{i,\sigma_i(1)}^* = \frac{\mu_i}{\lambda_{\sigma_1(i)}^p} \cdot \left(\sum_{\ell=1}^{|S_i|} \frac{\lambda_{\sigma_i(1)}^p}{\mu_i + \sum_{j \in S_i^\ell} \lambda_j^p} \cdot \hat{\lambda}_{i,\ell} \right) = \sum_{\ell=1}^{|S_i|} \frac{\mu_i}{\mu_i + \sum_{j \in S_i^\ell} \lambda_j^p} \cdot \hat{\lambda}_{i,\ell} \ .$$

Consequently, it suffices to establish the first part of Claim 5. The proof is based on an induction over $\ell \in [1, |S_i|]$. For $\ell = |S_i|$, equation (24) immediately implies that $\hat{\lambda}_{i,|S_i|} \geq 0$ and $x_{i,\sigma_i(\ell)}^* = \frac{\lambda_{\sigma_i(\ell)}^p}{\mu_i + \sum_{j \in S_i^\ell} \lambda_j^p} \cdot \hat{\lambda}_{i,\ell}$. Now, we examine the case where $\ell \leq |S_i| - 1$. By rearranging (25), we derive the equation

$$x_{i,\sigma_i(\ell)}^* = \sum_{q=\ell}^{|S_i|} \frac{\lambda_{\sigma_i(\ell)}^p}{\mu_i + \sum_{j \in S_i^q} \lambda_j^p} \cdot \hat{\lambda}_{i,q} \ .$$

Hence, it remains to show that $\hat{\lambda}_{i,\ell} \geq 0$. To this end, observe that

$$\begin{aligned}
\hat{\lambda}_{i,\ell} &= \left(\frac{\mu_i + \sum_{j \in S_i^\ell} \lambda_i^p}{\lambda_{\sigma_i(\ell)}^p} \right) \cdot \left(x_{i,\sigma_i(\ell)}^* - \sum_{q=\ell+1}^{|S_i|} \left(\frac{\lambda_{\sigma_i(\ell)}^p}{\mu_i + \sum_{j \in S_i^q} \lambda_j^p} \right) \cdot \hat{\lambda}_{i,q} \right) \\
&\geq \left(\frac{\mu_i + \sum_{j \in S_i^\ell} \lambda_i^p}{\lambda_{\sigma_i(\ell)}^p} \right) \cdot \left(x_{i,\sigma_i(\ell)}^* - \frac{x_{i,\sigma_i(\ell)}^*}{x_{i,\sigma_i(\ell+1)}^*} \cdot \sum_{q=\ell+1}^{|S_i|} \left(\frac{\lambda_{\sigma_i(\ell+1)}^p}{\mu_i + \sum_{j \in S_i^q} \lambda_j^p} \right) \cdot \hat{\lambda}_{i,q} \right) \\
&= \left(\frac{\mu_i + \sum_{j \in S_i^\ell} \lambda_i^p}{\lambda_{\sigma_i(\ell)}^p} \right) \cdot \left(x_{i,\sigma_i(\ell)}^* - x_{i,\sigma_i(\ell)}^* \right) \\
&\geq 0 \ ,
\end{aligned}$$

where the inequality holds since $\frac{x_{i,\sigma_i(\ell)}^*}{\lambda_{\sigma_i(\ell)}^p} \geq \frac{x_{i,\sigma_i(\ell+1)}^*}{\lambda_{\sigma_i(\ell+1)}^p}$ by definition of σ. The second equality follows from our inductive hypothesis, implying that $x_{i,\sigma_i(\ell+1)}^* = \sum_{q=\ell+1}^{|S_i|} \frac{\lambda_{\sigma_i(\ell+1)}^p}{\mu_i + \sum_{j \in S_i^q} \lambda_j^p} \cdot \hat{\lambda}_{i,q}$. $\square$

B.6. Proof of Lemma 12

We invoke a well-known criterion for stochastic dominance relationships between time-homogeneous Markov chains. In what follows, given a countable partially-order set $\mathcal{Y}$, a subset $A \subseteq \mathcal{Y}$ is said to be monotone if, for every $x, y \in \mathcal{Y}$, $x \le y$ and $x \in A$ imply that $y \in A$.

Theorem (Stochastic dominance) *Let $Y^{(1)}, Y^{(2)}$ be two stochastic processes taking values in a countable partially-ordered set $\mathcal{Y}$. Suppose that $Y^{(1)}$ and $Y^{(2)}$ have time-homogeneous stochastic kernels $P^{(1)}$ and $P^{(2)}$. Then, $Y^{(1)} \preceq Y^{(2)}$ if and only if, for every $x, y \in \mathcal{Y}$, and every monotone set $A \subseteq \mathcal{Y}$:*

$$x \le y \ \textit{with} \ x \in A \ \textit{or} \ y \notin A \implies \sum_{z \in A} P^{(1)}_{x,z} \le \sum_{z \in A} P^{(2)}_{y,z} \tag{34}$$

For a proof of this theorem, we refer the interested readers to the work of Brandt and Last (1994, Example 3.9). López et al. (2000) provides a constructive proof of this result, where the general conditions for stochastic dominance established by Brandt and Last (1994) are expressed as a function of the associated intensity matrices. We apply the above-stated stochastic dominance theorem to the stochastic processes $Y^{(1)} = \{\tilde{X}_t\}_{t \ge 0}$ and $Y^{(2)} = \{\bar{X}_t\}_{t \ge 0}$. To this end, let $x, y \in \mathcal{X}$ be two vectors such that $x \le y$, and let $A \subseteq \mathcal{X}$ be a monotone set such that $x \in A$ or $y \notin A$. In the remainder of this section, we verify that condition (34) of the stochastic dominance theorem is met, meaning that $\sum_{z \in A} \tilde{\mathcal{Q}}_{x,z} \le \sum_{z \in A} \bar{\mathcal{Q}}_{y,z}$.

Case 1: $y \notin A$. In this case, we infer that $x \notin A$ since $x \le y$ and A is monotone. Let $\mathcal{Z}_A$ be the set of states $z \in A$ such that $\tilde{Q}_{x,z} > 0$. Observe that $z \ge x$; otherwise, $z \le x$ would imply that $x \in A$ since A is monotone, thereby contradicting our hypothesis. By inspecting the transition rate equations (30)-(31), we immediately infer that for every state $z \in \mathcal{Z}_A$ that can be reached from state x in the virtual Markov chain is of the form $z = x + e_{i,\ell}$ for some $(i, \ell) \in \mathcal{L}^{>0}_x$. Based on this observation, we define the injective mapping $\psi(z) = y + e_{i,\ell}$. Note that $\psi(z) \ge x + e_{i,\ell} = z$, thus $\psi(z) \in A$ since A is monotone. Consequently, we have

$$\sum_{z \in A} \tilde{\mathcal{Q}}_{x,z} = \sum_{z \in \mathcal{Z}_A} \tilde{\mathcal{Q}}_{x,z} = \sum_{z \in \mathcal{Z}_A} \bar{\mathcal{Q}}_{y,\psi(z)} \le \sum_{z \in A} \bar{\mathcal{Q}}_{y,z} \ ,$$

where the second equality follows from the observation that, for every state $z = x + e_{i,\ell}$ where $(i, \ell) \in \mathcal{L}^{>0}_x$, we have $\bar{\mathcal{Q}}_{x,z} = \hat{\lambda}_{i,\ell} = \tilde{\mathcal{Q}}_{y,\psi(z)}$, where the first equality proceeds from equation (28), and the latter equality holds due to equation (30).

Case 2: $x \in A$. By the monotone property of the set A, we infer from the condition $x \le y$ that $y \in A$. As such, the desired inequality $\sum_{z \in A} \tilde{\mathcal{Q}}_{x,z} \le \sum_{z \in A} \bar{\mathcal{Q}}_{y,z}$ is equivalent to $\sum_{z \in \bar{A}} \tilde{\mathcal{Q}}_{x,z} \ge \sum_{z \in \bar{A}} \bar{\mathcal{Q}}_{y,z}$ where $\bar{A} = \mathcal{X} \setminus A$. To establish the desired inequality, we construct a mapping $\phi : \mathcal{Z}_{\bar{A}} \to \mathcal{L}^{>0}_y$, where $\mathcal{Z}_{\bar{A}}$ is the collection of states $z \in \bar{A}$ such that $\bar{\mathcal{Q}}_{y,z} > 0$. For every $z \in \mathcal{Z}_{\bar{A}}$, observe that $z < y$;

otherwise, we would have $z \in A$ since A is monotone. Moreover, the transition rate equations (28)-(29) imply that there exists $(i,\ell) \in \mathcal{L}_y^{>0}$ such that $z = y - e_{i,\ell}$, for every given $z \in \mathcal{Z}_{\bar{A}}$. Based on this observation, we define $\phi(z) = (i,\ell)$. Consequently, we have

$$\sum_{z \in \bar{A}} \bar{\mathcal{Q}}_{y,z} = \sum_{z \in \mathcal{Z}_{\bar{A}}} \bar{\mathcal{Q}}_{y,z} \le \sum_{z \in \mathcal{Z}_{\bar{A}}} \tilde{\mathcal{Q}}_{x,x-e_{\phi(z)})} \le \sum_{z \in \bar{A}} \tilde{\mathcal{Q}}_{x,z} \ ,$$

where the first inequality follows from the observation that, for every state $z = y - e_{i,\ell}$ where $z \in \mathcal{Z}_{\bar{A}}$, we have $\bar{\mathcal{Q}}_{y,z} = |y^{i,\ell}| \cdot \mu_i + \sum_{j \in \mathcal{T}_y^{i,\ell}} \hat{\lambda}_j^p \le |x^{i,\ell}| \cdot \mu_i + \sum_{j \in S_i^\ell} \hat{\lambda}_j^p = \tilde{\mathcal{Q}}_{x,x-e_{\phi(z)}}$, where the first equality proceeds from equation (29), the inequality proceed from Claim 6, and the latter equality holds due to equation (31). □

Claim 6. *$|x^{i,\ell}| = |y^{i,\ell}|$ for every $(i,\ell) \in \phi\langle \mathcal{Z}_{\bar{A}} \rangle$.*

Proof. Suppose ad absurdum that $|x^{i,\ell}| < |y^{i,\ell}|$. It immediately follows that $x \le y - e_{i,\ell} = \phi^{-1}(i,\ell)$. Since A is monotone, we infer that $\phi^{-1}(i,\ell) \in A$, thereby contradicting that $\phi^{-1}(i,\ell) \in \bar{A}$. □

Appendix C: Proof of Theorem 3

We construct a family of instances $\mathcal{I}_\epsilon$, indexed by $\epsilon > 0$, such that $c_\epsilon^{\text{off}} = O(\epsilon \cdot L_\epsilon^*)$, where L_ϵ^* is the optimal value of the linear programming benchmark (see Section 4.2), and c_ϵ^{off} is the offline benchmark associated with the instance $\mathcal{I}_\epsilon$. For simplicity of notation, the reference to ϵ is sometimes omitted in our construction.

- There are two types $\mathcal{T} = \{1,2\}$, with the arrival rates $\lambda_1 = \frac{1}{\epsilon^2}$ and $\lambda_2 = \frac{1}{\epsilon}$.
- The abandonment rates are $\mu_1 = 0$ and $\mu_2 = 1$.
- The cost function is specified as follows: $c(1,1) = c(2,2) = 0$ and $c(1,2) = 1$.
- The abandonment penalty is $c_a(1) = c_a(2) = \frac{2}{\epsilon} + 1$.

Clearly, the instance $\mathcal{I}_\epsilon$ satisfies the triangle inequality (Assumption 1). It is worth emphasizing that, by picking $\mu_1 = 0$, we considerably simplify our subsequent analysis, but this specification of μ_1 is by no means necessary. Instead, we could have chosen any sufficiently small $\mu_1 = O(\epsilon)$ as a function of ϵ. The remainder of our proof is based on the following two claims.

Claim 7. *$L_\epsilon^* \ge \frac{1}{\epsilon}$.*

Proof. Given the instance $\mathcal{I}_\epsilon$, our LP benchmark takes the form of the linear program:

$$\begin{aligned}
(CB_\epsilon) \qquad \min \qquad & x_{1,2} + x_{2,1} + c_a(1) \cdot x_{1,a} + c_a(2) \cdot x_{2,a} && (35)\\
\text{s.t.} \qquad & 2 \cdot x_{1,1} + x_{1,2} + x_{2,1} + x_{1,a} = \frac{1}{\epsilon^2} \ , && \\
& 2 \cdot x_{2,2} + x_{1,2} + x_{2,1} + x_{2,a} = \frac{1}{\epsilon} \ , && (36)\\
& x_{2,a} \ge \epsilon \cdot x_{2,2} \ , \ x_{2,a} \ge \epsilon^2 \cdot x_{2,1} && (37)\\
& x_{1,1}, x_{2,2}, x_{1,2}, x_{2,1}, x_{1,a}, x_{2,a} \ge 0 &&
\end{aligned}$$

Consequently, letting $(x^*_{i,j})_{i,j\in\{1,2\}}$ and $(x^*_{i,a})_{i\in\{1,2\}}$ be optimal solution vectors for the problem (CB_ϵ), we have

$$\begin{aligned}
L^*_\epsilon &= x^*_{1,2} + x^*_{2,1} + c_a(1)\cdot x^*_{1,a} + c_a(2)\cdot x^*_{2,a} \\
&\geq \frac{1}{\epsilon} - 2\cdot x^*_{2,2} - x^*_{2,a} + c_a(2)\cdot x^*_{2,a} \\
&\geq \frac{1}{\epsilon} + \left(c_a(2) - \frac{2}{\epsilon} - 1\right)\cdot x^*_{2,a} \\
&= \frac{1}{\epsilon} \ ,
\end{aligned}$$

where the first inequality proceeds from constraint (36), and the second inequality immediately follows from constraint (37). □

Claim 8. $c^{\rm off}_\epsilon \leq 5$.

Proof. For every $t\geq 0$, we introduce an $\mathcal{F}_\infty$-measurable matching random variable M^∞_t, along with the corresponding realization graph $V^\infty_t = V_t \setminus \bigcup_{\zeta<t}\phi(M^\infty_t)$. Specifically, we consider the following case disjunction, inductively over $t\in\mathbb{R}$:

1. Suppose that $t=t_n$ is the arrival time of a vertex $n\in\mathbb{N}^*$ such that $\theta_n=2$ and $\vartheta_2(V^\infty_t)=2$. In this case, we define M^∞_t as the singleton matching formed by the pair of type-2 vertices.
2. Suppose that $t=t_n$ is the arrival time of a vertex $n\in\mathbb{N}^*$ such that $\theta_n=2$, $\vartheta_2(V^\infty_t)=1$, and $\vartheta_1(V^\infty_t)\geq 1$. If there exists no type-2 vertex arriving between t_n and $t_n+\delta_n$, we define M^∞_t as a singleton matching between n and a type-1 vertex in V^∞_t.
3. Suppose that $t\in\mathbb{N}^*$. In this case, M^∞_t is defined as a maximum-cardinality matching over the type-1 vertices of V^∞_t.
4. In any other case, we have $M^\infty_t=\emptyset$.

We begin by observing that $c^{\rm off}_\epsilon \leq c^\infty_\epsilon$, where the expected average cost $c^\infty_\epsilon = \limsup_{t\to+\infty}\frac{\mathbb{E}[C^\infty_t]}{t}$ is defined with respect to the family of cumulative costs $\{C^\infty_t\}_{t\geq 0}$, where, for every $t\geq 0$,

$$C^\infty_t = \sum_{\zeta\in\mathcal{E}(t)} c_a\cdot\left|D^\infty_\zeta\right| + \sum_{\zeta\in\mathcal{E}(t)}\sum_{e\in M^\infty_\zeta} c(e) \ , \tag{38}$$

and D^∞_ζ is the subset of vertices that leave unmatched at time ζ. The positive contributions to the cumulative cost come from type-2 vertices that either abandon the system unmatched, or get matched with a type-1 vertex. We proceed to upper bound the probability of each such event. To this end, for every $t\geq 0$, we define $N_1(t)$ and $N_2(t)$ as the number of type-1 and type-2 vertices arriving between the times $\lfloor t\rfloor$ and t, respectively. Next, we fix $t\in\mathbb{N}^*$, and we designate by n a vertex picked uniformly at random over $[N(t)]$. We define $\mathcal{E}^a_n$ as the event whereby $\theta_n=2$ and n leaves unmatched. In addition, we define $\mathcal{E}^1_n$ as the event whereby $\theta_n=2$ and n is matched with a type-1 vertex. Let ℓ be the first type-2 vertex arriving after time t_n. With these definitions at hand, observe that

$$\Pr\left[\mathcal{E}_n^1\middle|\,\theta_n = 2\right] \leq \Pr\left[t_\ell \geq t_n + \delta_n\right] = \frac{\epsilon}{1+\epsilon} \ . \tag{39}$$

Indeed, for every realization of the sojourn process, if $t_\ell < t_n + \delta_n$, then n is guaranteed to be matched with a type-2 vertex by $M_{t_n}^{\infty}$ or $M_{t_\ell}^{\infty}$ (see case 1 of the matching decisions). Using a similar sample path argument, we have

$$\begin{aligned}\Pr\left[\mathcal{E}_n^a\middle|\,\theta_n = 2\right] &\leq \Pr\left[t_\ell \geq t_n + \delta_n, N_1\left(t_n\right) \leq N_2\left(t_n\right)\right] \\ &= \Pr\left[t_\ell \geq t_n + \delta_n\right] \cdot \Pr\left[N_1\left(t_n\right) \leq N_2\left(t_n\right)\right] \\ &\leq \frac{\epsilon}{1+\epsilon} \cdot 2\epsilon \ ,\end{aligned} \tag{40}$$

where the first inequality holds by further noting that, for every realization of the sojourn process, $N_1(t_n) \geq N_2(t_n) + 1$ implies that $\vartheta_1(V_n^{\infty}) \geq 1$, meaning that vertex n cannot leave unmatched (see case 2 of the matching decisions). The equality holds since the inter-arrival times $t_\ell - t_n$ and the sojourn time δ_n are independent of $N_1(t_n)$ and $N_2(t_n)$. Lastly, the last inequality proceeds by noting that

$$\begin{aligned}&\Pr\left[N_1\left(t_n\right) \leq N_2\left(t_n\right)\right] \\ &\quad = \sum_{k=0}^{\infty} \left(\int_0^1 \Pr\left[N_1(u) + N_2(u) = k\right] du\right) \cdot \sum_{q=\lceil \frac{k}{2} \rceil}^{k} \binom{k}{q} \cdot \left(\frac{\lambda_2}{\lambda_1+\lambda_2}\right)^{q} \cdot \left(\frac{\lambda_1}{\lambda_1+\lambda_2}\right)^{k-q} \\ &\quad = \int_0^1 \Pr\left[N_1(u) + N_2(u) = 0\right] du + \frac{\lambda_2}{\lambda_1+\lambda_2} \cdot \sum_{k=1}^{\infty} \left(\int_0^1 \Pr\left[N_1(u) + N_2(u) = k\right] du\right) \cdot \Bigg(\\ &\qquad\qquad \sum_{q=\lceil \frac{k}{2} \rceil}^{k} \frac{k}{q} \cdot \binom{k-1}{q-1} \cdot \left(\frac{\lambda_2}{\lambda_1+\lambda_2}\right)^{q-1} \cdot \left(\frac{\lambda_1}{\lambda_1+\lambda_2}\right)^{k-q}\Bigg) \\ &\quad \leq \frac{1}{\lambda_1+\lambda_2} + 2 \cdot \frac{\lambda_2}{\lambda_1+\lambda_2} \\ &\quad \leq 2\epsilon \ ,\end{aligned}$$

where the first equality holds since $t_n - \lceil t_n \rceil$ is uniformly distributed over $[0,1)$ since n is picked uniformly at random over $[N(t)]$. Moreover, for every $u > 0$, the distribution of $N_2(u)$ conditional to $\{N_1(u) + N_2(u) = k\}$ is binomial with parameters k and $\frac{\lambda_2}{\lambda_1+\lambda_2}$.

By combining inequalities (39) and (40), we obtain

$$\begin{aligned}\mathbb{E}\left[C_t^{\infty}\right] &= \mathbb{E}\left[\sum_{m=1}^{N(t)} \mathbb{I}\left[\mathcal{E}_m^a, \theta_m = 2\right] \cdot c_a(2) + \mathbb{I}\left[\mathcal{E}_m^1, \theta_m = 2\right] \cdot c(1,2)\right] \\ &= \mathbb{E}\left[N(t)\right] \cdot \Pr\left[\theta_n = 2\right] \cdot \left(\Pr\left[\mathcal{E}_n^a\middle|\,\theta_n = 2\right] \cdot c_a(2) + \Pr\left[\mathcal{E}_n^1\middle|\,\theta_n = 2\right] \cdot c(1,2)\right) \\ &\leq \lambda_2 \cdot t \cdot \left(\frac{2\epsilon^2}{1+\epsilon}\left(\frac{2}{\epsilon} + 1\right) + \frac{\epsilon}{1+\epsilon}\right) \\ &\leq 5t \ .\end{aligned}$$

It immediately follows that $c_\epsilon^{\mathrm{off}} \le 5$. $\square$

$\square$

Appendix D: Proof of Theorem 4

We construct a family of instances $\mathcal{I}_\epsilon$ of the dynamic stochastic matching problem, indexed by $\epsilon \in (0,1)$, such that $c_\epsilon^{b(\eta_\epsilon)} = \Omega(\frac{1}{\epsilon}) \cdot c_\epsilon^{\pi_\epsilon^*}$, where c_ϵ^π is the expected average cost for any policy π with respect to the instance $\mathcal{I}_\epsilon$, while η_ϵ is the batching window associated with the best batching policy for the instance $\mathcal{I}_\epsilon$, and π_ϵ^* is an optimal deterministic stationary matching policy. In what follows, we describe the input parameters of the instance $\mathcal{I}_\epsilon$ for a fixed $\epsilon \in (0,1)$. For simplicity of notation, the reference to ϵ is omitted in our construction.

- There are two types $\mathcal{T} = \{1,2\}$, with respective arrival rates $\lambda_1 = \frac{1}{\epsilon^2}$ and $\lambda_2 = \frac{1}{\epsilon}$.
- The cost function is specified as follows: $c(1,1) = c(2,2) = 0$ and $c(1,2) = 1$.
- The abandonment rate is $\mu = 1$.
- The abandonment penalty is $c_a(1) = c_a(2) = 1$.

It is worth highlighting that the triangle inequality (Assumption 1) and the property of uniform abandonment rates (Assumption 2) are satisfied by the instance $\mathcal{I}_\epsilon$. The remainder of the proof is devoted to establishing the following lemmas, which complete the proof of Theorem 4.

LEMMA 13. $c_\epsilon^{\pi_\epsilon^*} \le 2$.

Proof. In the optimal matching policy, each active type-1 vertex is matched with the first subsequently arriving type-1 vertex before she abandons the system, if any. Similarly, each active type-2 vertex is matched with the first arriving type-2 vertex before she abandons the system, if any. Therefore, the expected average cost is upper bounded by

$$c_\epsilon^{\pi_\epsilon^*} \le \lambda_1 \cdot \frac{\mu}{\mu + \lambda_1} \cdot c_a(1) + \lambda_2 \cdot \frac{\mu}{\mu + \lambda_2} \cdot c_a(2) \le 2 \ ,$$

where the first inequality holds by observing that, under the optimal matching policy, the probability that a type-1 vertex abandons the system conditional on being active is precisely of $\frac{\mu}{\mu+\lambda_1}$. Similarly, the probability that a type-2 vertex abandons the system conditional on being active is of $\frac{\mu}{\mu+\lambda_2}$. $\square$

LEMMA 14. $c_\epsilon^{b(\eta_\epsilon)} \ge \frac{(e-1)}{2e^2} \cdot \frac{(1-\epsilon)^3}{\epsilon}$.

Proof. We separately examine three regimes relative to the batching window parameter η_ϵ.

Case 1: $\eta_\epsilon \leq \frac{\epsilon^2}{1+\epsilon}$. For every $k \in \mathbb{N}^*$, we seek to characterize the probability that there exists an odd number of type-1 and an odd number of type-2 vertices in $V^{b(\eta_\epsilon)}_{k\cdot\eta_\epsilon}$; this event is denoted by $\mathcal{E}_k$. The key observation is that, conditional on $\mathcal{E}_k$, the matching $M^{b(\eta_\epsilon)}_{k\cdot\eta_\epsilon}$ necessarily contains an edge between a vertex of type 1 and a vertex of type 2. Consequently, we derive a lower boun on the expected average cost incurred by the batching policy $b(\eta_\epsilon)$:

$$c^{b(\eta_\epsilon)}_\epsilon \geq \lim_{K\to+\infty} \frac{1}{K\cdot\eta_\epsilon} \cdot \sum_{k=1}^{K} \Pr\left[\mathcal{E}_k\right] \cdot c\left(1,2\right) \tag{41}$$

Now, in order to lower bound $\Pr[\mathcal{E}_k]$ for each $k \in \mathbb{N}^*$, we begin by observing that $\bar{V}^{b(\eta_\epsilon)}_{(k-1)\cdot\eta_\epsilon}$ is comprised of at most one vertex; otherwise, any two remaining vertices can always be matched, thereby contradicting the maximal cardinality of $M^{b(\eta_\epsilon)}_{(k-1)\cdot\eta_\epsilon}$. More specifically, the main ingredient of our analysis in Case 1 consists in showing that $\Pr[\vartheta(\bar{V}^{b(\eta_\epsilon)}_{(k-1)\cdot\eta_\epsilon}) = e_1]$ is lower bounded by a universal constant. To this end, we construct a time-homogeneous Markov chain $\{L_k\}_{k\geq 0}$ taking binary values such that L_k is stochastically smaller or equal to $\vartheta_1(\bar{V}^{b(\eta_\epsilon)}_{k\cdot\eta_\epsilon})$ for every $k \in \mathbb{N}$. Specifically, the Markov chain $\{L_k\}_{k\geq 0}$ is uniquely defined by the marginal probabilities:

$$\begin{cases} \Pr\left[L_0 = 0\right] = 1 \ , & \\ \Pr\left[L_k = 1 \middle| L_{k-1} = 0\right] = \dfrac{(1-\epsilon)}{e} \cdot \lambda_1 \cdot \eta_\epsilon & \text{for every } k \in \mathbb{N}^* \ , \qquad (42) \\ \Pr\left[L_k = 0 \middle| L_{k-1} = 1\right] = 3 \cdot \lambda_1 \cdot \eta_\epsilon & \text{for every } k \in \mathbb{N}^* \ . \qquad (43) \end{cases}$$

To establish the desired stochastic dominance property, let $N_1(t)$ and $N_2(t)$ denote the number type-1 and type-2 agents that arrive in the time interval $[0,t]$, respectively. Consequently, we have

$$\begin{aligned} \Pr\left[\vartheta_1(\bar{V}^{b(\eta_\epsilon)}_{k\cdot\eta_\epsilon}) = 1 \middle| \vartheta_1(\bar{V}^{b(\eta_\epsilon)}_{(k-1)\cdot\eta_\epsilon}) = 0\right] &\geq \Pr\left[\delta_1 > \eta_\epsilon\right] \cdot \Pr\left[N_1\left(\eta_\epsilon\right) = 1\right] \cdot \Pr\left[N_2\left(\eta_\epsilon\right) = 0\right] \\ &\geq e^{-\eta_\epsilon} \cdot \lambda_1 \cdot \eta_\epsilon \cdot e^{-\lambda_1\cdot\eta_\epsilon} \cdot e^{-\lambda_2\cdot\eta_\epsilon} \\ &\geq \frac{(1-\epsilon)}{e} \cdot \lambda_1 \cdot \eta_\epsilon \\ &= \Pr\left[L_k = 1 \middle| L_{k-1} = 0\right] \ , \end{aligned}$$

where the first inequality proceeds by noting that $\vartheta_1(\bar{V}^{b(\eta_\epsilon)}_{k\cdot\eta_\epsilon}) = 1$ if $\vartheta_1(\bar{V}^{b(\eta_\epsilon)}_{(k-1)}) = 0$ and there is precisely one type-1 vertex arriving in the interval $((k-1)\cdot\eta_\epsilon, k\cdot\eta_\epsilon]$ with a corresponding sojourn time greater than η_ϵ. The last equality holds by equation (42). Furthermore, we have

$$\begin{aligned} \Pr\left[\vartheta_1(\bar{V}^{b(\eta_\epsilon)}_{k\cdot\eta_\epsilon}) = 0 \middle| \vartheta_1(\bar{V}^{b(\eta_\epsilon)}_{(k-1)\cdot\eta_\epsilon}) = 1\right] &\leq \Pr\left[\delta_1 < \eta_\epsilon\right] + \Pr\left[N_1\left(\eta_\epsilon\right) + N_2\left(\eta_\epsilon\right) \geq 1\right] \\ &= 1 - e^{-\eta_\epsilon} + 1 - e^{-(\lambda_1+\lambda_2)\cdot\eta_\epsilon} \\ &\leq 3 \cdot \lambda_1 \cdot \eta_\epsilon \\ &= \Pr\left[L_k = 0 \middle| L_{k-1} = 1\right] \ , \end{aligned}$$

where the first inequality proceeds by observing that, conditional on the event $\{\vartheta_1(\bar{V}^{b(\eta_\epsilon)}_{(k-1)}) = 1\}$, the absence of any type-1 vertex in the realization graph right after time $k \cdot \eta_\epsilon$ implies that one of the two following conditions is necessarily met: (i) the type-1 vertex available at time $(k-1) \cdot \eta_\epsilon$ abandons the system by time $k \cdot \eta_\epsilon$; (ii) there is at least one vertex arriving in the interval $((k-1) \cdot \eta_\epsilon, k \cdot \eta_\epsilon]$. The last equality holds by equation (43).

Having shown that L_k is stochastically smaller or equal to than $\vartheta_1(\bar{V}^{b(\eta_\epsilon)}_{k \cdot \eta_\epsilon})$ for every $k \in \mathbb{N}$, we obtain a lower bound on the expected average cost using inequality (41):

$$\begin{aligned}
c^{b(\eta_\epsilon)}_\epsilon &\geq \lim_{K \to +\infty} \frac{1}{K \cdot \eta_\epsilon} \cdot \sum_{k=1}^{K} \Pr\left[\mathcal{E}_k\right] \\
&\geq \lim_{K \to +\infty} \frac{1}{K \cdot \eta_\epsilon} \cdot \sum_{k=1}^{K} \Pr\left[\vartheta_1(V^{b(\eta_\epsilon)}_{k \cdot \eta_\epsilon}) = 1, \vartheta_2(V^{b(\eta_\epsilon)}_{k \cdot \eta_\epsilon}) = 1 \middle| \vartheta_1(\bar{V}^{b(\eta_\epsilon)}_{(k-1) \cdot \eta_\epsilon}) = 1\right] \cdot \Pr\left[\vartheta_1(\bar{V}^{b(\eta_\epsilon)}_{(k-1) \cdot \eta_\epsilon}) = 1\right] \\
&\geq \lim_{K \to +\infty} \frac{1}{K \cdot \eta_\epsilon} \cdot \sum_{k=1}^{K} \left(\Pr\left[\delta_1 > \eta_\epsilon\right]\right)^2 \cdot \Pr\left[N_2\left(\eta_\epsilon\right) = 1\right] \cdot \Pr\left[L_{k-1} = 1\right] \\
&= \lambda_2 \cdot e^{-(2+\lambda_2) \cdot \eta_\epsilon} \cdot \left(\lim_{K \to +\infty} \frac{1}{K} \cdot \sum_{k=1}^{K} \Pr\left[L_{k-1} = 1\right]\right) \\
&\geq \frac{1}{(1+3e)} \cdot \frac{(1-\epsilon)^3}{\epsilon} ,
\end{aligned}$$

where the first inequality proceeds by observing that, conditional on $\{\vartheta_1(\bar{V}^{b(\eta_\epsilon)}_{(k-1) \cdot \eta_\epsilon}) = 1\}$, the outcomes $\vartheta_1(V^{b(\eta_\epsilon)}_{k \cdot \eta_\epsilon}) = 1$ and $\vartheta_2(V^{b(\eta_\epsilon)}_{k \cdot \eta_\epsilon}) = 1$ are necessarily implied by the fact that if there is precisely one type-2 vertex arriving in the interval $((k-1) \cdot \eta_\epsilon, k \cdot \eta_\epsilon]$ and there is no abandonment in the same period of time. The last inequality proceeds by utilizing our Case 1 hypothesis (i.e., $\eta_\epsilon \leq \frac{\epsilon^2}{1+\epsilon}$) and by characterizing the stationary distribution of the Markov chain $\{L_k\}_{k \geq 0}$ using equations (42) and (43).

Case 2: $\frac{\epsilon}{1+\epsilon} \geq \eta_\epsilon \geq \frac{\epsilon^2}{1+\epsilon}$. Following the same line of reasoning as in Case 1, the goal is to lower bound $\Pr[\mathcal{E}_k]$ for every $k \in \mathbb{N}^*$. To this end, we will utilize the following technical claim, which proof is deferred to the end of this section.

Claim 9. $\frac{1}{2} \cdot e^{-\eta_\epsilon} \cdot (1 - e^{-\lambda_1 \cdot \eta_\epsilon}) \leq \Pr\left[\vartheta_1(V_{\eta_\epsilon}) \notin 2 \cdot \mathbb{N}\right] \leq \frac{1}{2}$.

Next, we separately analyze the probability of $\mathcal{E}_k$ conditional on three mutually exclusive events.

1. *Case (2a):* $\bar{V}^{b(\eta_\epsilon)}_{(k-1) \cdot \eta_\epsilon} = \emptyset$. We denote by $\ell \in \mathbb{N}^*$ the first agent of type 2 arriving after time $(k-1) \cdot \eta_\epsilon$. Consequently, we have:

$$\begin{aligned}
\Pr\left[\mathcal{E}_k \middle| \bar{V}^{b(\eta_\epsilon)}_{(k-1) \cdot \eta_\epsilon} = \emptyset\right] &= \Pr\left[\vartheta_1(V_{\eta_\epsilon}) \notin 2 \cdot \mathbb{N}\right] \cdot \Pr\left[\vartheta_2(V_{\eta_\epsilon}) \notin 2 \cdot \mathbb{N}\right] \\
&\geq \Pr\left[\vartheta_1(V_{\eta_\epsilon}) \notin 2 \cdot \mathbb{N}\right] \cdot \Pr\left[N_2(\eta_\epsilon) = 1, \delta_\ell > \eta_\epsilon\right]
\end{aligned}$$

$$\geq \frac{1}{2} \cdot e^{-\eta_\epsilon} \cdot \left(1 - e^{-\lambda_1 \cdot \eta_\epsilon}\right) \cdot \eta_\epsilon \cdot \lambda_2 \cdot e^{-\eta_\epsilon \cdot \lambda_2} \cdot e^{-\eta_\epsilon}$$
$$\geq \frac{(e-1)}{2e^2} \cdot \frac{(1-\epsilon)^2}{\epsilon} \cdot \eta_\epsilon \ , \tag{44}$$

where the first equation holds since the arrivals of type-1 and type-2 vertices are independent, and the sojourn process satisfies the Markov property. The first inequality holds due to the inclusions of the events $\{N_2(\eta_\epsilon) = 1, \delta_\ell > \eta_\epsilon\} \subseteq \{\vartheta_2(V_{\eta_\epsilon}) = 1\} \subseteq \{\vartheta_2(V_{\eta_\epsilon}) \notin 2 \cdot \mathbb{N}\}$. The second inequality immediately follows from Claim 9.

2. *Case (2b):* $\vartheta(\bar{V}^{b(\eta_\epsilon)}_{(k-1)\cdot\eta_\epsilon}) = e_1$. Here, we have

$$\Pr\left[\mathcal{E}_k | \, \vartheta\left(\bar{V}^{b(\eta_\epsilon)}_{(k-1)\cdot\eta_\epsilon}\right) = e_1\right] \geq \Pr\left[\delta_1 > \eta_\epsilon\right] \cdot \Pr\left[\vartheta_1(V_{\eta_\epsilon}) \in 2 \cdot \mathbb{N}\right] \cdot \Pr\left[\vartheta_2(V_{\eta_\epsilon}) \notin 2 \cdot \mathbb{N}\right]$$
$$\geq \frac{1}{2} \cdot \Pr\left[\delta_1 > \eta_\epsilon\right] \cdot \Pr\left[\vartheta_2(V_{\eta_\epsilon}) \notin 2 \cdot \mathbb{N}\right]$$
$$\geq \frac{1}{2e} \cdot \frac{(1-\epsilon)}{\epsilon} \cdot \eta_\epsilon \ , \tag{45}$$

where the second inequality immediately follows from Claim 9, and the third inequality proceeds from computations similar to those leading to inequality (44).

3. *Case (2c):* $\vartheta(\bar{V}^{b(\eta_\epsilon)}_{(k-1)\cdot\eta_\epsilon}) = e_2$. Here, we have

$$\Pr\left[\mathcal{E}_k | \, \vartheta\left(\bar{V}^{b(\eta_\epsilon)}_{(k-1)\cdot\eta_\epsilon}\right) = e_2\right] \geq \Pr\left[\delta_1 > \eta_\epsilon\right] \cdot \Pr\left[\vartheta_1\left(V_{\eta_\epsilon}\right) \notin 2 \cdot \mathbb{N}\right] \cdot \Pr\left[\vartheta_2\left(V_{\eta_\epsilon}\right) \in 2 \cdot \mathbb{N}\right]$$
$$\geq \Pr\left[\delta_1 > \eta_\epsilon\right] \cdot \Pr\left[\vartheta_1(V_{\eta_\epsilon}) \notin 2 \cdot \mathbb{N}\right] \cdot \Pr\left[N_2(t) = 0\right]$$
$$\geq \frac{(e-1)}{2e^2} \cdot \frac{(1-\epsilon)}{\epsilon} \cdot \eta_\epsilon \ , \tag{46}$$

where the second inequality holds due to the inclusions of the events $\{N_2(t) = 0\} \subseteq \{\vartheta_2(V_{\eta_\epsilon}) = 0\} \subseteq \{\vartheta_2(V_{\eta_\epsilon}) \in 2 \cdot \mathbb{N}\}$. The third inequality immediately follows from Claim 9 and our Case 2 hypothesis (i.e., $\eta_\epsilon \leq \frac{\epsilon}{1+\epsilon}$).

By plugging inequalities (44)-(46) into inequality (41), we obtain

$$c^{b(\eta_\epsilon)}_\epsilon \geq \lim_{K \to +\infty} \frac{1}{K \cdot \eta_\epsilon} \cdot \sum_{k=1}^{K} \Pr\left[\mathcal{E}_k\right] \geq \lim_{K \to +\infty} \frac{1}{K \cdot \eta_\epsilon} \cdot \sum_{k=1}^{K} \frac{(e-1)}{2e^2} \cdot \frac{(1-\epsilon)^2}{\epsilon} \cdot \eta_\epsilon = \frac{(e-1)}{2e^2} \cdot \frac{(1-\epsilon)^2}{\epsilon} \ .$$

Case 3: $\eta_\epsilon > \frac{\epsilon}{1+\epsilon}$. This case is relatively simple. The idea is to lower bound the expected average abandonment penalty incurred by abandonments of type-1 vertices. To this end, we denote by Y_k the random number of type-1 vertices arriving during the time interval $((k-1) \cdot \eta_\epsilon, k \cdot \eta_\epsilon]$, and abandoning the system before time $k \cdot \eta_\epsilon$. Having defined Y_k for every $k \in \mathbb{N}$, we lower bound the expected average cost as follows:

$$c^{b(\eta_\epsilon)}_\epsilon \geq \lim_{K \to +\infty} \frac{1}{K \cdot \eta_\epsilon} \cdot \sum_{k=1}^{K} c_a(1) \cdot \mathbb{E}\left[Y_k\right] = \frac{1}{\eta_\epsilon} \cdot \mathbb{E}\left[Y_1\right] \ ,$$

where the last equality holds since the sojourn process is stationary. On the other hand, we have

$$\mathbb{E}\left[Y_1\right] \geq \lambda_1 \cdot \eta_\epsilon \cdot \left(1-e^{-\eta_\epsilon}\right) \geq \left(1-\frac{1}{e}\right) \cdot \frac{(1-\epsilon)}{\epsilon} \cdot \eta_\epsilon,$$

where the first inequality holds since $\lambda_1 \cdot \eta_\epsilon$ is the expected number of type-1 arrivals during the batching window, and $1-e^{-\eta_\epsilon}$ is a lower bound on their survival probabilities. The second inequality proceeds from our Case 3 hypothesis (i.e., $\eta_\epsilon > \frac{\epsilon}{1+\epsilon}$). Lastly, by combining the above two inequalities, we obtain

$$c_\epsilon^{b(\eta_\epsilon)} \geq \left(1-\frac{1}{e}\right) \cdot \frac{(1-\epsilon)}{\epsilon(1+\epsilon)} \ .$$

□

Proof of Claim 9. We begin by observing that:

$$\begin{aligned}
&\Pr\left[\vartheta_1(V_t) \notin 2 \cdot \mathbb{N}\right] \\
&= \int_0^t \lambda_1 \cdot e^{-\lambda_1 \cdot u} \cdot \Pr\left[\vartheta_1(V_t) \notin 2 \cdot \mathbb{N} \middle| t_1 = u, \theta_1 = 1\right] du \\
&= \int_0^t \lambda_1 \cdot e^{-\lambda_1 \cdot u} \cdot \left(1 - \Pr\left[\vartheta_1(V_{t-u}) \notin 2 \cdot \mathbb{N}\right]\right) \cdot \Pr\left[\delta_1 > t-u\right] du \\
&\qquad + \int_0^t \lambda_1 \cdot e^{-\lambda_1 \cdot u} \cdot \Pr\left[\vartheta_1(V_{t-u}) \notin 2 \cdot \mathbb{N}\right] \ \cdot \Pr\left[\delta_1 \leq t-u\right] du \\
&= \int_0^t \lambda_1 \cdot e^{-\lambda_1 \cdot u} \cdot e^{-(t-u)} du + \int_0^t \lambda_1 \cdot e^{-\lambda_1 \cdot u} \cdot \Pr\left[\vartheta_1(V_{t-u}) \notin 2 \cdot \mathbb{N}\right] \cdot \left(1 - 2e^{-(t-u)}\right) du \ . \qquad (47)
\end{aligned}$$

Consequently, we define the function $f(u) = \Pr[\vartheta_1(V_u) \notin 2 \cdot \mathbb{N}]$ for every $u \in [0, \eta_\epsilon]$. Suppose that there exists $u \in [0, \eta_\epsilon]$ for which $f(u) \geq \frac{1}{2}$, and let u^* be the minimal such $u \in [0, \eta_\epsilon]$. Since f is continuous, by noting that $f(u) < \frac{1}{2}$ for every $u \in B(0, \delta)$ given a sufficiently small neighborhood $\delta > 0$, we obtain

$$f\left(u^*\right) < \int_0^{u^*} \lambda_1 \cdot e^{-\lambda_1 \cdot u} \cdot e^{-(u^*-u)} du + \int_0^{u^*} \lambda_1 \cdot e^{-\lambda_1 \cdot u} \cdot \frac{1}{2} \cdot \left(1 - 2e^{-(u^*-u)}\right) du \leq \frac{1}{2} \ ,$$

thereby contradicting that $f(u^*) = \frac{1}{2}$. Consequently, we have just shown that $f(u) \leq \frac{1}{2}$ for every $u \in [0, \eta_\epsilon]$. Moreover, using a line of argumentation similar to equation (47), we have:

$$\begin{aligned}
f\left(\eta_\epsilon\right) &\geq \int_0^t \lambda_1 \cdot e^{-\lambda_1 \cdot u} \cdot \left(1 - \Pr\left[\vartheta_1(V_{t-u}) \notin 2 \cdot \mathbb{N}\right]\right) \cdot \Pr\left[\delta_1 > \eta_\epsilon\right] du \\
&\geq \frac{1}{2} \cdot e^{-\eta_\epsilon} \cdot \int_0^t \lambda_1 \cdot e^{-\lambda_1 \cdot u} du \\
&= \frac{1}{2} \cdot e^{-\eta_\epsilon} \cdot \left(1 - e^{-\lambda_1 \cdot \eta_\epsilon}\right) \ .
\end{aligned}$$

□

□

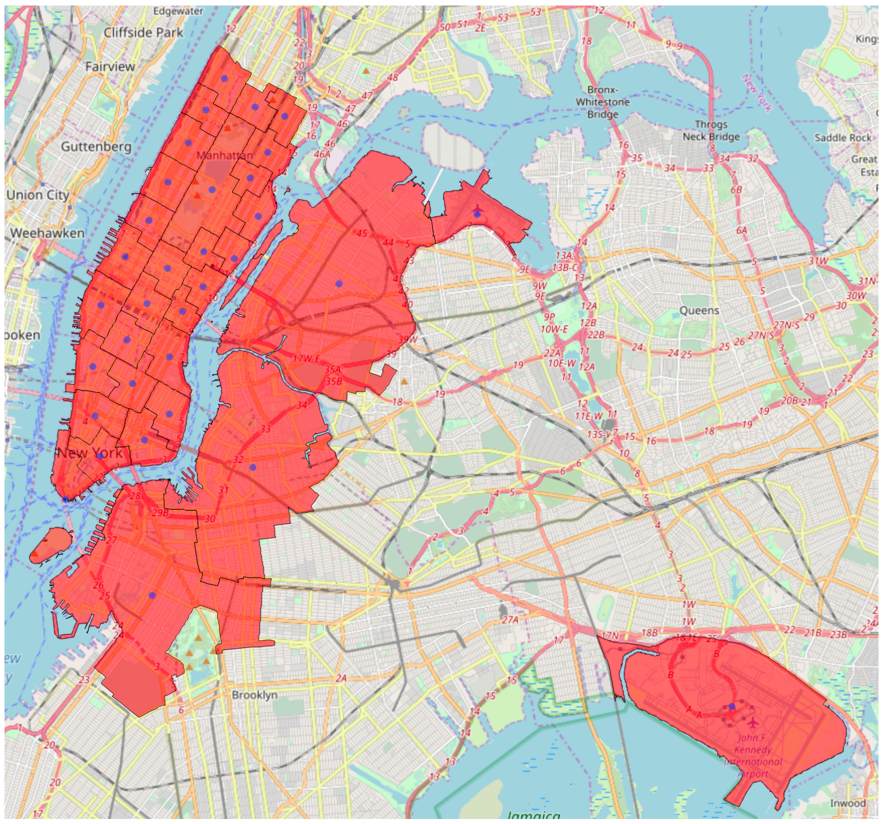
Appendix E: Empirical case study

E.1. Data cleaning

We observe that a significant fraction of the trip records have location coordinates of precisely zero, or identical pick-up and drop-off locations. These inputs of location are likely inaccurate or erroneous, and thus, we filter the corresponding observations from the data sets. Furthermore, we take a conservative approach with respect to timestamp information by eliminating all observations having a trip length shorter than 5 minutes or exceeding 1 hour. Overall, we filter approximately 25% of all trip observations.

E.2. Clustering method

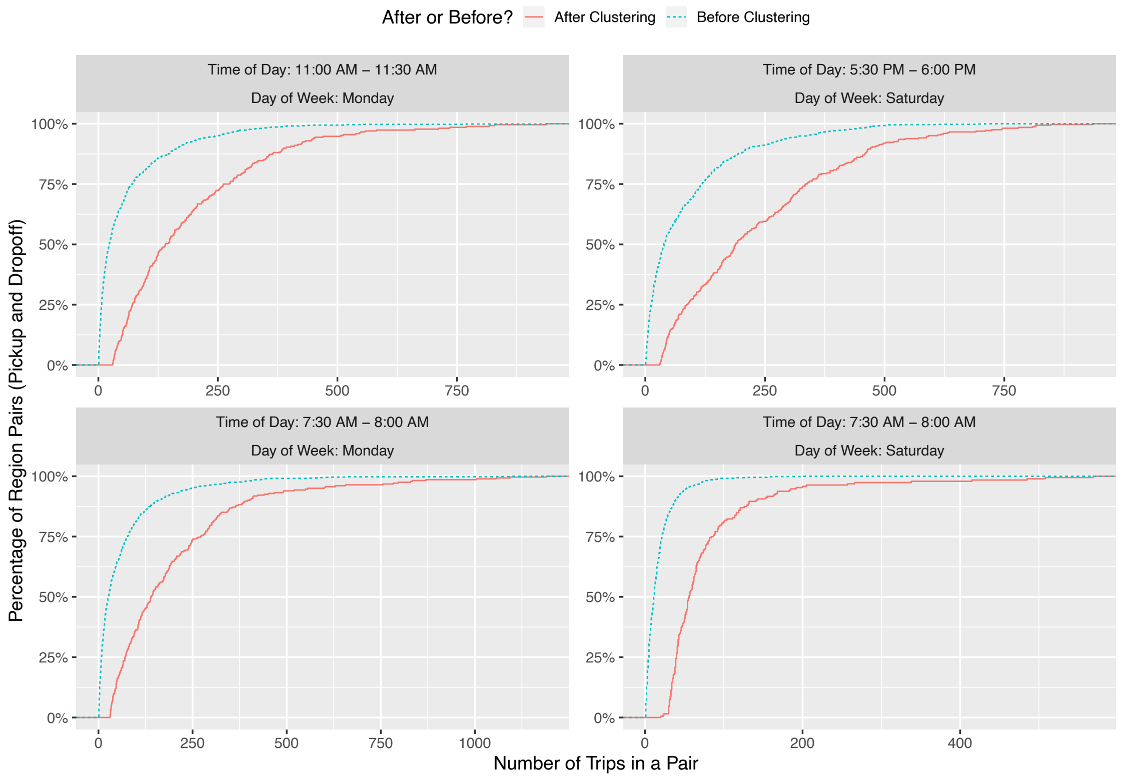
Figure 5 Decomposition of the NY area into 33 regions based on census-tract boundaries.



Our modeling approach hinges on defining a collection of types $i \in \mathcal{T}$ and their corresponding arrival rates λ_i (these model ingredients specify the agents' arrival process). Intuitively, each type should describe "similar" riders in terms of pick-up and drop-off locations. To this end, we develop a data-driven clustering method that identifies riders with close-enough pick-up and drop-off locations. This method proceeds in two steps:

- First, we build on the approach employed by Buchholz (2021) to decompose the NY area into "homogeneous regions". Specifically, the method consists in aggregating census-tract boundaries in NY to obtain 39 homogeneous regions with respect to the arrival rates, travel times and transition probabilities. Using an analogous method, we partition the NY area into 33 regions by aggregating census tracts. Centroid of a region is used as the pick-up or the drop-off location for a trip originating from or ending at the region, respectively. The resulting regions are visualized in Figure 5.
- Second, we assign each pair of pick-up and drop-off regions to a rider type using a clustering algorithm. To this end, the trip data is initially partitioned into 33 pre-clusters $\mathcal{C}_1, \ldots, \mathcal{C}_{33}$, each of which corresponds to the collection of trips that originate in the same region. Next, we run a K-means algorithm based on the drop-off locations of the riders within each pre-cluster $\mathcal{C}_s$. Hence, the collections of riders with nearly-identical pick-up locations described by each of the 33 pre-clusters are further refined into clusters having drop-off locations in close proximity. The number of clusters K_s within each pre-cluster $\mathcal{C}_s$ is chosen proportional to the number of trips $|\mathcal{C}_s|$, namely $K_s \approx \frac{|\mathcal{C}_s|}{\sum_{s=1}^{33} |\mathcal{C}_s|} \cdot |\mathcal{T}|$.

Figure 6 Empirical cumulative density for the number of trips in pairs of pick-up and drop-off regions.



The choice of the number of agent types $|\mathcal{T}|$ is subject to a trade-off between granularity and sparsity. Specifically, if the number of rider types $|\mathcal{T}|$ is too small, the geographic granularity is coarse, meaning that pick-up and drop-off locations could be highly variable within each single type. For example, if the trips are clustered solely on the basis of their pick-up region, the coefficient of variation of the trip lengths within a cluster is of 59.6% on average. Conversely, if the number of rider types $|\mathcal{T}|$ is large, the arrival rates cannot be accurately estimated due to the small sample size. Indeed, certain combinations of pick-up and drop-off regions exhibit a very small number of trips at certain times of day and days of week. For example, as illustrated by Figure 6, for 27% of the pairs of pick-up and drop-off regions, there are fewer than 8 trips throughout our 8-week time period that originate in the pick-up region and terminate in the drop-off region in question. Hence, to deal with data sparsity, we cluster the pairs of pick-up and drop-off regions; as shown by Figure 6, *all* resulting clusters contain more than 8 trips.

E.3. Computation of $\{\bar{\mathbf{c}}_{\mathbf{i}}\}_{\mathbf{i}\in\mathcal{T}}$

In Section 4.1, $\bar{c}_i$ is defined as the minimum of the function $f_i(.)$, where

$$f_i(\mathcal{A}) = \frac{\mu}{\mu + \sum_{k\in\mathcal{A}} \lambda_k} \cdot c_a(i) + \sum_{j\in\mathcal{A}} \frac{\lambda_j}{\mu + \sum_{k\in\mathcal{A}} \lambda_k} \cdot c(i,j) \ ,$$

for every $\mathcal{A} \subseteq \mathcal{T}$. While $\bar{c}_i$ can be computed by solving the linear program (AN_i) introduced in Section 4.1, we utilize a fixed-point formulation. To this end, observe that $\bar{c}_i = f_i(\mathcal{A}_i^*)$ and $\mathcal{A}_i^* = \{j \in \mathcal{T} : c(i,j) \leq f_i(\mathcal{A}_i^*)\}$ by Lemma 3. Thus, $\bar{c}_i$ is a solution of the fixed-point equation $f_i(\mathcal{A}_i(\alpha)) = \alpha$, where $\mathcal{A}_i(\alpha) = \{j \in \mathcal{T} | c(i,j) \leq \alpha\}$. Consequently, we define $\bar{c}_i$ as the empirical mean of the fixed-point solutions over all type-i riders n_1 in the training set. It is worth noting that we utilize normalized variant of the cost function $n_2 \mapsto \frac{\ell_{n_1}}{\ell_{n_1}+\ell_{n_2}} \cdot c(n_1,n_2)$.

E.4. Additional figures and plots

Table 2 Total cost of the proposed algorithms and the LP-benchmark (CB) .

Departure rate (μ)	Constant penalty (c_p)	Batching	Vertex-additive	Threshold	LP-benchmark (CB)
0.5	0.01	48.55	43.33	48.62	37.09
1	0.01	49.28	45.41	51.35	38.14
2	0.01	50.33	47.86	52.80	39.55
0.5	0.05	49.76	47.2	53.31	38.99
1	0.05	51.14	48.56	54.14	40.62
2	0.05	53.02	51.88	54.32	42.76

Figure 7 Aggregate demand for NY taxis on various days of week and times of day (January-February 2013).

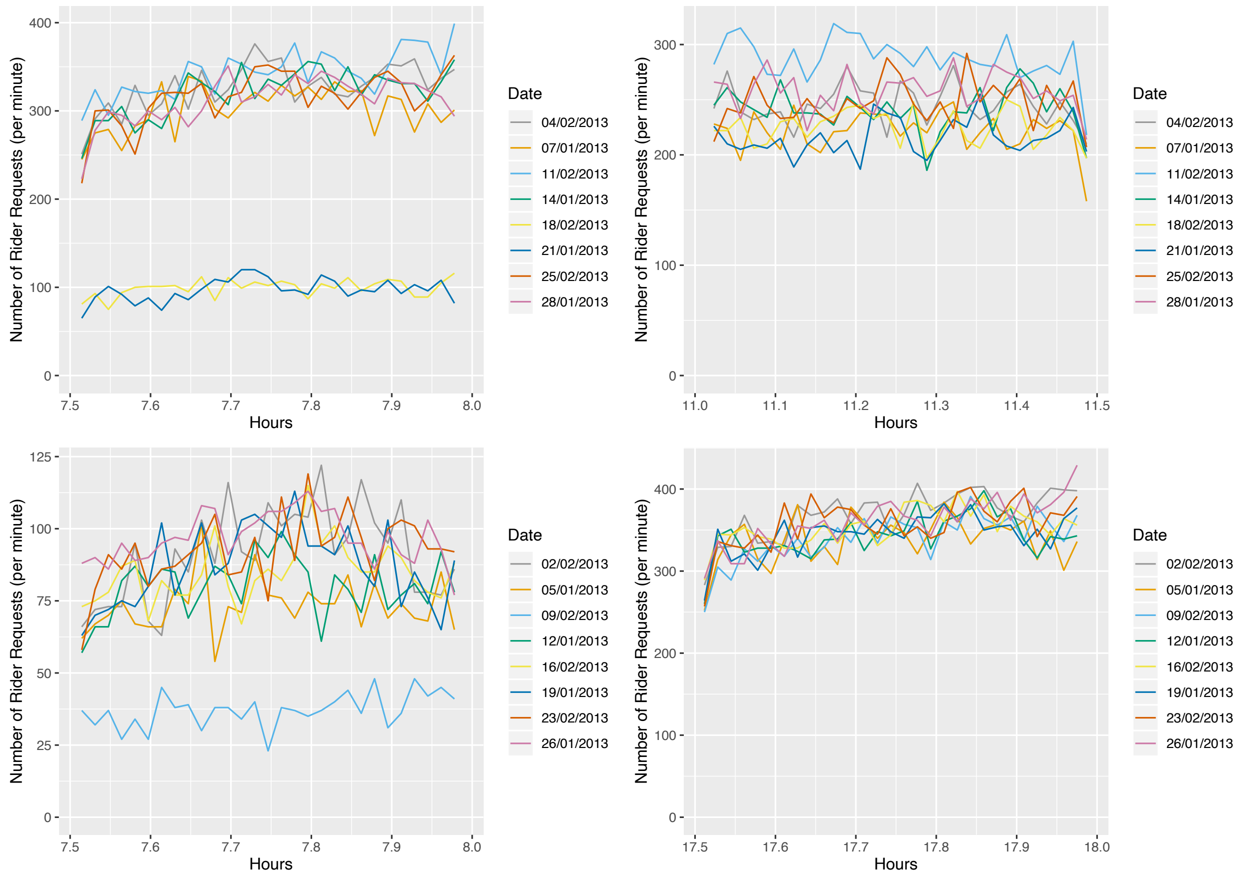


Figure 8 Pictorial illustration of all possible vehicle routes for matching two riders.

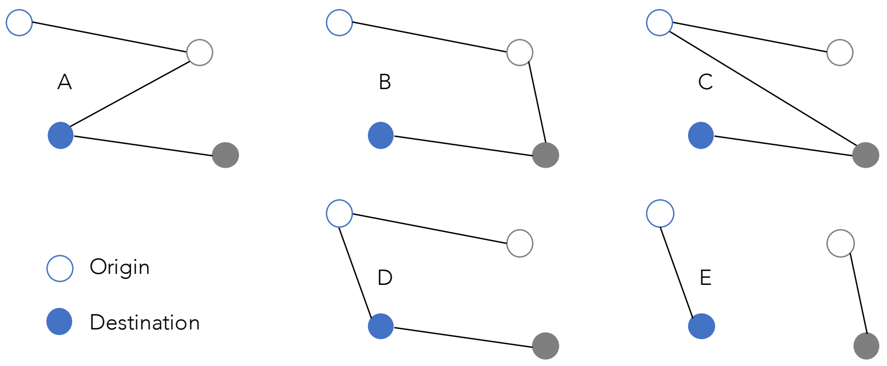
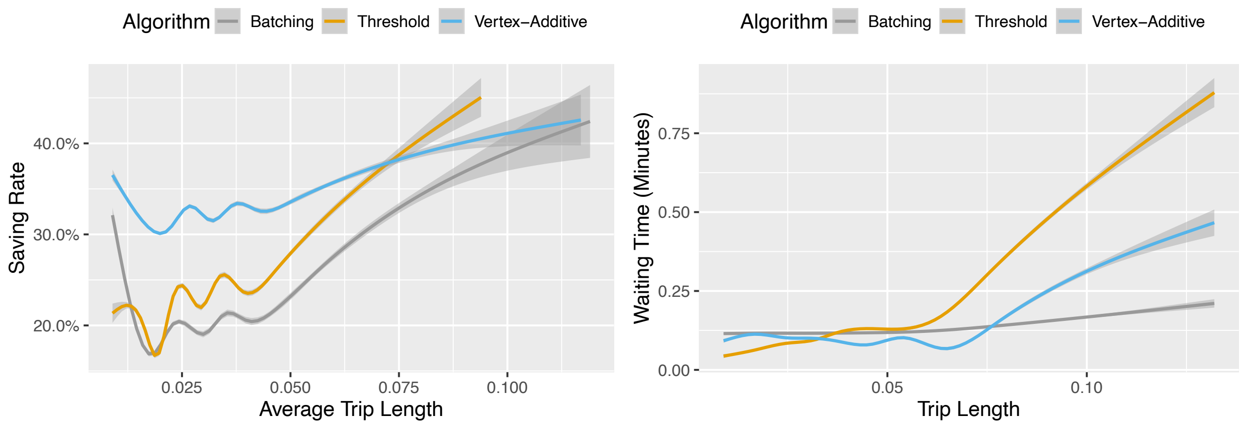
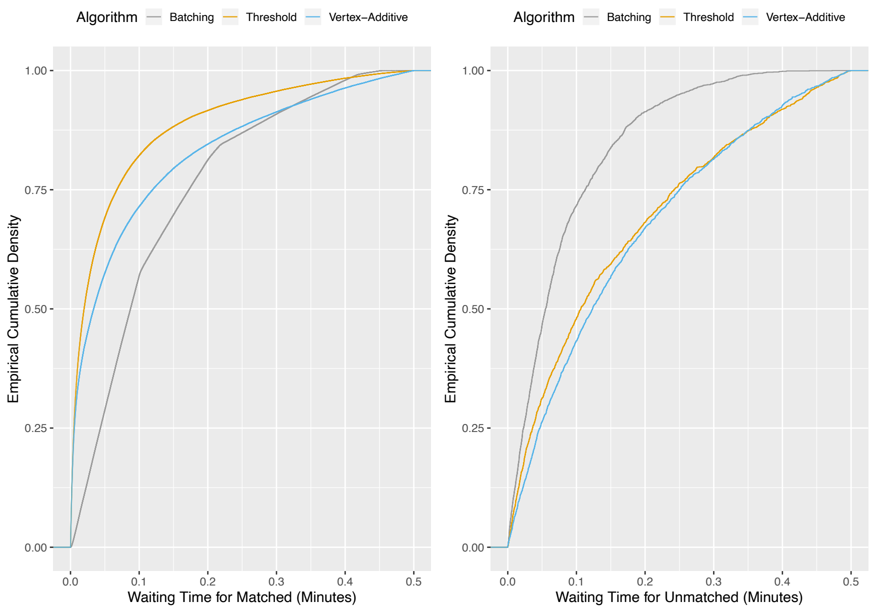


Figure 9 On the left, saving rates achieved by the tested algorithms as a function of trip length. On the right, average waiting times of riders as a function of trip length.



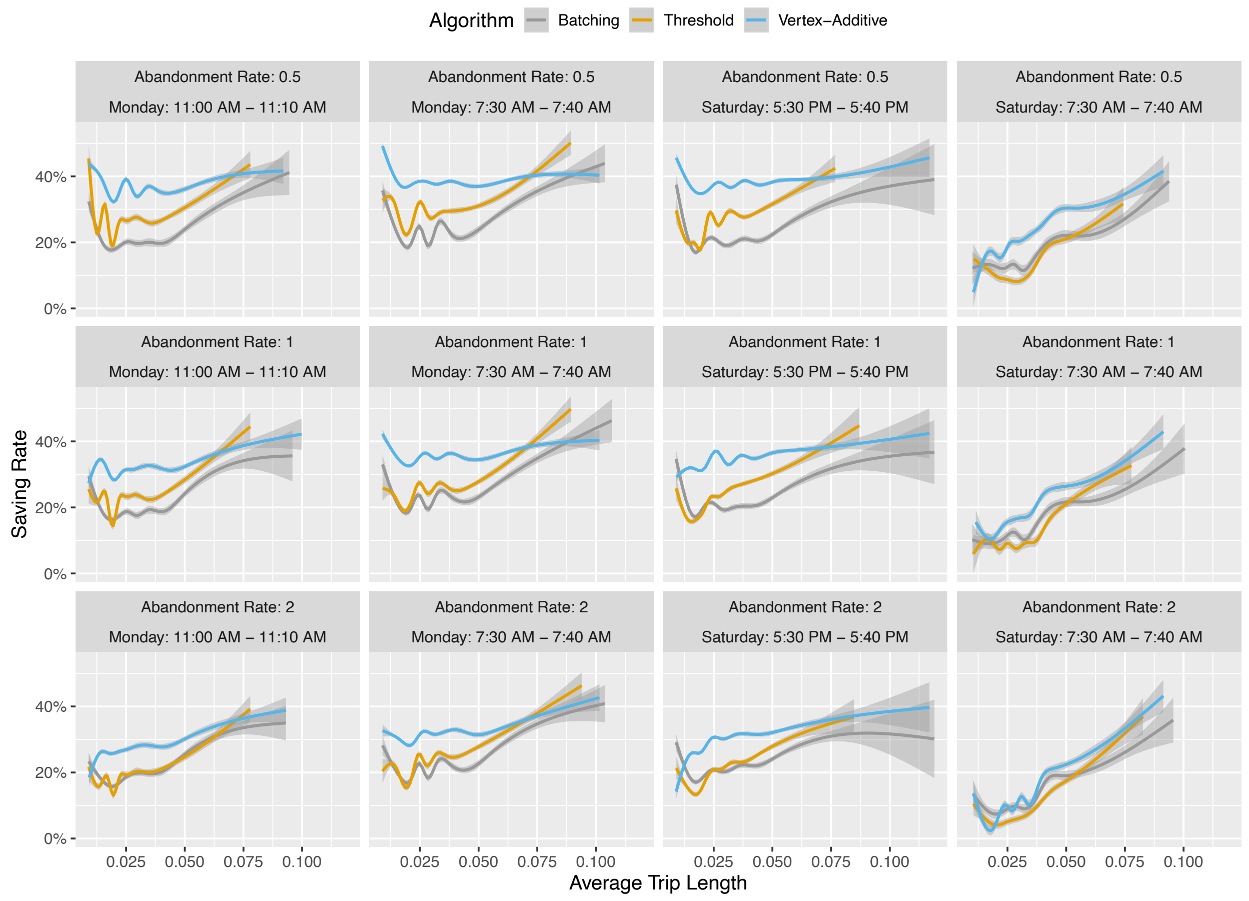
Note. Each algorithm is tuned to achieve a match rate of 90%(±1%). The "smooth" curves are generated using a local regression method (*geom_smooth* function of the R package *ggplot2*).

Figure 10 Empirical cumulative density of waiting times for each tested algorithm.



Note. Each algorithm is tuned to achieve a match rate of 90%(±1%).

Figure 11 Savings rates achieved by the tested algorithms as a function of trip length in various market conditions.



Note. Each algorithm is tuned to achieve a match rate of 90% (±1%).