On the Move: A Study of Dynamic Matching Mechanisms in BlaBlaCar's Mobility Marketplace

Bachelor's Thesis

Presented to the
Faculty of Management, Economics and Social Sciences at the
University of Cologne

In Partial Fulfillment of the Requirements for the Degree of
Bachelor of Science (B.Sc.)

First supervisor: Prof. Dr. Alexander Westkamp
Second supervisor: Prof. Dr. Christoph Schottmüller

Submitted in July 2024 by

Timon Josias Otis Brinker

Matriculation Number: 7390095

Contents

List of Figures

List of Tables

Acronyms

a.s.	almost surely
CLP	Continous Linear Programming
ETA	Estimated Time of Arrival
FD	First Dispatch
LP	Linear Programming
MDR	Maximum Dispatch Radius
RWQP	Randomized Weighted Queue Policy
WGC	Wild Goose Chase

1 Introduction

Ride-sharing has become increasingly popular over the last 15 years, experiencing only a temporary decline during the COVID-19 pandemic (Ivaldi and Palikot, 2023). There are two main types of ride-sharing. The first includes services such as *Uber* in North America and Europe, and *DiDi* in China, where drivers use their own vehicles to transport riders, offering a more affordable alternative to traditional taxis. This cost difference has occasionally sparked protests from taxi drivers (Deutsche Welle, 2019). The second type, known as carpooling, involves drivers offering free seats in their vehicles for pre-planned trips. The leading platform for this in Europe is *BlaBlaCar*, which we will discuss extensively in this thesis. Other notable platforms include *Poparide* in Canada.

Ride-sharing offers numerous benefits over solo driving. Most notably, it significantly reduces pollution. *BlaBlaCar* claims to have saved over a million cubic tons of CO2 emissions (BlaBlaCar, 2024b), while also enhancing road safety and facilitating social interaction among users. Scientific studies support these benefits, showing that ride-sharing improves overall efficiency and has positive social implications (Aiko et al., 2017). Research by Yu et al. (2017) analyzing pollution data from Beijing revealed substantial environmental improvements since the introduction of *DiDi*. Additionally, a study by Kirk, Cavalli, and Brazil (2020) indicated that the launch of *Uber* in the UK led to a 9% decrease in fatal drunk driving accidents.

This thesis is structured as follows: First, we will introduce *BlaBlaCar* and its current matching process. Subsequent sections will explore literature on dynamic matching to provide a comprehensive understanding. We will then examine three important studies in dynamic matching and compare their findings. Building on these insights, we aim to refine the matching process for *BlaBlaCar* using simulation results and literature-based proposals. Finally, we will conclude.

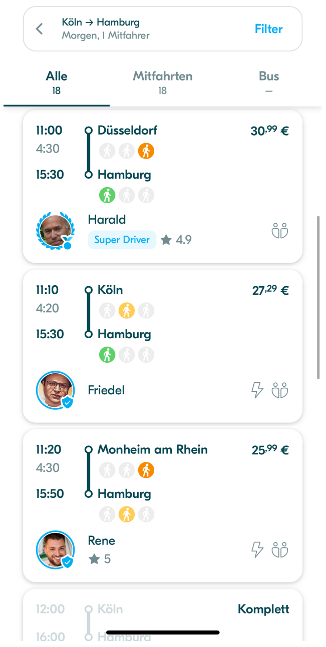
The primary objectives of this thesis are to provide an overview of various dynamic matching models and to analyze *BlaBlaCar*'s matching process, suggesting improvements based on existing research. Additionally, we will identify potential areas for future research.

1.1 Information about BlaBlaCar

BlaBlaCar is a leading platform in the ride-sharing industry, connecting drivers with empty seats to riders traveling in the same direction. As of 2021, *BlaBlaCar* expanded its offerings to include bus seats, which accounted for 20% of their service. The *BlaBlaBus* operations, considered an external option, are not the focus of this thesis. The user base of *BlaBlaCar* is notably diverse: one-third are students, while two-thirds are employed individuals. The gender distribution comprises 60% male and 40% female. Nearly 50% of the users are aged 30 or older, with 40% of individuals aged 18-35 in France registered on *BlaBlaCar*. The percentage of seniors (aged 55+) using the platform has doubled in the past six years. *BlaBlaCar* charges a commission ranging from 18% to 21% on rides facilitated through its platform.

1.2 BlaBlaCar's Matching Algorithm

The current matching algorithm of *BlaBlaCar* allows drivers to post rides, specifying available seats and the option for up to 3 stopovers where riders can join or leave the ride. Drivers can set preferences regarding whether riders are smokers or non-smokers, if they can bring animals, and luggage capacity. They have two options for ride bookings: riders can either book a seat directly or request a ride, giving drivers the opportunity to accept or decline the request. This thesis will focus on the request option, as it allows drivers to cancel trips without affecting their reputation adversely, which is not the case when a rider directly books a seat. Like many ride-sharing companies, e.g. *Uber*, *BlaBlaCar* employs a reputation system where both riders and drivers rate each other on a 5-point scale after the completion of a trip.


Köln → Hamburg
Morgen, 1 Mitfahrer
Filter

Alle 18 | Mitfahrten 18 | Bus –

11:00 Düsseldorf 30.99 €
4:30
15:30 Hamburg
Harald
Super Driver ★ 4.9

11:10 Köln 27.29 €
4:20
15:30 Hamburg
Friedel

11:20 Monheim am Rhein 25.99 €
4:30
15:50 Hamburg
Rene
★ 5

12:00 Köln Komplett
16:00 Hamburg

Figure 1. Typical search results from a rider request on the BlaBlaCar App for the next day for a ride from Cologne (Köln) to Hamburg (7.5.2024).

BlaBlaCar also provides an index for the quality of matches, based on distance and timing, evaluated on a 3-point scale (see Figure 1). From our observations, it appears that this rating system may have some technical issues, which, while noteworthy, do not have significant implications for this study. Regardless of the distance they must travel to the meeting location, riders can book any available trip, and drivers do not have access to information about where riders live or the extent of their travel to the agreed pick up location.

Riders browsing carpool rides are shown the total price they will pay to book the ride, while drivers receive the exact amount they set when offering a ride, as *BlaBlaCar* does not deduct any fees from the drivers. A more formalized discussion about the current situation and potential improvments will follow after presenting the three main models.

2 Literature Review

The rise of ride-sharing companies has spurred a significant body of scientific literature, primarily contributed by three disciplines: economics, computer science, and operations research. This literature review is structured as follows: We first discuss key ideas from economic dynamic matching, then delve into queuing applications from both mechanism design and operations research, and finally, we explore specific studies within the context of ride-sharing.

Dynamic Matching The theory of dynamic matching originated partly from the allocation of kidneys, transitioning from static to dynamic models. In this dynamic framework, patients needing new kidneys either receive a match or become unmatchable over time. One of the earlier works on dynamic kidney matching models is by Ünver (2010), with further advancements by J. P. Dickerson, A. D. Procaccia, and Sandholm (2019) and J. Dickerson, A. Procaccia, and Sandholm (2021). Shimer and Smith (2001) explore dynamic matching in the labor market, where the model fundamentally differs because employees benefit from waiting, potentially securing a better match.

Akbarpour, Li, and Gharan (2020) propose a more general dynamic matching framework and demonstrated that relatively simple mechanisms can achieve impressive results. Subsequent work by Ashlagi, Nikzad, and Strack (2022) examine a theoretical model of dynamic matching and thickness with two types, revealing that for highly heterogeneous types, some results no longer hold. Kanoria (2023) developes solutions for optimal dynamic matching algorithms, which are more complex compared to the simpler ones discussed in this thesis. Another closely related model by Anderson et al. (2014) aim to minimize average waiting time, with agents in the model never perishing.

Queuing Queuing theory has been extensively developed within economics and operations research. A notable contribution by Leshno (2022) investigates settings of overloaded waiting lists where both agents and items arrive stochastically. Similar methodologies are employed by Harrison (2000) and Plambeck and Ward (2006), akin to the approach of Özkan and Ward (2020): defining a queuing problem, deriving a possibly optimal policy, and subsequently proving its optimality. These problems can be *High-Volume Assemble-to-Order Systems* as in Plambeck and Ward (2006), or more general and deal with open processing networks as in Harrison (2000).

Ride Sharing Research on matching markets in the digital age, including ride-sharing, has been done by Azevedo and Weyl (2016), who provides a general overview of the current topic. We can segment ride-sharing into three categories: the first addresses scenarios where drivers travel long distances to pick up riders, the second focuses on surge pricing, and the third primarily examines matching mechanisms.

Castillo, Knoepfle, and Weyl (2024) discuss a market failure known as the Wild Goose Chase (WCG)[1], exploring solutions using surge pricing. Besbes, Castro, and Lobel (2021)

1. We will introduce this problem in more detail in section 3.2.2.

analyze a theoretical spatial environment with the primary goal of maximizing revenue for planners, finding results similar to those of Castillo, Knoepfle, and Weyl (2024). Feng, Kong, and Wang (2021) study the effects of implementing caps in ride-sharing apps, finding nonmonotonicity in waiting times similar to Castillo, Knoepfle, and Weyl (2024), but they focus on denying trips to certain customers as their main mechanism, a policy further discussed in section 3.4.

Despite public criticism, research on surge pricing, such as that by Cachon, Daniels, and Lobel (2017) and Castillo (2023), concludes that it generally increases total welfare, although price-sensitive riders may see a decrease in utility due to higher prices not compensated by reduced waiting times. Li et al. (2019) expand on the model of Castillo, Knoepfle, and Weyl (2024), supporting their findings with data from DiDi.

Other research, such as that by Akbarpour et al. (2021) and Banerjee, Kanoria, and Qian (2022), focuses more on pure matching solutions rather than pricing strategies. Unlike Özkan and Ward (2020), which uses a probabilistic optimal solution, Banerjee, Kanoria, and Qian (2022) considers a deterministic solution within a closed network, assuming a constant number of drivers, which is a strong assumption.

3 Models

This section presents three important models in the context of dynamic matching. We begin by introducing some notations that will be used throughout this section, followed by descriptions of the following models.

Castillo, Knoepfle, and Weyl (2024) primarily employ pricing as a matching mechanism and discuss several simple alternatives to a pure price-based mechanism. On the other hand, Akbarpour, Li, and Gharan (2020) present a pure dynamic matching model, which, although abstracted from the specific context of ride-sharing, can be utilized to analyze aspects of the matching process. This model provides a more theoretical framework for understanding dynamic interactions. Özkan and Ward (2020) apply methods from queuing theory to model ride-sharing within a city. The first part of the paper focuses on a pure matching strategy, while the second part introduces pricing as a potential mechanism to enhance efficiency.

In the concluding part of this section, we will compare these models and discuss their implications and effectiveness in the ride-sharing industry.

3.1 Definitions

Most notation will be introduced at the beginning of each paper. The following notations, however, will be used consistently throughout this thesis: $\mathbb{N}$ represents the set of natural numbers, and $\mathbb{N} \cup \{0\}$ represents the set of non-negative integers.

In all three models, t or T represents time. The specific meanings of t and T will be elaborated in each model. Let λ denotes the arrival rate of riders or agents, as used in Akbarpour, Li, and Gharan (2020). All three model have a planner, e.g. the ridesharing company which aims to maximize the total utility in each model.

Drivers are defined as individuals with empty vehicles, and riders are those who request a ride. Riders are willing to pay a price to be transported from their current location to another. In all models, riders have an external option for travel, such as taking a train. A match occurs when a driver takes a rider to her destination. The most common matching protocol is the First-Dispatch Protocol.

Definition 1 (First-Dispatch Protocol)**.** When a rider requests a trip, she is immediately matched to the nearest idle driver.

Both Özkan and Ward (2020) and Castillo, Knoepfle, and Weyl (2024) use the First-Dispatch Protocol as a benchmark. They evaluate their solutions against this protocol to highlight its inherent problems and to demonstrate the effectiveness of alternative approaches.

3.2 How Prices Can Fix the Matching Problem

Castillo, Knoepfle, and Weyl (2024) focus on ride-hailing and the market failure known as Wild Goose Chases (WGC) in their dynamic model. They theoretically explain this phenomenon and suggest how companies can act to eliminate this failure. Their main finding is that surge pricing can almost entirely prevent this market failures. They support their hypothesis with an empirical analysis using data from *Uber*. Additionally, they discuss other matching mechanisms that could reduce welfare loss due to WGC as market failure.

In this section we first explain the theoretical model from Castillo, Knoepfle, and Weyl (2024), followed by their empirical findings on surge pricing and the discussion of other mechanisms to enhance welfare.

3.2.1 Theoretical Framework

We begin by defining the demand for trips by riders, followed by the supply of drivers, then connecting these two to discuss the market equilibrium and potential failures.

Demand is defined as the number of people requesting a trip at a given pickup time T and price p. The demand function $D(T, p)$ is bounded, differentiable, and decreases with increases in both T and p, reflecting that more riders are interested when prices and pickup times are lower. The willingness-to-wait distribution among riders has a finite mean.

Drivers decide to work based on their expected hourly earnings e. The supply of drivers, denoted as $L = l(e)$, is measured in drivers per unit area, is differentiable, and increases with e. For $e = 0$, the supply is zero. The equilibrium number of trips within a fixed timeframe is Q, and τ represents the platform's commission rate. Therefore, the earnings for each driver are calculated as:

$$e = (1 - \tau)p\frac{Q}{L}$$

To derive the supply of trips, the authors categorize working drivers into three states: *idle* (waiting to be matched), *en route* (traveling to pick up a rider), and *driving* (transporting a rider to their destination). Consequently, the total number of drivers in a steady state is the sum of these states:

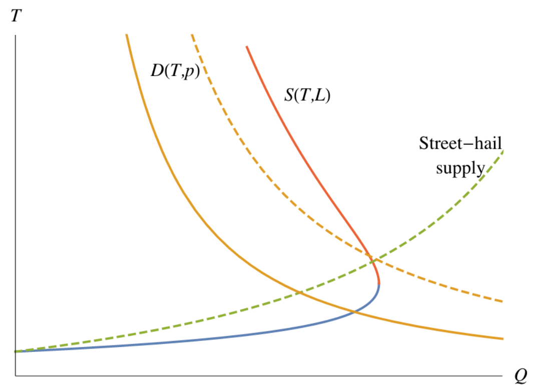
$$L = \underbrace{I}_{\text{Idle}} + \underbrace{tQ}_{\text{Driving}} + \underbrace{TQ}_{\text{En route}}$$

where t represents the average duration of a trip and T the average pickup time. This equation captures critical aspects of ride-hailing markets, where both drivers and riders are impacted by large values of T. The average pickup time, modeled as $T(I)$, decreases with an increase in the number of idle drivers. Castillo, Knoepfle, and Weyl (2024) assume $T(I)$ to be continuously differentiable and convexly decreasing from $\lim_{I \to 0} = \infty$ to $\lim_{I \to \infty} = 0$. The inverse function, $I(T)$, indicates the number of drivers required to achieve a certain pickup time T. The trip supply function is then defined as:

$$S(T, L) = \frac{L - I(T)}{t + T} = \frac{\text{Number of busy drivers}}{\text{Average busy time per trip}}$$

3.2.2 The Wild Goose Chase Phenomenon

To analyze market equilibrium within their framework, Castillo, Knoepfle, and Weyl (2024) maintain a fixed price and utilize the First-Dispatch-Matching algorithm, as employed by *Uber* at this time. Thus, the clearing of the market is determined not by price but by pickup times, implying that T effectively functions as the price variable, while p remains exogenous. The trip supply function $S(T, L)$ is continuous in T, and the number of drivers L acts as a supply shifter. For a graphical representation of this setting, see Figure 2.



Note: Figure 2 from Castillo, Knoepfle, and Weyl (2024). Note that the green, dashed line represents street-hail taxis, which are not discussed in this thesis.

Figure 2. Trip supply and demand with a fixed number of drivers.

The shape of the supply curve $S(T, L)$ is particularly interesting. To achieve a smaller T, indicating shorter pickup times, a planner needs more idle drivers. However, this increases the idle time, reducing the number of busy drivers and consequently the number of trips supplied. This dynamic is the primary force shaping the blue region in the graph.

In the red region, the denominator of the function becomes more significant. As it increases, drivers spend more time traveling to distant riders, leading to higher values of

T. This increase in the average time for trips results in a lower trip supply by exceeding its capacity.

The nonmonotonic nature of $S(T, L)$ implies that two different points can provide the same number of trips. Castillo, Knoepfle, and Weyl (2024) label the point in the blue as a 'good outcome' and the point in the red section, which involves longer pickup times, as the 'bad outcome'. The 'bad outcome' scenario is defined as a WCG:

Definition 2 (Wild Goose Chase)**.** A market is in a *Wild Goose Chase* if it exists in the decreasing region of the trip supply function.

Diagnosing Wild Goose Chases Castillo, Knoepfle, and Weyl (2024) provide an intuitive method for planners to determine if a WGC is occurring. The key diagnostic parameter is the elasticity of pickup time with respect to the number of idle drivers, defined as:

$$\varepsilon_I^T := \frac{\delta T(I)}{\delta I}\frac{I}{T}.$$

The elasticity, ε_I^T, indicates the responsiveness of pickup time T to changes in the number of idle drivers I. Using data from *Uber*, they estimate this elasticity to be between -0.25 and -0.45, indicating that an increase in idle drivers leads to a decrease in pickup times. The condition for identifying a WGC is expressed as follows:

$$s := \frac{I}{TQ} < \varepsilon_I^T$$

where, s represents *slack* in the system. Low slack indicates that there are relatively few idle drivers compared to those en route, leading to longer pickup times as drivers must travel greater distances to reach passengers. Therefore, the inequality represents the threshold at which the system falls into a WGC, where increasing demand exacerbates inefficiencies rather than improving the number of trips.

The intution behind this threshold the following: one can take the derivative of the trip supply function $S(T, L)$ with respect to T. When this derivative is negative, a decrease in the trip supply corresponds to an increase in pickup time T, indicating the presence of a WGC.

Equilibrium Following the classic microeconomic demand-supply model, Castillo, Knoepfle, and Weyl (2024) demonstrate that a stable equilibrium occurs at the point where demand equals supply. Furthermore we have an addional constrain on the number of working drivers. This relationship is represented by the following equations:

$$\begin{aligned} Q &= D(T, p) = S(T, L) \\ L &= l\left((1 - \tau)\, p \frac{Q}{L}\right) \end{aligned}$$

To determine the equilibrium, one must simultaneously solve for T, Q, and L. The first equation provides a function $Q(L)$, and the second equation $L(Q)$, allowing us to solve for their intersection. An equilibrium is considered stable if both functions independently converge to this point. Castillo, Knoepfle, and Weyl (2024) establish that at least one equilibrium always exists, but note that there a functions for which multiple equilibria exist, although none of their calibrations show this kind of behavior. In scenarios with multiple equilibria, they establish the following theorem:

Theorem 3. *A stable equilibrium always exists, possibly with $Q = L = 0$. If multiple equilibria are present, they can be ordered by either Q or L. The highest equilibrium is always stable.*

Revenue, Surplus, and Welfare We now examine the revenue of the platform, the surplus of drivers and riders, and the total welfare at an equilibrium price p and quantity Q. The revenue for the platform is calculated as τpQ. The driver surplus (DS) is defined as:

$$DS(Q, L, p) = \underbrace{(1-\tau)pQ}_{\text{Earnings}} - \underbrace{C(L)}_{\text{Costs}}$$

where $C(L) = \int_0^L l^{-1}(x)\,\mathrm{d}x$ represents the cost function for drivers, which is increasing and convex. To define the rider surplus (RS) and total welfare (W), Castillo, Knoepfle, and Weyl (2024) first introduce gross utility (U):

Definition 4 (Gross Utility)**.** Gross utility $U(Q, T)$ is continuously differentiable in (Q, T). It is decreasing in T and increasing in Q with $U_Q(Q, T) = p$.

Therefore, the RS is defined as follows[2]: $RS(Q, T) = U(Q, T) - pQ$ and the total welfare W is given by: $W(Q, L, T) = U(Q, T) - C(L)$. It is important to note that the total welfare W is the sum of the driver surplus, rider surplus, and platform revenue, expressed as $W = DS + RS + \text{revenue}$.

Pricing Strategies In this part, we examine how equilibrium adjustments occur when, ceteris paribus, the prices change. Castillo, Knoepfle, and Weyl (2024) consider a scenario with a fixed commission τ where the planner can only adjust the price p, in line with *Uber*'s pricing policy. When the market is at a good equilibrium, no action is required by the planner. However, in a WGC situation, the planner can either increase or decrease p. We first explore the implications of a price decrease and then discuss the effects of a price increase.

When the planner reduces the price from p to $p' < p$, this induces a negative feedback loop. Initially, a lower price boosts demand, shifting the demand curve to the upper right as depicted in Figure 2, resulting in fewer matchings and increased pickup times. At the same time, the decrease in price leads to reduced earnings, prompting fewer drivers to enter the market. This reduction in driver supply shifts the supply curve to the left, exacerbating the decrease in matchings and further elevating pickup times. This cycle continues until a new equilibrium is reached, characterized by significantly higher T and reduced matchings. Thus,

2. Note that the waiting cost for riders is assumed to be 0.

Castillo, Knoepfle, and Weyl (2024) conclude that a decrease in price leads to longer pickup times, fewer trips, reduced driver participation, lower driver surplus, decreased revenue, and diminished total welfare. The impact on rider RS is mixed; while lower prices may initially benefit riders, increased waiting times, if modeled, would adversely affect RS.

Conversely, an increase in price p tends to suppress demand, which naturally diminishes as the price approaches infinity. Such an increase can potentially lead out of a WGC situation, as formalized in the following theorem:

Theorem 5. *Suppose that the highest equilibrium of the market at price p is in a WGC. Then there exists a higher price $p' > p$ at which the highest equilibrium is no longer in a WGC.*

3.2.3 Empirical Analysis

Castillo, Knoepfle, and Weyl (2024) analyze a dataset provided by *Uber* covering the Manhattan region from December 1, 2016, to May 14, 2017. This analysis focuses solely on *UberX* data, as other service types employ slightly different matching mechanisms. During this period, *Uber* predominantly utilized a first-dispatch matching algorithm, with price adjustments aimed at preventing WGCs. The empirical evidence lends support to their theoretical model regarding the occurrence and impact of WGCs in ride-hailing markets.

The first key finding from their analysis is the nonmonotonic nature of the trip supply curve. Using the number of completed trips as the dependent variable, and estimated time of arrival (ETA) along with other controls as explanatory variables, the shape of the estimated trip supply curve closely resembles that posited in their theoretical framework. Both Ordinary Least Squares and Two-Stage Least Squares (with the number of riders opening the app as the instrumental variable), the estimations of the trip supply curves are very close to the shape of the one in their theoretical framework. This is the first indication that WGC is happening in the data.

The second main result confirms the occurrence of WGCs, identified using the condition $s < \varepsilon_I^T$. Analysis reveals that WGCs frequently occur during rush hours and near locations with high rider arrival rates. Despite *Uber*'s implementation of pricing strategies to mitigate WGCs, approximately 4.34% of time periods and 6.41% of trips still fall within the WGC category. Furthermore, rider cancellations and average pickup times increase significantly during these periods, leading to a decline in service quality.

3.2.4 Surge Pricing

From Theorem 5, we already know that an increase in price can effectively prevent WGCs. As noted in section 3.2.3, *Uber* employs a pricing mechanism known as surge pricing to counteract WGCs. This dynamic pricing strategy adjusts prices based on real-time demand and supply conditions. We will first outline necessary model adaptations, then define surge pricing, and finally compare it to uniform pricing.

Model Adaptation Castillo, Knoepfle, and Weyl (2024) redefine the demand curive by introducing λ, the arrival rate of potential riders, and $r(p)$, the fraction of riders willing to pay a higher price. They also include $g(T)$, representing the fraction of riders willing to wait for time T. The authors assume independence between the willingness to pay and wait, resulting in the demand function $D(T, p) = \lambda g(T) r(p)$. While this is a strong assumption, Castillo, Knoepfle, and Weyl (2024) provide examples for both negative and positive correlations and point to Buchholz (2021), who found a slightly positive correlation.

The labor supply function is defined as:

$$l(e) = A \left(\frac{e}{1 + \frac{1}{\varepsilon_l}} \right)^{\varepsilon_l}$$

where ε_l is the constant-elasticity of labor supply, and A acts as a supply shifter.

Utility is calculated using:

$$U(p, T) = \lambda g(T) \left[\underbrace{\int_p^\infty r(p') \mathrm{d}p'}_{\text{Riders utility}} + \underbrace{pr(p)}_{\text{Drivers utility}} \right].$$

where λ is a demand shifter.

For the next part of the analysis, Castillo, Knoepfle, and Weyl (2024) use estimates for all of the aforementioned parameters to quantify the utility and welfare losses in US Dollars. They differentiate between a strong and a weak market.

Definition 6 (Uniform Pricing)**.** Uniform pricing is a single, constant price for rides regardless of the time of day, demand, and supply.

Definition 7 (Surge Pricing)**.** Surge pricing involves dynamically adjusting prices based on real-time demand and supply conditions. Prices increase during a strong market and decrease during a weak market.

Castillo, Knoepfle, and Weyl (2024) calculate the optimal uniform price and surge prices for both a strong and a weak market, aiming to maximize total welfare. While the high surge price is marginally higher than the uniform price in strong markets, it is significantly lower in weak markets. They note that the optimal price in weak markets occasionally falls below *Uber*'s standard pricing, and *Uber* does not implement negative surge multipliers. According to Castillo, Knoepfle, and Weyl (2024), this results in a minor welfare loss of approximately 3.5%.

3.2.5 Alternative Matching Mechanisms

Given the reputational drawbacks associated with surge pricing (The Economist (2016), Ariely (2016), Goncharova (2017)), there is room for exploring other mechanisms that could effectively avoid WGCs without using surge pricing. Public perception often associate surge pricing with price discrimination. Despite the negative reputation of surge pricing companies like *Uber*, *Lyft*, and *DiDi*, use it in practice.

Castillo, Knoepfle, and Weyl (2024) present three alternative mechanisms to mitigate WGCs, though they find none as efficient as surge pricing. All three alternatives are based on the Maximum Dispatch Radius (MDR) parameter, R, and tend toward the first-dispatch protocol as $R \to \infty$. Extending the model to include these mechanisms could complicate the matching process, which is currently relatively simple.

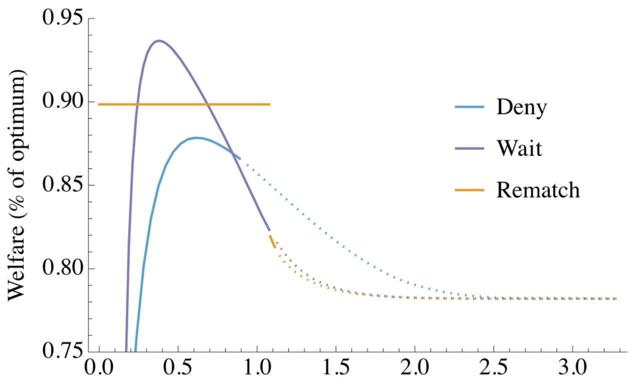
Definition 8 (Wait)**.** After a rider requests a trip, she is matched to the nearest driver within a radius R. If no driver is available within this radius, the rider waits until a driver enters the designated radius.

Definition 9 (Deny)**.** After a rider requests a trip, she is matched to the nearest driver within R. If no driver is within this radius, the rider is denied a trip.

Definition 10 (Rematch)**.** Riders are initially matched to the nearest available driver. If a closer driver becomes available within R after the initial match, the rider is rematched to this nearer driver.

Castillo, Knoepfle, and Weyl (2024) note that, in practice, the difference between Wait and Deny is minimal because riders can re-request a trip at no additional cost.

Empirical Analysis of the Alternative Matching Mechanisms For the further analysis, we assume that the market is a WGC. For the empirical results for New York, Castillo, Knoepfle, and Weyl (2024) demonstrate that for large, not optimal values of R (1.5 km for Wait and Rematch, and 2.5 km for Deny), the welfare achieved is approximately 78% of the optimal level, as illustrated in Figure 3.



Note: Figure 11 from Castillo, Knoepfle, and Weyl (2024). It shows the welfare of the three different matching algorithms.

Figure 3. Welfare as a function of the MDR for the three matching protocols

Despite the potential welfare improvements with alternative matching mechanisms to address WGCs, there are significant customer service concerns. Although the average wait time to resolve a WGC is about 9 to 35 seconds in the optimal setting, the uncertainty of the wait time may prompt customers to switch to competitors. This uncertainty undermines customer satisfaction, making these mechanisms less consumer-friendly. Specifically, under the Deny mechanism, between 6%-10% of ride requests are denied, potentially harming the reputation of ride-sharing platforms as reliable alternatives to traditional taxis. For Rematch, while the fraction of rematches under a large R is relatively small, it increases sharply as R decreases. This variability can confuse riders about which driver will pick them up, decreasing their confidence in the platform.

Castillo, Knoepfle, and Weyl (2024) empirically compare these three mechanisms against surge pricing in New York. They discover that even with imperfect surge pricing, welfare reaches 97.3% of the optimal, followed by Wait at 95.9%, Deny at 92.3%, and Rematch at 91.0%, when using the optimal R. Doing nothing yields only 86.9% of potential welfare. The authors conclude that surge pricing generally surpasses other mechanisms in effectiveness, except in mild WGC scenarios where Wait and Rematch can provide slightly higher welfare.

3.2.6 Discussion and Implications

Nowadays, companies like *Uber* and *Lyft* use a combination of surge pricing and other methods to enhance service efficiency and mitigate WGCs. *Uber*, for instance, aggregates all trip requests for several seconds and applies an algorithm designed to avoid WGCs (Yan et al., 2020), significantly improving upon the pure first dispatch protocol. Additionally, *Uber* incorporates rematching to boost the overall efficiency of its matching process. Castillo, Knoepfle, and Weyl (2024) demonstrate both theoretically and empirically that while *Uber* has successfully averted the majority of welfare losses associated with WGCs, there remains potential for further welfare enhancements of up to $1.87 million per week. This suggests that there is still room for improvement in their algorithms to fully optimize welfare outcomes.

3.3 A More General Dynamic Matching Model

The theoretical model of Akbarpour, Li, and Gharan (2020) is a dynamic matching model where agents arrive and depart stochastically. They apply techniques from random graph theory, where the structure of the networks depends on the matching algorithm implemented. The main challenge of this model is to find the optimal solution of the underlying Markov decision problem, which is computationally infeasible, for even morderate marekt sizes. Therefore alternatives algorithms are needed. Akbarpour, Li, and Gharan (2020) show that two algorithms, one where the planner has information about the agents' departure and one where the planner does not, are close to optimal in the corresponding envirmonets. Additionally, they explore ways in which the planner might extract information about the agents' departure times.

The main difference between the model of Akbarpour, Li, and Gharan (2020) and Castillo, Knoepfle, and Weyl (2024) is that the former is much more general and can be applied to various settings. Nonetheless, the results from Akbarpour, Li, and Gharan (2020) are useful for discussing optimal algorithms in ride-sharing. The model initially considers just one type of agent; however, subsequent work for example by Liu, Wan, and Yang (2024) has extended it to include two types, yielding similar results. In the context of ride-sharing, this model can be thought of as analogous to agents pooling for a ride, a service offered by many companies, such as *Uber Pool*, *Lyft Line*, and most notably for this thesis, *BlaBlaCar*. Throughout this section, we will refer to riders as agents, following the terminology used by Akbarpour, Li, and Gharan (2020).

3.3.1 Theoretical Framework

Akbarpour, Li, and Gharan (2020) develop their model within a continuous-time framework on a stochastic network, leveraging techniques from Erdős and Rényi (1960). This model operates over the interval $[0, T]$ and focuses on the dynamics of matching markets.

Model Overview Agents arrive into the system at a Poisson rate of λ ($\lambda \geq 1$), indicating that in every interval $[t, t+1]$, approximately λ new agents enter the market. The pool of agents at time t is denoted by A_t, and the pool size is $Z_t := |A_t|$. The pool of agents A_t can be described as a function of $t \in [0, T]$, where $A_0 = \emptyset$. We are interested in the limit behavior of A_t, under the differnt algorithms. We call the set of agents who enter at time t A_t^n, hence $\left|A_t^n\right| \leq 1$. Let A_{t_0,t_1}^n denotes the set of agents who enter in the interval $[t_0, t_1]$.

Each agent is subject to a Poisson process at rate $\theta = 1$, determining the point at which she becomes critical. An agent entering the market at t_0 becomes critical at time $t_0 + X$, where X is an exponential random variable with mean 1. This is the last moment she can get matched, either due to being matched or leaving unmatched (perishing). The moment an agent leaves the market is denoted as t_1, thus $t_0 \leq t_1 \leq t_0 + X$. We call the length of the interval that a is in the pool the *sojourn* of a: $s(a) := t_1 - t_0$. A_t^c is the set of agents who are critical at time t, and thus the following holds: $\left|A_t^c\right| \leq 1$.

Network Structure In their analysis, Akbarpour, Li, and Gharan (2020) use the classic Erdős-Rényi random graph model. This paragraph explains their model adaptation. Each agent in the model is represented as a node, and any two nodes are compatible (connected by an edge) with a probability $p := \frac{d}{\lambda}$, where d represents the average degree of the graph (number of edges per node). For $t \geq 0$, the set of edges at time t is denoted as $E_t \subset A_t \times A_t$, and $G_t = (A_t, E_t)$ represents the network. The subset $N_t(a) \subset A_t$ contains all agents compatible with agent a. Compatible pairs remain so over time.

In the long term, G_t follows an Erdős-Rényi random graph with parameter $\frac{d}{\lambda}$ and maintains an average degree d. We will call $M_t \subset E_t$ a matching if and only if no two edges in M_t share the same node, ensuring that each agent is matched with at most one other agent at a time. A matching algorithm selects a possibly empty M_t in the current G_t. The agents (nodes) of M_t leave the market immediately. Furthermore, we assume that the matching algorithm at time t_0 only knows G_t for $t \leq t_0$ and has no information for $t' > t_0$. Note that all $A_t, E_t, N_t(a)$, and Z_t are functions of the matching algorithm.

Optimization Goal Akbarpour, Li, and Gharan (2020) assume that the waiting cost for all agents is zero. The planner's objective is to maximize social welfare by matching as many agents as possible. The two main constraints for the objective are: first, not all agents are present simultaneously, and second there is uncertainty regarding future arrivals and departures. The planner wants to choose a matching algorithm (ALG(T) from now on).

$$\mathrm{ALG}(T) := \{a \in A : a \text{ is matched by ALG by time } T\}$$

An agent gets utility if she is matched, thus the planner wants to match as many agents as possible. Let ALG(T) denote the set of matched agents at time T. The loss of matching algorithm ALG(T) is defined as follows:

$$\mathbf{L}(\mathrm{ALG}) := \frac{\mathbb{E}\left[|A - \mathrm{ALG}(T) - A_T|\right]}{\mathbb{E}\left[|A|\right]} = \frac{\mathbb{E}\left[|A - \mathrm{ALG}(T) - A_T|\right]}{\lambda T}$$
$$= \frac{\text{Expected number of perished agents}}{\text{Expected number of agents}}$$

Since minimizing the loss is the same objective as maximizing the number of matched agents and it is easier to compute boundaries for the loss, the number of matched agents and the loss are treated equally for the remainder of this thesis.

The solution to this optimization problem can be computed using a Markov decision problem. The state space $\mathscr{S}$ of the Markov decision problem is defined by the pair (H, B). Here, H represents the undirected graph, and B represents the set of critical agents or nodes. Due to the underlying Poisson process, this set has at most one element. The action space, e.g. what the planner can do given the graph H, is the set of possible matchings. The structure of such a graph makes it extremely complicated to calculate the exact Markov decision problem and can be computed only for a very small number of agents. Thus, the remainder of the paper from Akbarpour, Li, and Gharan (2020) employs techniques from graph theory to calculate a theoretical upper bound for the loss and algorithms that are simple to compute.

Optimal Matching Boundaries The theoretical upper bound for the number of matchings in a Markov decision problem is can be calculated using the Omniscient algorithm (OMN). It has full knowledge about future events and the entire realization of the graph. This algorithm achieves the maximum number of possible matchings. The loss associated with the OMN is defined as:

$$\mathbf{L}(\mathrm{OMN}) := \frac{\mathbb{E}\left[|A - \mathrm{OMN}(T) - A_T|\right]}{\mathbb{E}\left[|A|\right]} = \frac{\mathbb{E}\left[|A - \mathrm{OMN}(T) - A_T|\right]}{\lambda T}$$

Given that OMN has foreknowledge of the stochastic process, any other algorithm ALG(T) will necessarily match fewer or an equal number of agents: $|\mathrm{ALG}(T)| \leq |\mathrm{OMN}(T)|$. The optimal algorithm that has access to the set of critical agents at time t is denoted as $\mathrm{OPT}^{c}(T)$ and OPT(T) represents the optimal algorithm without access to critical agent information.

Let $\mathrm{ALG}^{c}(T)$ be any an algorithm that has access to critical agents, and ALG(T) which does not. Therefore, the relationships among their loss functions are:

$$\mathbf{L}(\mathrm{ALG}) \geq \mathbf{L}(\mathrm{OPT}) \geq \mathbf{L}(\mathrm{OPT}^{c}) \geq \mathbf{L}(\mathrm{OMN})$$

and

$$\mathbf{L}(\mathrm{ALG}^{\mathrm{c}}) \geq \mathbf{L}(\mathrm{OPT}^{\mathrm{c}}) \geq \mathbf{L}(\mathrm{OMN}).$$

It is important to note that the loss functions are based on the expected number of matched agents, which in turn depends on the realization of G. For some realization of G the number of matched agents, e.g. the loss, might be the other way around.

3.3.2 Algorithm Analysis

This section presents the properties of the OPT$^{\mathrm{c}}$ algorithm and introduces two simpler algorithms, highlighting the differences between them.

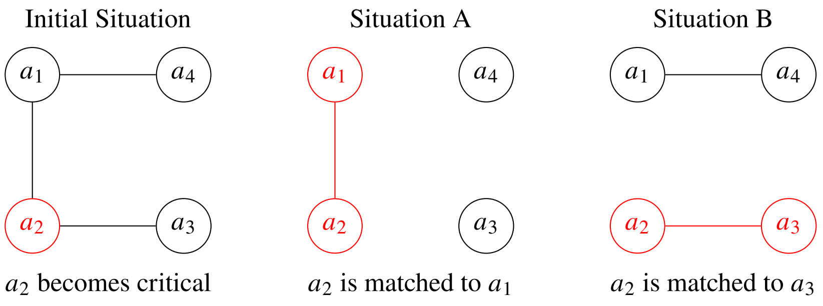
The OPT$^{\mathrm{c}}$ algorithm, which has access to the set of critical agents at time t, is characterized by two essential properties as outlined by Akbarpour, Li, and Gharan (2020):

(1) A pair of agents a, b will be matched by OPT$^{\mathrm{c}}$ if and only if one of them is critical.

(2) If an agent a becomes critical at time t and $N_t(a) \neq \emptyset$, then OPT$^{\mathrm{c}}$ will match a.

Therefore, OPT$^{\mathrm{c}}$ waits until an agent a becomes critical and then selects a partner from $N_t(a)$, if $N_t(a) \neq \emptyset$. The challenge lies in deciding which partner to choose, which depends on the network structure, as illustrated in Figure 4. In this case, it is optimal to match a_2 to a_3 because this allows for a potential match between a_1 and a_4.

To conclude, OPT$^{\mathrm{c}}$ employs two strategies to minimize loss: first, it uses timing to thicken the market, and second, it optimizes over network structure to facilitate optimal matches. While the first strategy is straightforward to implement, the second strategy involves combinatorial complexity. Akbarpour, Li, and Gharan (2020) tries to separate these effects by introducing two new algorithms.



Note: Based on Fig. 2 in Akbarpour, Li, and Gharan (2020). It is always better to match a_2 to a_3 than to a_4 in this example since then a possible match between a_1 and a_4 is preserved.

Figure 4. Illustration of the Effects of a Matching on the Network Structure

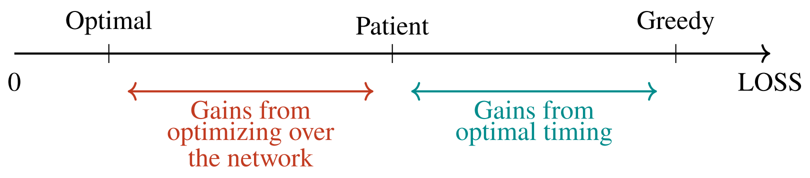
The first algorithm is a simple "match as you go": As soon as a feasible match is observable, the planner matches these two agents.

Definition 11 (Greedy Algorithm)**.** If a new agent a enters the market at time t, they are matched with an arbitrary agent in $N_t(a)$ if $N_t(a) \neq \emptyset$.

Since no more than one new agent arrives at any given time, the Greedy algorithm tends to match agents immediately upon their appearance, resulting in G_t being almost always empty. This approach does not use any information about the set of critical agents. The other algorithm exploits the timing component of the OPT^c but does not optimize over the network:

Definition 12 (Patient Algorithm)**.** If an agent a becomes critical at time t, this algorithm matches them uniformly at random with another agent from $N_t(a)$ if $N_t(a) \neq \emptyset$.

Thus the difference between the Patient algorithm and the Greedy algorithm captures all the gains from the optimal timing, while the difference between the Patient algorithm and the Optimal algorithm captures the gains from optimizing over the network structure. The different effects on the loss are also illustrated in Figure 5. Therefore $\mathbf{L}(\text{Greedy}) \geq \mathbf{L}(\text{Patient})$.


Optimal
Patient
Greedy
0
LOSS
Gains from optimizing over the network
Gains from optimal timing

Note: Based on Akbarpour (2017a). The "true" difference depends on the values of m and λ. The figure illustrates that the two effects can be strictly separated.

Figure 5. Illustration of Gains of the Different Algorithms

3.3.3 Algorithmic Performance and Market Thickness

In this section, we discuss the performance of the differnt dynamic matching algorithms by analyzing the limiting behavior of random graphs. Akbarpour, Li, and Gharan (2020) follow the standard approach for large but sparse graphs established by Erdős and Rényi (1960). In their model, this means that we study the outcomes of the policies when $\lambda \to \infty$. To eliminate nuisance terms, d is held constant, and T goes to infinity at a faster rate than $\log(\lambda)$. This implies $p = \frac{d}{\lambda} \to 0$, suggesting almost no connections between nodes. Since this is the standard way of presenting results from random graphs, this should not be taken seriously in the sense that there are no matches, but rather that when the graph becomes larger, it does not become denser.

From Section 3.3.2, we understand that the Patient algorithm outperforms the Greedy algorithm. Theorem 13 quantifies this difference, indicating that the loss for the Greedy algorithm decreases linearly with d, while for the Patient algorithm, the loss decreases exponentially with d.

Theorem 13. *For $d \geq 2$, as $T, \lambda \to \infty$,*

$$\begin{aligned} \mathbf{L}(\text{Greedy}) &\geq \frac{1}{2d+1}, \\ \mathbf{L}(\text{Patient}) &\leq \frac{1}{2} \cdot e^{\frac{-d}{2}}. \end{aligned}$$

The intuition behind theorem 13 is as follows[3]: The Greedy algorithm matches as early as possible, thus the graph is almost always empty (has no edges) and in comparison to the Patient algorithm has a lower order. If a new agent enters the market in which there are z agents, the probability that the new agent has no matches is $\left[1-\left(\frac{d}{\lambda}\right)\right]^{z}$. One can see that this term falls exponentially in d. At the same time, as d rises, the Greedy algorithm matches more rapidly and the equilibrium stock of agents z is smaller[4]. The combination of these two effects cancels the exponential loss out, yielding to a linear relationship.

On the other hand, the Patient algorithm waits to match agents until they become critical. Thus it can exploit the exponential effect when a new agent joins the market, resulting in the loss falling exponentially in d.

From Figure 4, we already know that the Patient algorithm is not optimal since it does not account for the network structure. Akbarpour, Li, and Gharan (2020) show that the gains from optimizing over the network structure are minor compared to those from being patient. Theorem 14 provides a precise lower bound for the loss of ($\mathrm{OPT^c}$).

Theorem 14. *For $d \geq 2$, as $T, \lambda \to \infty$,*

$$\mathbf{L}(\mathrm{OPT^c}) \geq \frac{e^{-\left(\frac{d}{2}\right)(1+\mathbf{L}(\mathrm{ALG}))}}{d+1}$$

Substituting $\mathbf{L}(\text{Patient})$ for $\mathbf{L}(\mathrm{ALG})$ we get that $\mathbf{L}(\mathrm{OPT^c}) \geq \frac{e^{-d/2\left(\frac{1+e^{-d/2}}{2}\right)}}{d+1}$. This is even for moderate values of d close to the performance of $\mathbf{L}(\text{Patient})$.

This comparative approach highlights that even the most sophisticated algorithm, which can identify and prioritize critical agents, cannot substantially outperform simpler algorithms beyond an exponential factor related to the market thickness parameter d. By incorporating $\mathbf{L}(\mathrm{ALG})$, the theorem emphasizes that the stochastic nature of agent arrivals and departures imposes a fundamental limit on performance improvements. As a result, the potential gains from refining matching decisions are naturally bounded by the inherent dynamics of the market, reinforcing the robustness and practical effectiveness of simpler, local algorithms such as the Patient algorithm.

We can conclude that the gains from optimizing over the network are somewhat small in comparison to the gains of being patient. However, subsequent research expanding on Akbarpour, Li, and Gharan (2020) indicates that when there is more than one type of agent, it becomes more important to deal with the question “whom to match".

3.3.4 Impact of Information Availability

This section considers the scenario where the planner does not have precise short-term information about the set of critical agents. Akbarpour, Li, and Gharan (2020) discusses

3. Some intuition for the proof of theorem 13 can be found in Appendix A.1 for the interested reader.
4. Remember that the number of agents in G is endogenous and depends on the matching algorithm.

how this changes the performance when the planner has more information and when she has less information. They show that information about critical agents and market thickness are complements and propose a mechanism to extract the information of critical agents when it is not known to the planner.

Less Information Assuming that the planner lacks information about the departure times of agents, the OPT algorithm becomes optimal within this context. Akbarpour, Li, and Gharan (2020) establish that the performance of the OPT algorithm is bounded as follows:

$$\frac{1}{2d+1} \leq \mathbf{L}(\mathrm{OPT}).$$

While the OPT algorithm could potentially wait to thicken the market, the findings indicate that such a strategy does not yield substantial gains, and its performance in terms of loss is comparable to that of the Greedy algorithm (both approximately $\frac{1}{2d}$). These results are stated in Theorem 15.

Theorem 15. *For $d \geq 2$, as $T, \lambda \to \infty$,*

$$\frac{1}{2d+1} \leq \mathbf{L}(\mathrm{OPT}) \leq \mathbf{L}(\mathrm{Greedy}) \leq \frac{\log(2)}{d}$$

More information Recall that the OMN algorithm can use information about the future arrival and departure of agents. Under these (very strong) assumptions, one can show that the lower bound of the OMN algorithm is the following:

$$\frac{e^{-d}}{d+1} \leq \mathbf{L}(\mathrm{OMN})$$

In comparison, both the Patient and OPT^{c} algorithms, the OMN algorithm does not perform much better. All three have an exponential loss in d. Akbarpour, Li, and Gharan (2020) get the following theorem:

Theorem 16. *For $d \geq 2$, as $T, \lambda \to \infty$,*

$$\frac{e^{-d}}{d+1} \leq \mathbf{L}(\mathrm{OMN}) \leq \mathbf{L}(\mathrm{Patient}) \leq \frac{1}{2} \cdot e^{\frac{-d}{2}}$$

Information and Market Thickness as Complements From the two modifications of the model, we can conclude three main results. First, having knowledge about which agents are critical at any given time and market thickness are complements. Second, without information about the departure times of agents, no matching algorithm can achieve an exponentially small loss. Third, and most importantly, the Patient algorithm, which is not computationally intensive, yields very good results. Additional information about the departure times of agents provides only marginal gains.

Mechanisms for Information Extraction Recognizing the importance of departure information for achieving optimal results, Akbarpour, Li, and Gharan (2020) change their algorithm to incentivize agents to submit their departure time, e.g., when they become critical truthfully.

To do that they introduce a utility function for agents. If an agent departs the market without being matched, their utility is zero. For those who are matched, utility now depends on the duration of their stay in the market, or their *sojourn* $s(a)$, with utility discounted at a rate r:

$$u(a) \ := \begin{cases} e^{-rs(a)} & \text{if } a \text{ is matched,} \\ 0 & \text{otherwise.} \end{cases}$$

Moreover, agents know when they become critical but cannot see the network's state at time t. The planner, on the other hand, has insight into G_t, the current state of the network, but lacks information on when agents become critical. The proposed mechanism by Akbarpour, Li, and Gharan (2020) is the following.

Definition 17 (Adjusted Patient Algorithm)**.** Ask agents to report when they become critical. When an agent reports being critical, the market maker attempts to match her with a random neighbor. If the agent has no neighbors, the market maker treats her as if she is perished; that is, she is considered permanently unmatchable.

Agents have a function $c_a(\cdot)$ over the interval $[t, t + dt]$. When not critical, an agent reports being critical at a rate $c_a(t)\mathrm{d}t$. This implies that at every infinitesimal interval, the agent is able to report that she is critical. If an agent truly becomes critical, she reports immediately. Under these assumptions, Akbarpour, Li, and Gharan (2020) present the following theorem:

Theorem 18. *Suppose the market is in the stationary distribution, $d \geq 2$, and $d = \text{polylog}(m)$. If $0 \leq r \leq e^{-d/2}$, then the truthful strategy profile is an ϵ-Nash equilibrium for the adjusted Patient algorithm, where $\epsilon \longrightarrow 0$ as $m \longrightarrow \infty$.*

The first condition of Theorem 18 is that the potential matchings (edges) per agent grow relatively slowly compared to the market size (nodes), e.g. the graph is still a sparse graph. The condition on the discount rate r indicates that agents are not overly impatient. The ϵ-Nash equilibrium result implies that agents will tolerate minor inefficiencies in their strategy choices, allowing for slight deviations from perfect optimality that are smaller than ϵ. The logic behind this result is that the adjusted Patient algorithm penalizes agents who misreport their critical status. As long as an agent remains in the market, there is a positive chance she will be matched with a neighbor who becomes critical. If she believes this probability exceeds the cost of impatience, she will wait until truly critical. Reporting critical status prematurely when having no neighbors results in a utility of zero. According to Akbarpour, Li, and Gharan (2020), alternative, softer punishment mechanisms might involve matching a critical agent with another who has not falsely reported critical status, but implementing such mechanisms complicates the analysis due to deviations from the Erdős-Rényi graph model.

3.3.5 Discussion and Implications

To conclude, Akbarpour, Li, and Gharan (2020) find that information and market thickness are complements and, under mild assumptions, can lead to large gains in welfare. The information about departure times is highly valuable if and only if there are no substantial waiting costs, otherwise, agents might not truthfully report their actual departure times. While the simpler algorithms proposed by Akbarpour, Li, and Gharan (2020) do not fully leverage network structure, they perform relatively close to the optimum.

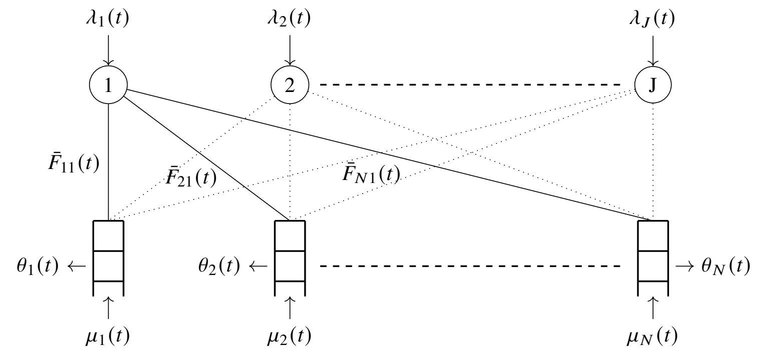
Akbarpour, Li, and Gharan (2020) also note that their findings are less robust in markets with small values of $d < 4$. In such scenarios, they recommend that planners should more carefully consider network structure. Specifically, when an agent a_i becomes critical at time t, the planner should choose another agent $a_j \in N_t(a_i)$ such that $\left|N_t(a_j)\right| \leq |N_t(a_m)|$ for all $a_m \in N_t(a_i)$. Implementing this strategy leads to significant performance improvements in markets characterized by a low degree of connectivity.

Another important aspect discussed by Akbarpour, Li, and Gharan (2020) is their assumption of homogeneous agents. They acknowledge the challenges of incorporating even two distinct agent types into their model and provide an example where one agent type is more likely to find a match than the other. Ignoring these differences can result in higher losses compared to algorithms that recognize and adjust for agent heterogeneity. They reference Ashlagi, Nikzad, and Strack (2022), which applies their model empirically and finds that in scenarios with strong heterogeneity between agent types, the general findings do not hold. Liu, Wan, and Yang (2024) further explores this concept by theoretically discussing the effects of two distinct agent types in a more generalized model, emphasizing the complexities and potential inefficiencies that arise from agent heterogeneity.

3.4 Optimizing Ride-Sharing through Queuing Models

The last model we present is from Özkan and Ward (2020), which primarily relies on pure matching techniques, although it also briefly considers the impact of pricing. The primary motivation for focusing on pure matching stems from the negative reputation associated with surge pricing, despite its potential to enhance efficiency (The Economist (2016), Ariely (2016), Goncharova (2017)). Another distinct aspect of this model is its inclusion of time-varying rates, which adds a layer of complexity and realism to the analysis. A graphical representation of their model is shown in Figure A.1, where each circle represents an area where impatient riders arrive, and the corresponding queue below it represents drivers arriving. In this model, riders must be matched almost immediately upon arrival or they will leave, i.e., perish. Drivers will also leave their queue if they wait too long. Each rider has a different probability of being compatible with a driver from each area.

This section will be structured such that we will present the theoretical framework, and then a continuous linear programming (CLP)-based policy with time-varying parameters, followed by a myopic linear programming (LP)-based policy and some alternativese to this policy. Finally, we compare the policies and discuss the simulation results from Özkan and Ward (2020).



Note: Based on Figure 1 in Özkan and Ward (2020). λ (μ) represents the arrival rate of riders (drivers). θ represents the departure rate of drivers and $\bar{F}$ represents the probability that a rider and driver are compatible.

Figure 6. Graphical presentation of the two-sided matching system

3.4.1 Theoretical Framework

The area, e.g., a city, is partitioned into $N \in \mathbb{N}$ disjoint areas. Drivers have an independent arrival rate $\mu_i(t)$ according to their area (type) $i \in \mathcal{N} := \{1, 2, \ldots, N\}$. This arrival rate is a non-homogeneous Poisson process with rate $\mu_i(t)$ at time $t \in \mathbb{R}_+$ with the cumulative rate function $\Gamma(t) := \int_0^t \mu_i(s)\,\mathrm{d}s$. This incorporates that drivers have different working hours, and at some times of the day, more drivers want to work than at other times. The time a driver wants to wait for a rider is exponentially distributed with a time-dependent rate $\theta_i(t)$ for all $i \in \mathcal{N}$ and $t \in \mathbb{R}_+$ (departure rate).

Riders can request a ride and are matched according to their type $J \in \mathbb{N}$. This categorization is due to their arrival area, their destination, and their priority status. Like drivers, type $j \in \mathcal{J} := \{1, 2, \ldots, J\}$ riders arrive in a non-homogeneous Poisson process $\lambda_j(t)$ at time $t \in \mathbb{R}_+$ and the cumulative rate function is $\Lambda(t) := \int_0^t \lambda_j(s)\,\mathrm{d}s$. This time dependency could be interpreted such that during rush hour more riders request a ride than at 3 AM on weekdays.

A matching policy $\pi = (\pi_1, \ldots, \pi_J)$ determines which rider is matched to which driver. Each π_j tracks the drivers offered to riders of type j. Let $\pi_j(k) = i$ mean that the planner wants to match the kth type j rider to a type i driver for $j \in \mathcal{J}$, $k \in \mathbb{N}$, $i \in \mathcal{N} \cup \{0\}$. Let $\pi_j(k) = 0$ mean that no driver is offered to the rider. Riders accept their match if their waiting time is not too long, e.g., most riders would prefer a 10-minute walk over a 10-minute waiting time. When the pickup time is too long, resulting in an unmatched situation or when no driver is offered to a rider, the rider perishes. The driver then stays in his arrival area. The process which tracks the cumulative number of type i drivers matched to type j riders under the policy π in $[0, t]$ is called $D_{ij}^{\pi}(t)$. The number of drivers in area i at time $t \in \mathbb{R}_+$ is $Q_i^{\pi}(t)$ and depends on π:

$$Q_i^{\pi}(t) = \underbrace{Q_i(0)}_{\text{Initial condition}} + \underbrace{A_i\left(\mu_i(t)\right)}_{\text{Arrivals}} - \underbrace{R_i\left(\int_0^t \theta_i(s) Q_i^{\pi}(s)\,\mathrm{d}s\right)}_{\text{Departure of unmatched drivers}} - \underbrace{\sum_{j \in \mathcal{J}} D_{ij}^{\pi}(t)}_{\text{Departure of matched drivers}} \geq 0$$

where A_i and R_i are unit rate Poisson processes. It is important to note that the mechanism of Özkan and Ward (2020) do not track what a driver does when he picks up a rider or leaves the area.

The planner's goal is to match as many riders as possible. Ideally, there would always be a driver available close to a potential rider when $Q_i^\pi(t) > 0$ for all $i \in \mathcal{N}$ and $t \in \mathbb{R}_+$; however, this scenario is often unrealistic. Therefore, matches occur only if the riders' waiting times are acceptably short. This is formalized in the following way. The time horizon is divided into countable disjoint intervals using a sequence $\{\tau_m, m \in \mathbb{N} \cup \{0\}\}$ where $\tau_m \in \mathbb{R}_+$, $\tau_m < \tau_{m+1}$ for all $m \in \mathbb{N}$, and $\tau_m \to \infty$ as $m \to \infty$.

Adapting to changing traffic conditions, the pickup time for a type j rider from a type i driver at any $t \in [\tau_m, \tau_{m+1}]$ is represented as $t_{ij}(t) \in \mathbb{R}_+$ for all $i \in \mathcal{N}$, $j \in \mathcal{J}$, and $m \in \mathbb{N} \cup \{0\}$. Each type j rider has an i.i.d. threshold $\{a_j^k(m), k \in \mathbb{N}\}$, independent from other processes for all $m \in \mathbb{N} \cup \{0\}$ and $j \in \mathcal{J}$. If $a_j^k(m) \geq t_{ij}(t)$, the rider will accept the waiting time and be matched. This setup allows for the calculation of the probability that a type i driver will be matched with a type j rider at a given time t.

$$\bar{F}_{ij}(t) := \sum_{m=0}^{\infty} \mathbf{P}(a_j^k(m) \geq t_{ij}(m)) \mathbb{I}(t \in [\tau_m, \tau_{m+1}))$$

Here, $\mathbb{I}$ denotes an indicator function which is 1 if t is within the specified interval, and 0 otherwise.

Incorporating the first dispatch policy (FD), similar to Castillo, Knoepfle, and Weyl (2024), Özkan and Ward (2020) defines this approach in their model as follows: A type j rider arriving at $r \in [0, T]$ is offered a driver from the set:

$$\underset{\{i \in \mathbb{N} : Q_i^{\text{FD}}(t-) > 0\}}{\operatorname{argmin}} \sum_{m \in \mathbb{N} \cup \{0\}} t_{ij}(m) \mathbb{I}(t \in [\tau_m, \tau_{m+1}]).$$

Should multiple drivers be available, the planner may select one at random.

The total number of matches under a policy π is represented by:

$$D_{ij}^\pi(t) := \sum_{k=1}^{E_j(\Lambda(t))} \sum_{m \in \mathbb{N} \cup \{0\}} \mathbb{I}\Big(v_j(k) \in [\tau_m, \tau_{m+1}), a_j^k(m) \geq t_{ij}(m), \pi_j(k) = i\Big),$$

for all $i \in \mathcal{N}$, $j \in \mathcal{J}$, and $t \in \mathbb{R}_+$. Here, E_j is a unit rate Poisson process, and $v_j(k)$ denotes the arrival time of the kth type j rider. The outer sum accounts for the number of ride requests, while the inner sum totals the intervals and checks if timing, acceptance, and policy conditions are met.

The planner aims to identify a policy that maximizes the number of matchings within a finite time horizon across all possible scenarios ω:

$$\max_{\pi \in \Pi} \sum_{i \in \mathcal{N}, j \in \mathcal{J}} w_{ij} D^{\pi}_{ij}(T, \omega), \quad \forall \omega \in \Omega^{5}$$

where, $w_{ij}, i \in \mathcal{N}, j \in \mathcal{J}$ are weights that the planner may adjust depending on specific objectives. In scenarios where all matchings are valued equally, all weights would equal 1.

Özkan and Ward (2020) note that for simplicity, they assume exogenous rider and driver behavior. This is a strong assumption since one could easily argue that the behavior depend on the planner and are thus endogenous. However Zhong, Wan, and Shen (2020) found evidence that ride-sharing can be modeled with a Markovian queuing model with exogenous behavior, which strengthens Özkan and Ward (2020) assumption. Nevertheless, they have very strong objectives because solving either requires specifying a policy that maximizes the number of matchings on every sample path.

Admission Policy Özkan and Ward (2020) rigorously define how an admission policy should operate and the information it should utilize to match riders to drivers. They outline a filtration of the entire feature space, which a policy can use to match drivers and riders. Crucially, the policy is designed to prevent any foresight into specific future arrivals, maintaining a realistic scenario where policies cannot predict exact future events, though they are permitted to anticipate arrivals.

For the upper bound of their objective, they incorporate information regarding riders' willingness to wait; however, this information is not used in later proposed policies. It is important to note that the exact waiting tolerance of individual riders at any given time t is unknown, which would otherwise enable the planner to cherry pick riders willing to wait for a very long periods.

The planner has the capability to forecast arrival rates for different types of riders and drivers, as well as driver acceptance probabilities. The set of all admissible policies is Π and we will call the first dispatch policy π_{FD}. Özkan and Ward (2020) show that $\pi_{FD} \in \Pi$. Furthermore, Π is finite.

3.4.2 Upper Bound Derivation through Continuous Linear Programming

This section first establishes a benchmark and then introduces a policy that is asymptotically optimal. Since most of this section relies on the concept of fluid scaling, we will present a short and intuitive introduction[6]. To determine an upper bound on system performance, Özkan and Ward (2020) employ CLP, which utilizes fluid scaling. This technique is frequently used in optimization problems in queuing theory and, unlike LP, accounts for the expected number of future arrivals and departures, allowing the system to strategically reserve drivers for upcoming riders rather than operating myopically.

5. Notice that Ω is part of the definition of the complete probability space, included here due to certain technical details that are not further explored in this discussion.

6. For a more formal introduction, refer to Shortle et al. (2018).

Concept of Fluid Scaling Fluid scaling is a mathematical technique applied to approximate the behavior of large, stochastic systems using deterministic models. Typically, an item arrives at a random rate within a system, such as in a simple queuing process. Over a large time horizon, the expected values smooth out the inherent randomness, transforming the queuing process into a scaled version by adjusting time and system size with a large parameter n. As n approaches infinity, the behavior of the scaled system converges to that of a deterministic fluid model, applying the Law of Large Numbers. This reduces the impact of random fluctuations, making the system's behavior more predictable and closely aligned with its average behavior (Shortle et al., 2018).

Optimal policies derived using fluid models are then applicable to the original stochastic system, as discussed in Shortle et al. (2018). Another implication of employing fluid scaling and CLP is that all results occur with a probability converging to 1, indicating that these results hold almost surely (a.s.) except for some rare sample paths.

Large Market Assumption Özkan and Ward (2020) explore the scalability of their matching system using fluid scaling, which necessitates a large market assumption. They examine a sequence of matching systems, indexed by n, $n \in \mathbb{N}$, each adapted from the structure introduced in Section 3.4.1 but with parameters that depend on n. For the nth matching system, the cumulative arrival and service rates are defined as follows:

$$\Lambda_i^n(t) \ := \ \int_0^t \lambda_i^n(s)\,\mathrm{d}s, \quad \Gamma_j^n(t) \ := \ \int_0^t \mu_j^n(s)\,\mathrm{d}s,$$

where μ_j^n and λ_i^n are Lebesgue-measurable rate functions and well-defined. Özkan and Ward (2020) ensure positive driver departure rates across all systems. A matching policy $\pi = \{\pi^n, n \in \mathbb{N}\}$ specifies a strategy for each n.

Assumption 19 (Large Market)**.** $\frac{\Lambda_i^n}{n} \longrightarrow \Lambda_i$, $\frac{\mu_j^n}{n} \longrightarrow \mu_j$, and $\theta_i^n \longrightarrow \theta_i$ uniformly on compact intervals, as $n \longrightarrow \infty$ for all $i \in \mathcal{N}$ and $j \in \mathcal{J}$ such that $\Gamma_i = \int_0^t \mu_i(t)\,\mathrm{d}t$ for all $t \in \mathbb{R}_+$. Furthermore, the unit rate Poisson processes A_i, R_i and E_k are independent, and $\theta_i(t), \lambda_j(t)$, and $\mu_i(t)$ are bounded.

Özkan and Ward (2020) transform their variables to a fluid scaling. These are defined as follows.

$$\begin{aligned}
\bar{A}_i^n(t) \ &:= \ A_i(nt)/n, \quad \bar{R}_i^n(t) \ := \ R_i(nt)/n, \quad \bar{E}_j^n(t) \ := \ E_j(nt)/n, \\
\bar{\Gamma}_i^n \ &:= \ \Gamma^n/n, \quad \bar{\Lambda}_j^n \ := \ \Lambda^n/n, \quad \bar{Q}_i^{\pi,n} \ := \ Q_i^{\pi,n}/n, \quad \bar{D}_{ij}^{\pi,n} \ := \ D_{ij}^{\pi,n}/n. \\
\bar{Q}_i^{\pi,n}(t) \ &= \ \bar{Q}_i^n(0) + \bar{A}_i^n(\bar{\mu}_i^n(t)) - \bar{R}_i^n\left(\int_0^t \theta_i^n(s)\bar{Q}_i^{\pi,n}(s)\mathrm{d}s\right) - \sum_{j\in\mathcal{J}} \bar{D}_{ij}^{\pi,n}(t).
\end{aligned}$$

Where $\bar{Q}_i^{\pi,n}(t)$ is the scaled queue length process which will be further used from Özkan and Ward (2020).

Assumption 20 (Initial Conditions)**.** For all $i \in \mathcal{N}$, $\bar{Q}_i^n(0) \longrightarrow \bar{Q}_i(0)$ as $n \longrightarrow \infty$, where $\bar{Q}_i(0) \in \mathbb{R}_+$.

Assumption 20 states that the scaled initial number of drivers in all areas is asymptotically the same across all systems as the market grows. Both Assumption 20 and 19 hold for the remainder of the summary of the paper from Özkan and Ward (2020).

Decision Variables The decision variables in the model are denoted by $\{\boldsymbol{q}, \boldsymbol{x}\} := \{q_i, x_{ij}, i \in \mathcal{N}, j \in \mathcal{J}\}$. Here, q_i represents the number of type i drivers at time t, ensuring the system's inflow and outflow are balanced, thus maintaining stability. Let $x_{ij}(t)$ indicates the fraction of type j riders who are offered type i drivers at time t, and thus beeing the matching process. The expression $\lambda_j(s)\bar{F}_{ij}(s)x_{ij}(s)$ calculates the instantaneous matching rate between type i drivers and type j riders.

The planner's objective is to maximize the weighted number of matches between drivers and riders:

$$
\begin{aligned}
\max_{\boldsymbol{q},\boldsymbol{x}} \quad & \sum_{i\in\mathcal{N}, j\in\mathcal{J}} w_{ij} \int_0^T \lambda_j(s)\bar{F}_{ij}(s)x_{ij}(s)\,\mathrm{d}s \\
\text{s.t.} \quad & q_i(t) \;=\; \bar{Q}_i(0) + \mu_i(t) - \int_0^t \theta_i(s)q_i(s)\,\mathrm{d}s - \sum_{j\in\mathcal{J}} \int_0^t \lambda_j(s)\bar{F}_{ij}(s)x_{ij}(s)\,\mathrm{d}s, \\
& \sum_{i\in\mathcal{N}} x_{ij}(t) \;\leq\; 1, \\
& q_i(t) \;\geq\; 0, \quad x_{ij}(t) \;\geq\; 0, \\
& \ldots
\end{aligned}
$$

The first constraint ensures the balance of the queue length by accounting for the initial number of drivers, arrivals, departures, and matches made. The second constraint ensures that each rider is offered at most one driver. The third constraint maintains that the number of drivers available for matching in any area is non-negative, and similarly, the probability of matching is always non-negative. Furthermore, Özkan and Ward (2020) incorporate additional technical measurement constraints related to the decision variables.

The optimal solution to the problem is defined by a unique feasible queue length process, defined as $\{\tilde{\boldsymbol{q}}, \tilde{\boldsymbol{x}}\} := \{q_i, x_{ij}, j \in \mathcal{N}, j \in \mathcal{J}\}$. Özkan and Ward (2020) state with the optimal solution to the CLP theorem 21.

Theorem 21. *Under any admissible policy π,*

$$
\limsup_{n\to\infty} \sum_{i\in\mathcal{N}, j\in\mathcal{J}} w_{ij}\bar{D}_{ij}^{\pi,n}(T) \;\leq\; \sum_{i\in\mathcal{N}, j\in\mathcal{J}} w_{ij} \int_0^T \lambda_j(s)\bar{F}_{ij}(s)\tilde{x}_{ij}(s)\,\mathrm{d}s, \quad \text{a.s.}
$$

This means that the optimal solution value of the CLP is an asymptotic upper bound on the fluid-scaled objective under any admissible policy π.

3.4.3 A Continious Linear Programm Policy

In this section, we introduce a randomized policy that can asymptotically achieve the optimal upper bound established in section 3.4.2. Özkan and Ward (2020) describe their policy as follows:

Definition 22 (Randomized Policy)**.** If a type j rider arrives in the system at time t, the planner makes a random selection from the set $\mathcal{N} \cup \{0\}$ such that the probability that the outcome is i is $x_{ij}(t)$ for all $i \in \mathcal{N}$ and the probability that the outcome is 0 is $1 - \sum_{i \in \mathcal{N}} x_{ij}(t)$. If the outcome is i for some $i \in \mathcal{N}$ and there is a type i driver in the system, then the system controller offers a type i driver to the rider. If the outcome is i for some $i \in \mathcal{N}$ but there is no type i driver in the system or if the outcome is 0, then no driver is offered to the rider.

This policy, denoted as π_R, depends solely on $\boldsymbol{x}$ as a decision variable. Özkan and Ward (2020) demonstrate that π_R is admissible according to their criteria for an admissible policy. The introduction of randomness into the matching decisions allows the policy to manage uncertainties and avoid over-committing to specific matches, enhancing flexibility in handling real-time variability. However, the drawback of such a probabilistic approach is that not every rider is guaranteed a ride even when spare drivers are available, making it not match conserving compared to other policies like π_{FD}.

Özkan and Ward (2020) show that π_R can mimic any feasible process pair of the CLP. This is captures in therorm 23.

Theorem 23. *Let $\{\boldsymbol{q}, \boldsymbol{x}\}$ be a feasible process pair for the CLP. For all $i \in \mathcal{N}$, as $n \to \infty$,*

$$\sup_{0 \le t \le T} \left| \sum_{j \in \mathcal{J}} \bar{D}_{ij}^{\pi_R(\boldsymbol{x}),n}(t) - \int_0^t \lambda_j(s) \bar{F}_{ij}(s) x_{ij}(s)\, \mathrm{d}s \right| \xrightarrow{a.s.} 0 \textit{ and } \left\| Q_i^{\pi_R(\boldsymbol{x}),n} - q_i \right\|_T \xrightarrow{a.s.} 0.$$

The first part of theorem 23 measures the maximum absolute difference in the deterministic number of matches from the optimum achieved by π_R. It asserts that this difference converges to zero for all $t \in [0, T]$. This implies that at no point does the actual number of matches substantially deviate from the expected number, ensuring that the policy performs consistently with the predictions over time.

The second part of theorem 23 focuses on the queue lengths within the system. It demonstrates that the supremum norm of the queue lengths under the randomized policy converges to the fluid limit as the system size increases. Employing the supremum norm likely serves to provide a measure of the worst-case scenario in the differences between the two, thereby strengthening their results.

We already know that the cumulative number of matches made cannot exceed the optimal CLP objective value. However, by using $\tilde{\boldsymbol{x}}$ from the optimal solution of the CLP, the randomized policy can achieve the asymptotic upper bound on the number of matches. This is evident in the theorem when one inserts $\tilde{\boldsymbol{x}}$ instead of any feasible process states. Using the optimal solution $\tilde{\boldsymbol{x}}$ transforms the randomized policy to a policy which anticipates the future and can adjust to time varing arrival rates.

3.4.4 A Linear Programming Policy

Given that solving a CLP problem is NP-hard, it is challenging, and sometimes impossible, to calculate a solution. This section explores three cases where the problem can be simplified into a LP model, with the first two cases not involving any pricing and the third being a joint optimization scenario.

Busy drivers and time-homogeneous parameters The first of the two conditions requires that drivers be fully utilized; that is, as soon as a driver arrives in the system, the planner matches them with a waiting rider. The second condition assumes that all parameters are time-homogeneous. Although these conditions appear distinct, they are similar in their effects on the system dynamics. Both lead to a stable queue length for drivers: in the first scenario, any arriving driver is quickly matched, typically keeping the queue length near zero, while constant arrival rates in the second scenario balance the inflow and outflow of drivers and riders. Consequently, both conditions facilitate a constant flow through the system.

These scenarios also imply that the planner does not need to anticipate future events, making the system myopic. The decision variable, $x_{ij}(t)$, represents the fraction of type j riders that should be offered to a driver waiting in area i at each time $t \in [0, T]$. The associated LP formulation is outlined as follows:

$$\begin{aligned}
\max_{x(t)} \quad & \sum_{i\in\mathcal{N}, j\in\mathcal{J}} w_{ij}\lambda_j(t)\bar{F}_{ij}(t)x_{ij}(t) \\
\text{s.t.} \quad & \sum_{j\in\mathcal{J}} \lambda_j(t)\bar{F}_{ij}(t)x_{ij}(t) \;\le\; \mu_i(t), \\
& \sum_{i\in\mathcal{N}} x_{ij}(t) \;\le\; 1, \\
& x_{ij}(t) \;\ge\; 0,
\end{aligned}$$

The objective is the weigthed instantaneous matching rate between drivers in area i and type j riders is $\lambda_j(t)\bar{F}_{ij}(t)x_{ij}(t)$. The first constraint ensures that the total matching rate for drivers of type i does not exceed the arrival rate of these drivers, effectively maintaining queue length stability. The second constraint ensures that the total matching probability for any rider does not exceed 1. The third constraint ensures that all matching rates are non-negative, which is necessary for the feasibility of the solution. The optimal solution to this LP is denoted by $\boldsymbol{x}^*$. To establish the conditions under which this solution is optimal, Özkan and Ward (2020) state lemma 24:

Lemma 24. *Suppose that*

$$\begin{aligned}
\bar{Q}_i(0) \;&=\; 0, \quad \forall i \;\in\; \mathcal{N}, \\
\textit{and} \sum_{i\in\mathcal{N}, j\in\mathcal{J}} \int_0^t \lambda_j(s)\bar{F}_{ij}(s)x_{ij}(s)\,\mathrm{d}s \;&=\; \sum_{i\in\mathcal{N}} \Gamma_i(t) \quad \textit{for all } t \;\in\; [0, T],
\end{aligned}$$

The first condition of lemma 24 requires that all areas start with an empty queue. It ensures that the initial state of the system is well-defined and that the subsequent dynamics can be clearly attributed to the matching process and arrival rates. The second condition states that the system perfectly balances demand (rider arrivals) with supply (driver arrivals) through the matching process. Therfore the total number of matches made over time corresponds consistently with the total number of drivers arriving in the system.

With these conditions met, Özkan and Ward (2020) are able to compute the optimal solution, $\boldsymbol{x}^*$, and establish the upper bound of the number of matches for the problem. These findings are summarized in Theorem 25:

Theorem 25. *If lemma 24 holds, under any admissible policy π, we have*

$$\limsup_{n\to\infty} \sum_{i\in\mathcal{N},j\in\mathcal{J}} \bar{D}_{ij}^{\pi,n}(T) \leq \sum_{i\in\mathcal{N},j\in\mathcal{J}} \int_0^T \lambda_j(s)\bar{F}_{ij}(s)x_{ij}^*(s)\,ds, \quad a.s.$$

Özkan and Ward (2020) apply the same π_R policy as discussed in section 3.4.3. They demonstrate that π_R remains admissible within the context of LP, provided that the optimal solution $\boldsymbol{x}^*$ is Lebesgue measurable. Under this condition and lemma 24, they are able to prove that $\pi_R(\boldsymbol{x}^*)$ can achieve the upper bound on the number of matches as presented in theorem 25.

Prices This section addresses the third scenario, where neither the arrival rates are time-homogeneous nor are all drivers fully utilized at a uniform price p. By jointly optimizing pricing and matching, the system can achieve full utilization of drivers and optimal performance. The intuition here is similar to surge pricing presented in Castillo, Knoepfle, and Weyl (2024), where the planner may adjust the price for riders to stimulate higher demand. In this model the planner can change in the price affects also drivers and thus also adjusts suppy.

To implement this strategy, the planner must solve the following LP problem. It is important to note that λ_j and μ_i are constant functions influenced by the price p, determined by the planner at $t = 0$. Additionally, the departure rate θ_i may vary with both time and price p. The upper limit for chargeable prices is $\bar{p} \in \mathbb{R}_+$. The planner can set a price vector $\boldsymbol{p} := \{p_{ij}, i \in \mathcal{N}, j \in \mathcal{J}\}$ as an addional decision variable.

$$\begin{aligned}
\max_{\boldsymbol{x},\boldsymbol{p}} \quad & \sum_{i\in\mathcal{N},j\in\mathcal{J}} w_{ij}\lambda_j(\boldsymbol{p})\bar{F}_{ij}x_{ij} \\
\text{s.t.} \quad & \sum_{j\in\mathcal{J}} \lambda_j(\boldsymbol{p})\bar{F}_{ij}x_{ij} \leq \mu_i(\boldsymbol{p}), \\
& \sum_{i\in\mathcal{N}} x_{ij} \leq 1, \\
& x_{ij} \geq 0, \quad p_{ij} \in [0, \bar{p}].
\end{aligned}$$

This LP problem mirrors the scenario with fully utilized drivers, with the addition of p_{ij} as a decision variable. The added constraint ensures that the planner sets non-negative, chargeable prices. The optimal solution of the prices to this LP is denoted as $\boldsymbol{p}^*$, and the associated randomized policy, incorporating both the matching vector $\boldsymbol{x}$ and the price vector $\boldsymbol{p}$, is referred to as $\pi_{R,\boldsymbol{p}}(\boldsymbol{x})$. Özkan and Ward (2020) confirm that this randomized policy is admissible under their defined conditions.

Özkan and Ward (2020) show that when the surge multiplier in an area decreases, the rider arrival rate in that area increases, and the driver arrival rate in that area decreases. This change in the mutiplier does not affect the rates in the other areas. Furthermore they require that drivers have to be paid.

Corollary 26. *Suppose that the first condition of lemma 24 holds and $\{\boldsymbol{x}^*, \boldsymbol{p}^*\}$ is an optimal solution of the above LP. Then, under any admissible policy $\pi(\boldsymbol{p})$,*

$$\limsup_{n\to\infty} \sum_{i\in\mathcal{N}, j\in\mathcal{J}} w_{ij}\bar{D}_{ij}^{\pi(\boldsymbol{p}),n}(T) \;\leq\; \sum_{i\in\mathcal{N}, j\in\mathcal{J}} T w_{ij}\lambda_j(\boldsymbol{p}^*)\bar{F}_{ij}x_{ij}^*, \quad a.s.$$

Moreover, if $w_{ij} = 1$ for all $i \in \mathcal{N}$ and $j \in \mathcal{J}$, then $\pi_{R,\boldsymbol{p}^*}(\boldsymbol{x}^*)$ is asymptotically optimal; that is,

$$\lim_{n\to\infty} \sum_{i\in\mathcal{N}, j\in\mathcal{J}} \bar{D}_{ij}^{\pi_{R,\boldsymbol{p}^*}(\boldsymbol{x}^*),n}(T) \;=\; \sum_{i\in\mathcal{N}, j\in\mathcal{J}} T\lambda_j(\boldsymbol{p}^*)\bar{F}_{ij}x_{ij}^*, \quad a.s.$$

An imporant consideration is that the under a good pricing starategy the first constraint of the LP is always binding. A good pricing startegy means that there are no idle drivers, otherwise lowering the price would mean that fewer drivers and more riders would come to an area. Özkan and Ward (2020) formalize this observation with the following theorem:

Theorem 27. *There exists an optimal solution in which all drivers are always busy, e.g. the first constraint of the LP is binding.*

This theorem suggests that when multiple optimal solutions to the LP exist, the planner can always opt for the one where the constraint is binding. While this scenario theoretically maximizes utility by ensuring continuous driver engagement, it overlooks a critical aspect of ride-sharing reliability compared to traditional cap services. If all drivers are always busy there might be a small perserntage of drivers who are not offered a seat are rejected and thus perish. These kind of problems are not explicitly considered in Özkan and Ward (2020).

However, a quite simple solution to this kind of problem is already stated in Castillo, Knoepfle, and Weyl (2024). A small price increase in all areas will result in a slighly higher arrival rate for riders and lower arrival rate for riders such that there is always a small number of spare drivers in the city.

3.4.5 Alternative Matching Policies

As previously discussed, the randomized policy may have sample paths where a rider is not offered a ride even though a suitable driver is available and would otherwise leave the system. This section presents three alternative policies that are admissible according to Özkan and Ward (2020). Two of these three policies are match-preserving, ensuring that a rider is always offered a driver if one is available. The first new policy is the Randomized Weighted Queue Policy (RWQP), the second is deterministic, and the third is a hybrid. All three alternative policies can be implemented using either CLP or LP frameworks.

Randomized Weighted Queue Policy The RWQP is a modification of the randomized policy, incorporating not only the matching probability vector $\boldsymbol{x}$ but also the queue lengths of drivers, enhancing the match potential based on availability. Let $Q_i(t-)$ represent the queue length of type i drivers just before time t in $\mathbb{R}_+$. The RWQP is defined as follows:

Definition 28 (Randomized Weighted Queue Policy)**.** If a type j rider arrives in the system at time t, the system controller offers a type i driver with probability

$$\frac{x_{ij}(t)Q_i(t-)}{\sum_{i\in\mathcal{N}} x_{ij}(t)Q_i(t-)},$$

if the denominator in is strictly positive. Otherwise, the system controller does not offer any driver to the rider.

This policy assigns a higher probability to drivers from queues with larger lengths at $Q_i(t-)$. The advantage of RWQP is its responsiveness to fluctuations in driver availability at specific locations. For instance, if an unusually high number of drivers converge in area i, they are more likely to be matched fast, thus preventing them from leaving the market unmatched, which might occur under the standard π_R.

Deterministic Policy The next alternative is a deterministic policy, as also first dispatch policy. Recall that $D_{ij}(t-)$ tracks the number of type i drivers matched to type j riders in $[0, t-]$.

Definition 29 (Deterministic Policy)**.** If a type $j \in \mathcal{J}$ rider arrives in the system at time t, the system controller offers a driver from the set determined by:

$$\underset{\{i\in\mathcal{N}:Q_i(t-)>0\}}{\arg\min} \left\{ D_{ij}(t-) - \int_0^t \lambda_j(s)\bar{F}_{ij}(s)x_{ij}(s)\,\mathrm{d}s \right\}.$$

If more than one driver type meets this criterion, the planner may select one at random. If no drivers are available, then no driver is offered.

This policy minimizes the difference between the actual number of matches and the expected number of matches, ensuring that the selected driver type has the smallest discrepancy between actual and expected matches.

Hybrid Policy The hybrid policy combines the advantages of π_R with a backup option to ensure a safe match. Özkan and Ward (2020) define it as follows:

Definition 30 (Hybrid Policy)**.** Suppose that a type j rider arrives in the system and there is a driver in the system. Then the planner uses π_R to determine a matching. If the outcome is $i \in \mathcal{N}$ and there is a type i driver in the system, the planner offers a type i driver to the rider. If the outcome is $i \in \mathcal{N}$ but there is no type i driver in the system or if the outcome is 0, the rider is offered the driver type specified by the first dispatch policy. If there is no driver in the system, then the rider perishes.

It is important to note that Özkan and Ward (2020) conjecture that all three of these policies are asymptotically optimal. However, proving such optimality can be technically rigorous and complex. We will explore potential issues with all four policies in section 5, following the presentation of simulation results for each policy in the next section.

3.4.6 Simulation Analysis

Özkan and Ward (2020) present different simulation results to support their theoretical analysis. They utilize a simple structure with three areas, as depicted in Figure A.1. Each simulation varies in terms of arrival rates, departure rates, and probabilities.

The results demonstrate that all four policies perform asymptotically the same in large markets, while in smaller markets, the deterministic and hybrid policies tend to perform better. Moreover, significant differences emerge between the CLP and LP when using time-varying rates. These discrepancies are more pronounced when drivers are more patient and nearly disappear when drivers are very impatient, quickly exiting the queue after their arrival. These simulations support the conjecture that all four proposed policies are asymptotically optimal.

3.4.7 Discussion and Implications

Özkan and Ward (2020) establish a robust theoretical framework for optimizing real-time matches between drivers and riders. These policies generally assume a large market (Assumption 19), which may not be applicable in smaller ride-sharing markets or in ridepooling services like *BlaBlaCar*. Nonetheless, simulation results suggest that these policies offer improvements over a simple first dispatch policy even in smaller markets, making them a valuable starting point for developing optimal policies in smaller markets.

Unfortunately, their analysis does not account for the potential emergence of WGCs. Consider the following example:

Example 31. Suppose all drivers are utilized and one arrives in area i. Simultaneously, a rider requests a very short ride in area $-i$, which is far away from area i[7]. Additionally, assume the rider accepts the long wait for a driver from area i. Under the Hybrid Policy, the type i driver would be offered to this rider, setting the stage for a potential WGC.

This suggests the need for further research in this area. Another limitation is the practical challenge of solving the CLP, resulting that the planner has to rely on the less optimal LP policy, despite knowing that a better solution exists. Enhancements could be explored by tracking drivers after the drop-off to refine the model further. Özkan and Ward (2020) provide insights on how this aspect could be integrated into their existing framework.

7. Note that this example is simplified, and "far away" is used in a general sense.

3.5 Comparison

This section compares the three models presented, illustrating their similarities and differences in results. Furthermore, it concludes with the main takeaways from the current literature, which are crucial in our recommendations for BlaBlaCar.

Although all three models address dynamic matching, each provides insights from a different perspective, with varying degrees of generality and specificity. Castillo, Knoepfle, and Weyl (2024) focus on practical issues in ride-hailing, Akbarpour, Li, and Gharan (2020) present a more general theoretical framework, and Özkan and Ward (2020) offer a detailed model for real-time ride-sharing using methods from queuing theory. Despite their different perspectives, all models aim to maximize the number of matches.

Castillo, Knoepfle, and Weyl (2024) seek to maximize total utility by addressing the problem of WGC, a type of market failure not discussed in the other two models. According to Castillo, Knoepfle, and Weyl (2024), surge pricing, is a solution to WGC. An example previously provided shows how the model of Özkan and Ward (2020) can result in a WGC. Özkan and Ward (2020) utilize pricing, but within a different framework, influencing arrival rates and allowing only to be set once . However, it should be possible to incorporate dynamic pricing into their model.

Both Akbarpour, Li, and Gharan (2020) and Özkan and Ward (2020) offer pure matching algorithms, yet they yield different outcomes due to their underlying goals. Akbarpour, Li, and Gharan (2020) aim not to provide an optimal theoretical algorithm but a simplified version that is computationally feasible and close to the optimal solution. Özkan and Ward (2020) introduce the CLP policy, which is computationally intensive and even infeasible for some markets. However, both models show that, in their respective settings, a planner can achieve a higher number of matches if the agents are patient. In Akbarpour, Li, and Gharan (2020), there is only one type of agent, whereas Özkan and Ward (2020) include patient drivers. We conjecture that patient riders would also enhance efficiency in the model of Özkan and Ward (2020).

An important consideration is market size. While both Castillo, Knoepfle, and Weyl (2024) and Özkan and Ward (2020) address large markets, Akbarpour, Li, and Gharan (2020) also discuss strategies for smaller numbers of agents. This assumption might be valid for large ride-sharing companies in major cities, but in smaller towns and companies, the reality of a smaller market with fewer drivers and riders must be acknowledged.

Key Takeaways The primary insight from this analysis is that by strategically addressing matching policies, planners can significantly enhance the number of successful matches. A crucial factor in optimizing these matching scenarios is the patience of drivers and riders; moreover, having reliable information about expected durations within the system proves to be invaluable. Another takeaway is the role of pricing strategies in enhancing matching efficiency. Surge pricing emerges as an effective solution to the market failure known as WGC. Additionally, implementing area-specific pricing can strategically influence the distribution of drivers and riders across different regions, further optimizing the matching process.

4 Application on BlaBlaCar

In this section, we explore potential implications of three distinct models for *BlaBlaCar*. Due to the absence of specific data or information regarding the primary challenges faced by *BlaBlaCar*, we must rely on assumptions regarding the company's goals and values. The initial assumptions are more realistic, whereas the final assumption has a about strong objective. The last assumption is about the behavior of riders.

4.1 Assumptions

BlaBlaCar presents itself on their website BlaBlaCar (2024a) as primarily a ride-pooling service, emphasizing its contributions to societal and environmental benefits. They assert that their carpooling services save approximately 1,6 million tonnes of CO_2 annually. They claim that this is due to doubling the number of people traveling while using the same number of cars. However, they do not cover if these people otherwise would have been traveling by train, or maybe not traveling at all (BIPE, d'Espous, and Wagner, 2019). Therefore, the 1,6 million tonnes likely represents an upper limit. Additionally, *BlaBlaCar* claims to foster social connections by bringing together individuals who might not otherwise meet, supported by survey data from active members of the *BlaBlaCar* community (d'Espous, Deniau, and Nguyen, 2018). Based on this information, we conclude that *BlaBlaCar* prioritizes maintaining its image as a company more focused on advancing society and climate rather than a revenue-maximizing company.

The company's approach to pricing decisions significantly influences its public image. As of the writing of this thesis, *BlaBlaCar* recommends a price point to drivers, who can choose to adjust this recommendation. Importantly, riders do not see the direct price per seat because *BlaBlaCar* includes additional charges—both to cover VAT payments to the government and a service fee to fund operational costs. Unfortunately, details regarding the service fee, remain undisclosed. However the additional charges are between 18% to 21%. Giving drivers power over setting the prices has two benefits. First, it results in lower prices through competition and second, it preserves *BlaBlaCar*'s image as a ride pooling platform.

Assumption 32 (Pricing Decisions)**.** *BlaBlaCar* enables drivers to set their prices after receiving a pricing recommendation. The company's revenue from the ride-pooling service primarily comes from the service fees added on top.

BlaBlaCar promotes an experience where drivers and riders enjoy meaningful journeys together (d'Espous, Deniau, and Nguyen, 2018). Consequently, it is essential for drivers to have the autonomy to decline passengers based on past interactions or personal preferences. Similarly, riders should select their favorite drivers and not be matched with someone they do not prefer.

Assumption 33 (Quality of Matches)**.** *BlaBlaCar* must allow drivers and riders to refuse or select each other, ensuring that the matching mechanism accounts for individual preferences. This means that *BlaBlaCar* cannot use a simple mechanism which does not take preferences into account.

Furthermore, it is presumed that *BlaBlaCar*'s core objective within its carpooling service is to maximize the number of successful driver-rider matches, rather than focusing solely on profit.

Assumption 34 (Objective)**.** The primary goal of *BlaBlaCar* is to maximize successful matches between drivers and riders.

Lastly, we consider the behavior of riders. Given *BlaBlaCar*'s competitive pricing, it is reasonable to assume high price sensitivity among riders. Research by Farajallah, Hammond, and Pénard (2019) supports this, indicating that drivers who offer lower prices per seat tend to reach full capacity first. Additionally, riders appear to demonstrate low time sensitivity, potentially willing to travel to more remote pickup locations if the price is right.

Assumption 35 (Behavior of Riders)**.** We assume that riders exhibit high price elasticity, meaning that small changes in prices will lead to significant fluctuations in the number of riders. Additionally, we assume that riders are not very time-sensitive, such that they are willing to travel to distant pickup locations if the price is sufficiently low[8].

4.2 Analysis

This section outlines a conceptual framework for modeling the assumptions discussed previously. Given the scope of this thesis it does not include a technical and rigorous model, we will proceed with an informal description of the model and focusing on intuition.

Model Description Assume that both riders and drivers arrive according to independent Poisson processes, which can vary over time to accommodate for fluctuating supply and demand, such as during holidays compared to workdays. Each driver has a fixed departure time that they must communicate to *BlaBlaCar*, making this information known to the platform's planner.

Riders, on the other hand, have two options: a fixed departure, where they specify the exact date of travel, or a soft departure, indicating availability over multiple days. These details are also accessible to the planner, who can then estimate the likelihood of compatibility between a driver and a rider. Compatibility assessment can be conducted in two ways: first, if they have previously shared a ride and both left positive reviews, their compatibility score approaches 1. Secondly, both participants receive a quality score ranging from 0 to 1—based on their profiles, with 1 being the most favorable. This score helps predict potential matches, under the assumption that individuals compatible with a wide range of others are likely to match well together. Additionally, drivers have the option to select stopovers during their journey, which are observable and can further influence match probabilities.

8. This assumption is supported by data from France, where most *BlaBlaCar* customers are young adults. Typically, students possess some form of public transport ticket allowing them unrestricted movement within a broad area of the city.

Goal The objective of the planner is to maximize the number of successful matches, subject to certain constraints. First, the departure times of the driver and rider must coincide, meaning both must be at the same time in the network. Secondly, both parties must find each other acceptable based on their respective criteria.

Possible Mechanism An optimal matching mechanism should consider factors beyond simple distance or compatibility scores. To illustrate:

Example 36. Assume two drivers (a, b) and two riders (c, d) are all compatible with each other. The distance from driver a to rider d is less than from a to c. Both drivers start in *Cologne*; driver a is headed to Hamburg with a stop in *Bremen*, while driver b is going to *Bremen*. Rider c wants to reach *Hamburg*, and rider d wishes to travel to *Bremen*. It is evident that the optimal arrangement would be for a to take c and b to take d. A mechanism focusing solely on distance might leave driver b and rider c without a match in *Cologne* (because of the closer distance it would recommend a to match d).[9]

A promising approach to refine this matching process is inspired by the model from Özkan and Ward (2020). The main difference here is that the probability of beeing acceptable is determined not just by the given by areas but also by the compatibility scores discussed earlier. The planner can then be more flexible with riders who have a soft departure, in comparaison to riders with a fixed deadline.

To encourage truthful reporting of preferences, the planner could implement a system similar to that proposed by Akbarpour, Li, and Gharan (2020), penalizing riders who frequently misreport their preferences. For instance, if a rider attempts to secure a match on several consecutive days for the same route, it could indicate greater flexibility than originally stated.

As the planner has to ensure mutual acceptance of driver and rider the mechanism there must be some sort of proposal for the other side. As this is already the case it would be benefical if only the riders could propose for a ride to the drivers. To encorporate this into the model one could start with probably the most prominet model in this field by Gale and Shapley (1962).

We will use our ideas from this section give some advide to *BlaBlaCar* in section 4.4.

4.3 Simulation

To validate our proposed suggestions, we conducted several simulations using the `jupyter` environment and the `matchingmarkets` package developed by Ranger (2021). This package includes various matching algorithms, notably the Greedy and Patient algorithms described by Akbarpour, Li, and Gharan (2020). Moreover, it facilitates the modification of both the arrival and departure rates of participants, enabling simulations of scenarios where agents exhibit more patience as in Akbarpour, Li, and Gharan (2020). The detailed code and the results from these simulations are documented in Appendix **??**.

9. A similar example can be drawn with compatibility-based recommendation.

Table 1. Results of the Simulation

Arrival	Departure	Algorithm	Matches	Perished	Loss%
1	1	Greedy	873.6	4127.0	0.8253
		Patient	1512.8	3487.8	0.6975
1.5	1	Greedy	1220.0	3780.6	0.7560
		Patient	2036.2	2965.0	0.5929
1	3	Greedy	860.8	4139.4	0.8278
		Patient	2572.4	2428.2	0.4856
1.5	3	Greedy	1266.0	3735.2	0.7469
		Patient	3175.8	1824.6	0.3649
1	5	Greedy	882.0	4118.2	0.8236
		Patient	3301.2	1699.4	0.3398
1.5	5	Greedy	1234.0	3767.0	0.7532
		Patient	3890.0	1110.8	0.2221
1	10	Greedy	888.8	4111.6	0.8223
		Patient	4291.6	709.4	0.1419
1.5	10	Greedy	1234.8	3765.8	0.7531
		Patient	4673.8	327.6	0.0655

Note: The table summarizes the simulation results presented in Appendix **??**. Each row represents averages over 5 simulations, each with 5000 agents, resulting in a total of 25000 agents per treatment. The standard deviation for Loss is ≤ 0.01.

For our simulations, we specifically explore scenarios representative of a thin market, characterized by relatively low arrival rates, which accurately reflects the operational dynamics of *BlaBlaCar*. We introduced significant heterogeneity in departure rates, going up to a Poisson process with a mean up to 10. This rate is 10 respective 6.6 times higher than the arrival rates we tested. Across all scenarios, the probability of a match between any two agents was held constant at 0.2.[10]

The conclusion drawn from our simulations is that the Patient algorithm exhibits robust performance when agents remain in the system for extended periods. This observation aligns well with the theoretical analysis presented by Akbarpour, Li, and Gharan (2020). Moreover, we observed that the effectiveness of this algorithm persists across scenarios with varying match probabilities, both higher and lower than the baseline.

4.4 Suggestions for BlaBlaCar

This section outlines practical recommendations for *BlaBlaCar*, derived from our analysis in section 4.2 and the results of simulations discussed in section 4.3.

Improve Matching Our first recommendation focuses on enhancing the matching process. According to Akbarpour, Li, and Gharan (2020), in a thin market, the benefits of patience in matching are not as pronounced as they are in a thicker market. Additionally, a

10. Additionally, the `matchingmarkets` package offers the capability to graphically represent random processes over short time horizons. Due to technical limitations, these visualizations are not included in the main body of this thesis. For the relevant code, we refer to Appendix **??**.

thinner market makes it computationally more viable to be selective about who is matched with whom. Therefore, we suggest that *BlaBlaCar* refine their matching strategies. One practical approach could involve enabling driver changes during the journey, particularly for longer trips. Since drivers often make stopovers, typically in major cities, facilitating driver switches could render long-distance carpooling more practical and appealing. Furthermore, ways to thicken the market should be exploited such as the the differnce between soft and hard departure mentioned in out analyis.

Another improvement could involve nudging drivers towards optimal matches. Similar to how *BlaBlaCar* advises drivers on pricing, they could guide drivers in accepting riders, especially when the number of requests exceeds available seats. This guidance could prioritize riders based on optimal fit for the driver's route and schedule. The methodologies developed by Özkan and Ward (2020) could be adapted to refine these recommendations, ensuring that matches are not only feasible but also maximally beneficial for all parties involved.

Pricing Our findings are influenced by the research of Castillo, Knoepfle, and Weyl (2024). As we assumed above that riders are very price-sensitive, and we know that *BlaBlaCar* is able to influence the price through their markup fee. We therefore recommend implementing a dynamic pricing model similar to surge pricing. This approach could prioritize filling seats in cars that would otherwise travel with many empty seats, encouraging riders to book these options first. However, this strategy might raise concerns among drivers, especially since research by Farajallah, Hammond, and Pénard (2019) indicates that more experienced drivers, who generally offer lower prices, tend to fill up quicker, potentially leading to dissatisfaction regarding uneven earning opportunities.

More Drivers The final suggestion draws from several models discussed throughout this thesis and focuses on increasing the number of drivers, which is indirectly related to the matching process. As noted by Akbarpour et al. (2021), having a surplus of drivers is highly beneficial for ride-sharing platforms, and the same principle applies to carpooling services. Encouraging more individuals to share rides with strangers, particularly for longer journeys, could significantly enhance both the capacity and diversity of *BlaBlaCar*'s offerings. This increase in drivers would likely lead to a thicker market, which not only improves the potential for better matches but also drives down prices due to more competition among drivers.

5 Discussion

The key findings from our analysis highlight the potential advantages of implementing surge pricing, despite its generally negative public perception. Additionally, we observed that increasing the market thickness by ensuring the availability of spare drivers in virtually all areas at all times can be beneficial for ridesharing services (The LP model from Özkan and Ward (2020) is here an exception). We also discovered that combining moderate surge pricing with an efficient matching solution could enhance operational efficiency. Our analysis suggests that even relatively simple matching algorithms can achieve results close to those of more complex systems. This is particularly relevant in situations where agents lack detailed knowledge about the underlying processes and must trust the system to engage willingly. Based on these results, we identified three potential areas for improvement in BlaBlaCar's operations, which could potentially attract more customers.

However, analysis has several limitations. Firstly, the absence of empirical data from *BlaBlaCar* constrained us to base our assumptions only on publicly available information, which might not fully capture the company's objective or goals. Secondly, the models we utilized are built on strong assumptions that might not accurately reflect real-world conditions. Specifically, the assumption of time-homogeneous parameters appears unrealistic. Moreover, implementing some of the algorithms suggested by Özkan and Ward (2020) might prove to be computationally impossible. Finally, it is important to acknowledge that our recommendations for *BlaBlaCar* are derived from theoretical models that either do not align perfectly with our initial assumptions or lack rigorous technical validation.

We have identified several areas for potential future research. To our knowledge, there appears to be no existing model specifically tailored to the unique context of *BlaBlaCar*. The framework proposed by Özkan and Ward (2020), which addresses ridesharing services like *Uber* and *Lyft*, could serve as a robust foundation. However, there are small but important differences that must be taken into account.

Additionally, bridging the gap between shared economy literature and dynamic matching mechanisms would be intresting. Most existing studies within the shared economy context focus on preference optimization in static environments, such as in the work of Andersson et al. (2018). A more comprehensive model that integrates dynamic matching with individual preferences, rather than solely compatibility probabilities. Such a model could explore more complex dynamic matching models, including feedback loops like reciprocity within a dynamic framework. Moreover, conducting empirical research using data from *BlaBlaCar* could significantly enhance our understanding of dynamic matching, reciprocity, and the broader shared economy.

We believe that the results derived from our research could assist *BlaBlaCar* in improving its matching mechanism, and thus increasing the number of successful matches. This would amplify its already impressive environmental impact and at the same time produce more invaluable social interactions and benefits for society. As the shared economy continues to grow, advancements in matching mechanisms could offer novel solutions to various matching challenges, contributing to broader societal improvements.

Appendix A

Subsection A.1 Proof idea of Theorem 13

This section presents a proof sketch of theorem 13. To do this, we first provide an intuition on how one can bound the pool size of the network Z_t and then uses these bounds to derive the bounds of the loss. The ideas of the proof can be found in Akbarpour, Li, and Gharan (2020), Akbarpour (2017b) and Akbarpour (2017c).

Pool Size For the greedy matching algorithm, consider the pool size Z_t at time t. We already know that this graph is almost always empthy. Thus during a small time interval dt, the following observations can be made:

- d$t \cdot \lambda$ new agents arrive.
- d$t \cdot Z_t$ agents become critical and thus perish.

Thus, the expected pool size at time $t + \mathrm{d}t$ is given by:

$$\mathbb{E}(Z_{t+\mathrm{d}t}|Z_t) = Z_t + \underbrace{\mathrm{d}t \cdot \lambda \left(1 - \frac{d}{\lambda}\right)^{Z_t}}_{\text{unmatched new agents}} + \underbrace{\mathrm{d}t \cdot \lambda \left(1 - \left(1 - \frac{d}{\lambda}\right)^{Z_t}\right)(-1)}_{\text{matched agents}} - \underbrace{\mathrm{d}t \cdot Z_t}_{\substack{\text{critical}\\ \text{agents}\\ \text{perishing}}} + O(\mathrm{d}t^2)$$

where, the term $O(\mathrm{d}t^2)$ includes lower-order terms from the Markovian dependency, which are neglected from now on and $\left(1 - \frac{d}{\lambda}\right)^{Z_t}$ is the probability of having an egde with another agent in the graph. As this nonlinear equation is difficult to solve directly, Akbarpour, Li, and Gharan (2020) use the following approximation:

$$\left(1 - \frac{d}{\lambda}\right)^{z} \geq 1 - \frac{zd}{\lambda}$$

With some algebraic transformation, it follows that:

$$\frac{\mathrm{d}\mathbb{E}(Z_t)}{\mathrm{d}t} \geq \lambda \left[2\left(1 - \frac{\mathbb{E}(Z_t)d}{\lambda}\right) - 1\right] - \mathbb{E}(Z_t)$$

And thus we get to an lower bound of the pool size for the Greedy algorithm:

$$\mathbb{E}(Z_t) \geq \frac{\lambda}{2d + 1}$$

For the Patient algorithm it is easier to get to a lower bound of the pool size. If an algorithm does not match any agents, the pool size is pool size λ. If an algorithm match all critical agents, the pool size is $\frac{\lambda}{2}$ (an agent must be matched to another non critical agent when she becomes critical). Therefore, the pool size of the Patient algorithm must be $Z_t \in [\frac{\lambda}{2}, \lambda]$.

Loss To show the respective loss for both algorithms, we now use the bounds of the pool size and first show the loss for the Patient algorithm and then for the Greedy algorithm.

For the Patient algorithm, we know that the graph can be model by an Erdős-Rényi graph with parameter $\frac{d}{\lambda}$, the perishing rate is given by $Z_t(1-d/\lambda)^{Z_t-1}$, which represents the probability of zero matches.

Akbarpour, Li, and Gharan (2020) claim that since Z_t is sufficient to represent the network for both algorithms and that it is Markovian. Thus it converges to a unique stationary distribution with mixing time $\tau_{mix}(\epsilon) \approx \log(\lambda)\log(\frac{1}{\epsilon})$ to ϵ-total variation distance[11]. Due to its high concentration, Z_t can be assumed to be approximately $\mathbb{E}(Z_t)$. Therefore, the loss for the Patient algorithm can be approximated as:

$$\boldsymbol{L}(\text{Patient}) \approx \frac{\mathbb{E}(Z_t)\left(\frac{1-d}{\lambda}^{\mathbb{E}(Z_t)-1}\right)}{\lambda} = \frac{\text{perishing rate}}{\text{arrival rate}}$$

and given that $\mathbb{E}(Z_t) \geq \frac{\lambda}{2}$, it follows that:

$$\boldsymbol{L}(\text{Patient}) \leq \frac{1}{2} \cdot e^{-\frac{d}{2}}$$

For the Greedy algorithm, we can approximate the loss as:

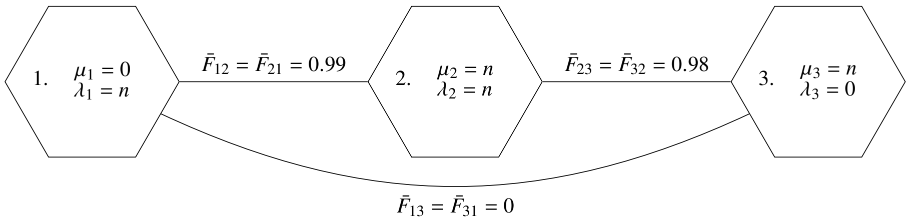
$$\boldsymbol{L}(\text{Greedy}) \approx \mathbb{E}(Z_t)\frac{1}{\lambda} = \frac{\text{perishing rate}}{\text{arrival rate}}$$

and that $\mathbb{E}(Z_t) \geq \frac{\lambda}{2d+1}$, it follows that:

$$\boldsymbol{L}(\text{Greedy}) \geq \frac{1}{2d+1}.$$

Thus we have shown using the bounds for the pool size that therom 13 holds.

Subsection A.2 Addional Figure for section 3.4.6



1. $\mu_1 = 0$ $\lambda_1 = n$
$\bar{F}_{12} = \bar{F}_{21} = 0.99$
2. $\mu_2 = n$ $\lambda_2 = n$
$\bar{F}_{23} = \bar{F}_{32} = 0.98$
3. $\mu_3 = n$ $\lambda_3 = 0$
$\bar{F}_{13} = \bar{F}_{31} = 0$

Note: Based on Fig. 3 in Özkan and Ward (2020). λ (μ) represents the arrival rate of riders (drivers). As before $\bar{F}$ represents the probability that a rider and driver are compatible. For the differnt market sizes let $n \in \{1, 10, 100\}$ for the differnt siplatzhalterlations.

Figure A.1. Graphical presentation of the siplatzhalterlation setup of Özkan and Ward (2020)

11. This means $\forall \epsilon \in \mathbb{R}_+, \exists t$ s.t. $|$ unique stationary distribution $- Z_t| < \epsilon$.

Appendix B Siplatzhalterlation

This Appendix provides the most important part of the code and the output for the results of the siplatzhalterlation in section 4.3. It also provides some context for each code block. Note that since the complte code is a bit longer, and most of it was adopted from Ranger (2021), we provide only the most important part.

Subsection B.1 Siplatzhalterlation Code

```
# Import of the most important packages/functions.
import matchingmarkets as mm
from scipy.stats import poisson
```

Code for the output presented in Table 1. Note that the code is not written in the most efficient way possible and with more emphasis on readability. The computational losses are nevertheless pretty small, since most computation is used for the actual siplatzhalterlations.

```
t = 0 # to have a nicer output
for i, j in [ # for loop over different arrival/departure times
(0.5, 1,),
(1, 1),
(1.5, 1),
(0.5, 3),
(1, 3),
(1.5, 3),
(0.5, 5),
(1, 5),
(1.5, 5),
(0.5, 10),
(1, 10),
(1.5, 10),
]:
thin_market_greedy = mm.siplatzhalterlation( # settings for market size and the greedy
                                            algorithm
runs=5,
time_per_run=35000,
max_agents=5000,
logAllData=False,
arrival_rate=i,
time_to_crit=poisson.rvs,
crit_input=j,
algorithm=mm.arbitraryMatch,
metaAlgorithm=mm.meta_greedy,
matchUtilFct=mm.utilSameType,
compatFct=mm.neighborSameType,
typeGenerator=mm.randomType,
numTypes=5,
)
t += 1
thin_market_patient = copy(thin_market_greedy)
thin_market_patient.metaAlgorithm = mm.meta_patient # change copy to patient
thin_market_greedy.run() # actual siplatzhalterlation
thin_market_patient.run()
```

```
# print output statements
print(f"\n\n THIS IS NOW ROUND {t}.\n Arrival: {i}, Departure: {j}")
print(f"\n\n {t}. GREEDY ALGORITHM RESULTS\n---Thin Market---")
thin_market_greedy.stats()
print(f"\n\n {t}. PATIENT ALGORITHM RESULTS\n---Thin Market---")
thin_market_patient.stats()
```

Code for a graphical representation for both algorithms. For us, we needed to make small changes in the settings for matplotlib, but probably it depends on the source-code editor one is using.

```
thin_market_greedy = mm.siplatzhalterlation(
runs=1,
time_per_run=10,
max_agents=5000,
logAllData=False,
arrival_rate=2,
time_to_crit=poisson.rvs,
crit_input=1,
algorithm=mm.arbitraryMatch,
metaAlgorithm=mm.meta_greedy,
matchUtilFct=mm.utilSameType,
compatFct=mm.neighborSameType,
typeGenerator=mm.randomType,
numTypes=5,
)
thin_market_patient = copy(thin_market_greedy)
thin_market_patient.metaAlgorithm = mm.meta_patient
thin_market_greedy.run()
thin_market_patient.run()

thin_market_patient.graph(plot_time=0.5)
thin_market_greedy.graph(plot_time=0.5)
```

Subsection B.2 Siplatzhalterlation Output

```
ROUND 1.
Arrival: 0.5, Departure: 1



1. GREEDY ALGORITHM RESULTS
---Thin Market---
Siplatzhalterlation Results
35000  periods
5  runs
Stat    value  (std dev)
==================================
Welfare:   480.4 ( 5.9867 )
matches:   480.4  ( 5.9867 )
perished:  4520.0  ( 5.5498 )
loss%:     0.9039  ( 0.0012 )



1. PATIENT ALGORITHM RESULTS
```

```
---Thin Market---
Siplatzhalterlation Results
35000  periods
5  runs
Stat    value  (std dev)
==================================
Welfare:   867.4 ( 13.6909 )
matches:   867.4  ( 13.6909 )
perished:  4133.4  ( 14.2632 )
loss%:     0.8265  ( 0.0028 )


ROUND 2.
Arrival: 1, Departure: 1


2. GREEDY ALGORITHM RESULTS
---Thin Market---
Siplatzhalterlation Results
35000  periods
5  runs
Stat    value  (std dev)
==================================
Welfare:   873.6 ( 32.2341 )
matches:   873.6  ( 32.2341 )
perished:  4127.0  ( 31.8245 )
loss%:     0.8253  ( 0.0064 )


2. PATIENT ALGORITHM RESULTS
---Thin Market---
Siplatzhalterlation Results
35000  periods
5  runs
Stat    value  (std dev)
==================================
Welfare:   1512.8 ( 28.2376 )
matches:   1512.8  ( 28.2376 )
perished:  3487.8  ( 28.5335 )
loss%:     0.6975  ( 0.0057 )


ROUND 3.
Arrival: 1.5, Departure: 1


3. GREEDY ALGORITHM RESULTS
---Thin Market---
Siplatzhalterlation Results
35000  periods
5  runs
Stat    value  (std dev)
==================================
Welfare:   1220.0 ( 57.2294 )
matches:   1220.0  ( 57.2294 )
```

```
perished:  3780.6  ( 57.0249 )
loss%:     0.7560  ( 0.0114 )


3. PATIENT ALGORITHM RESULTS
---Thin Market---
Siplatzhalterlation Results
35000  periods
5  runs
Stat    value  (std dev)
==================================
Welfare:   2036.2 ( 45.2875 )
matches:   2036.2  ( 45.2875 )
perished:  2965.0  ( 45.0910 )
loss%:     0.5929  ( 0.0090 )


ROUND 4.
Arrival: 0.5, Departure: 3


4. GREEDY ALGORITHM RESULTS
---Thin Market---
Siplatzhalterlation Results
35000  periods
5  runs
Stat    value  (std dev)
==================================
Welfare:   471.6 ( 26.8745 )
matches:   471.6  ( 26.8745 )
perished:  4528.8  ( 26.4832 )
loss%:     0.9057  ( 0.0054 )


4. PATIENT ALGORITHM RESULTS
---Thin Market---
Siplatzhalterlation Results
35000  periods
5  runs
Stat    value  (std dev)
==================================
Welfare:   1578.0 ( 30.2126 )
matches:   1578.0  ( 30.2126 )
perished:  3422.2  ( 30.1622 )
loss%:     0.6844  ( 0.0060 )


ROUND 5.
Arrival: 1, Departure: 3


5. GREEDY ALGORITHM RESULTS
---Thin Market---
Siplatzhalterlation Results
35000  periods
```

```
5  runs
Stat    value  (std dev)
==================================
Welfare:   860.8 ( 10.7778 )
matches:   860.8  ( 10.7778 )
perished:  4139.4  ( 11.0923 )
loss%:     0.8278  ( 0.0022 )


5. PATIENT ALGORITHM RESULTS
---Thin Market---
Siplatzhalterlation Results
35000  periods
5  runs
Stat    value  (std dev)
==================================
Welfare:   2572.4 ( 29.9105 )
matches:   2572.4  ( 29.9105 )
perished:  2428.2  ( 30.5771 )
loss%:     0.4856  ( 0.0060 )


ROUND 6.
Arrival: 1.5, Departure: 3


6. GREEDY ALGORITHM RESULTS
---Thin Market---
Siplatzhalterlation Results
35000  periods
5  runs
Stat    value  (std dev)
==================================
Welfare:   1266.0 ( 34.1057 )
matches:   1266.0  ( 34.1057 )
perished:  3735.2  ( 35.1078 )
loss%:     0.7469  ( 0.0069 )


6. PATIENT ALGORITHM RESULTS
---Thin Market---
Siplatzhalterlation Results
35000  periods
5  runs
Stat    value  (std dev)
==================================
Welfare:   3175.8 ( 20.8749 )
matches:   3175.8  ( 20.8749 )
perished:  1824.6  ( 20.8672 )
loss%:     0.3649  ( 0.0042 )


ROUND 7.
Arrival: 0.5, Departure: 5
```

```
7. GREEDY ALGORITHM RESULTS
---Thin Market---
Siplatzhalterlation Results
35000  periods
5  runs
Stat    value  (std dev)
==================================
Welfare:   464.8 ( 20.1037 )
matches:   464.8  ( 20.1037 )
perished:  4535.2  ( 20.1037 )
loss%:     0.9070  ( 0.0040 )


7. PATIENT ALGORITHM RESULTS
---Thin Market---
Siplatzhalterlation Results
35000  periods
5  runs
Stat    value  (std dev)
==================================
Welfare:   2138.4 ( 23.1655 )
matches:   2138.4  ( 23.1655 )
perished:  2861.6  ( 23.1655 )
loss%:     0.5723  ( 0.0046 )


ROUND 8.
Arrival: 1, Departure: 5


8. GREEDY ALGORITHM RESULTS
---Thin Market---
Siplatzhalterlation Results
35000  periods
5  runs
Stat    value  (std dev)
==================================
Welfare:   882.0 ( 32.0998 )
matches:   882.0  ( 32.0998 )
perished:  4118.2  ( 32.2639 )
loss%:     0.8236  ( 0.0064 )


8. PATIENT ALGORITHM RESULTS
---Thin Market---
Siplatzhalterlation Results
35000  periods
5  runs
Stat    value  (std dev)
==================================
Welfare:   3301.2 ( 22.1396 )
matches:   3301.2  ( 22.1396 )
perished:  1699.4  ( 21.8046 )
loss%:     0.3398  ( 0.0044 )
```

```
ROUND 9.
Arrival: 1.5, Departure: 5


9. GREEDY ALGORITHM RESULTS
---Thin Market---
Siplatzhalterlation Results
35000  periods
5  runs
Stat    value  (std dev)
==================================
Welfare:   1234.0 ( 17.7989 )
matches:   1234.0  ( 17.7989 )
perished:  3767.0  ( 17.8774 )
loss%:     0.7532  ( 0.0036 )


9. PATIENT ALGORITHM RESULTS
---Thin Market---
Siplatzhalterlation Results
35000  periods
5  runs
Stat    value  (std dev)
==================================
Welfare:   3890.0 ( 27.1662 )
matches:   3890.0  ( 27.1662 )
perished:  1110.8  ( 26.8879 )
loss%:     0.2221  ( 0.0054 )


ROUND 10.
Arrival: 0.5, Departure: 10


10. GREEDY ALGORITHM RESULTS
---Thin Market---
Siplatzhalterlation Results
35000  periods
5  runs
Stat    value  (std dev)
==================================
Welfare:   466.4 ( 44.2610 )
matches:   466.4  ( 44.2610 )
perished:  4533.8  ( 43.9290 )
loss%:     0.9067  ( 0.0088 )


10. PATIENT ALGORITHM RESULTS
---Thin Market---
Siplatzhalterlation Results
35000  periods
5  runs
Stat    value  (std dev)
```

```
==================================
Welfare:   3232.4 ( 52.4008 )
matches:   3232.4  ( 52.4008 )
perished:  1768.4  ( 52.5304 )
loss%:     0.3536  ( 0.0105 )


ROUND 11.
Arrival: 1, Departure: 10


11. GREEDY ALGORITHM RESULTS
---Thin Market---
Siplatzhalterlation Results
35000  periods
5  runs
Stat    value  (std dev)
==================================
Welfare:   888.8 ( 39.6101 )
matches:   888.8  ( 39.6101 )
perished:  4111.6  ( 39.7724 )
loss%:     0.8223  ( 0.0079 )


11. PATIENT ALGORITHM RESULTS
---Thin Market---
Siplatzhalterlation Results
35000  periods
5  runs
Stat    value  (std dev)
==================================
Welfare:   4291.6 ( 16.1815 )
matches:   4291.6  ( 16.1815 )
perished:  709.4  ( 15.9449 )
loss%:     0.1419  ( 0.0032 )


ROUND 12.
Arrival: 1.5, Departure: 10


12. GREEDY ALGORITHM RESULTS
---Thin Market---
Siplatzhalterlation Results
35000  periods
5  runs
Stat    value  (std dev)
==================================
Welfare:   1234.8 ( 32.1148 )
matches:   1234.8  ( 32.1148 )
perished:  3765.8  ( 31.9775 )
loss%:     0.7531  ( 0.0064 )


12. PATIENT ALGORITHM RESULTS
```

```
---Thin Market---
Siplatzhalterlation Results
35000  periods
5  runs
Stat    value  (std dev)
==================================
Welfare:   4673.8 ( 8.7040 )
matches:   4673.8  ( 8.7040 )
perished:  327.6  ( 7.5789 )
loss%:     0.0655  ( 0.0015 )
```

References

Aiko, Satomi, Ryo Itabashi, Toru Seo, Takahiko Kusakabe, and Yasuo Asakura. 2017. “Social benefit of optimal ride-share transport with given travelers’ activity patterns.” *Transportation Research Procedia* 27: 261–69. https://doi.org/https://doi.org/10.1016/j.trpro.2017.12.140. [1]

Akbarpour, Mohammad. 2017a. “Dynamic Matching Markets - Part 4 - Designing Algorithms.” Accessed May 24, 2024. https://www.youtube.com/watch?v=I_DohtDLRRM. [16]

Akbarpour, Mohammad. 2017b. “Dynamic Matching Markets - Part 6 - Proof Ideas Main Theorem.” Accessed June 22, 2024. https://www.youtube.com/watch?v=5O-HDmxWDFQ. [39]

Akbarpour, Mohammad. 2017c. “Dynamic Matching Markets - Part 7 - Continue Proof Ideas.” Accessed June 22, 2024. https://www.youtube.com/watch?v=-rUv-UxleNk. [39]

Akbarpour, Mohammad, Yeganeh Alimohammadi, Shengwu Li, and Amin Saberi. 2021. “The Value of Excess Supply in Spatial Matching Markets.” arXiv: 2104.03219. [4, 37]

Akbarpour, Mohammad, Shengwu Li, and Shayan Oveis Gharan. 2020. “Thickness and Information in Dynamic Matching Markets.” *Journal of Political Economy* 128 (3). https://doi.org/10.1086/704761. [3, 4, 12–20, 32, 35, 36, 39, 40]

Anderson, Ross, Itai Ashlagi, David Gamarnik, and Yash Kanoria. 2014. “A dynamic model of barter exchange.” In *Proceedings of the 2015 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA),* 1925–33. https://doi.org/10.1137/1.9781611973730.129. [3]

Andersson, Tommy, Ágnes Cseh, Lars H. Ehlers, and Albin Erlanson. 2018. “Organizing time banks: Lessons from matching markets.” IEHAS Discussion Papers MT-DP - 2018/18. https://hdl.handle.net/10419/222031. [38]

Ariely, Dan. 2016. “Why We Hate Uber’s Surge Pricing.” Accessed May 23, 2024. http://www.wsj.com/articles/why-we-hate-ubers-surge-pricing-1453311373. [10, 20]

Ashlagi, Itai, Afshin Nikzad, and Philipp Strack. 2022. “Matching in Dynamic Imbalanced Markets.” *Review of Economic Studies* 90 (3). https://doi.org/10.1093/restud/rdac044. [3, 20]

Azevedo, Eduardo M., and E. Glen Weyl. 2016. “Matching markets in the digital age.” *Science* 352 (6289). https://doi.org/10.1126/science.aaf7781. [3]

Banerjee, Siddhartha, Yash Kanoria, and Pengyu Qian. 2022. “Large Deviations Optimal Scheduling of Closed Queueing Networks.” arXiv: 1803.04959. [4]

Besbes, Omar, Francisco Castro, and Ilan Lobel. 2021. “Surge Pricing and Its Spatial Supply Response.” *Management Science* 67 (3). https://doi.org/10.1287/mnsc.2020.3622. [3]

BIPE, Le, Verena Butt d’Espous, and Laure Wagner. 2019. “Zero Empty Seats.” Study on behalf of BlaBlaCar, accessed June 15, 2024, https://blog.blablacar.com/newsroom/news-list/zeroemptyseats. [33]

BlaBlaCar. 2024a. "About Us." Accessed June 15, 2024. https://blog.blablacar.com/about-us. [33]

BlaBlaCar. 2024b. "Etudiants." Translated with Google Translate's Website Translator, accessed June 5, 2024, https://blog.blablacar.fr/about-us/etudiants. [1]

Buchholz, Nicholas. 2021. "Spatial Equilibrium, Search Frictions, and Dynamic Efficiency in the Taxi Industry." *Review of Economic Studies* 89 (2). https://doi.org/10.1093/restud/rdab050. [10]

Cachon, Gerard P, Kaitlin M Daniels, and Ruben Lobel. 2017. "The role of surge pricing on a service platform with self-scheduling capacity." *Manufacturing & Service Operations Management* 19 (3). https://doi.org/10.1287/msom.2017.0618. [4]

Castillo, Juan Camilo. 2023. "Who Benefits from Surge Pricing?" https://doi.org/10.2139/ssrn.3245533. [4]

Castillo, Juan Camilo, Daniel T. Knoepfle, and E. Glen Weyl. 2024. "Matching and Pricing in Ride Hailing: Wild Goose Chases and How to Solve Them," https://doi.org/10.2139/ssrn.2890666. [3–12, 22, 28, 29, 32, 37]

d'Espous, Verena Butt, Kevin Deniau, and Annie Nguyen. 2018. "Bringing People Closer." Study on behalf of BlaBlaCar, accessed June 15, 2024, https://blog.blablacar.com/wp-content/uploads/2018/01/BlaBlaCar-Bringing-People-Closer.pdf. [33]

Deutsche Welle. 2019. "Germany's taxis protest Uber plans." Accessed July 9, 2024. https://www.dw.com/en/germanys-taxi-drivers-protest-uber-deregulation-plans/a-48277628. [1]

Dickerson, John, Ariel Procaccia, and Tuomas Sandholm. 2021. "Dynamic Matching via Weighted Myopia with Application to Kidney Exchange." *Proceedings of the AAAI Conference on Artificial Intelligence* 26 (1). https://doi.org/10.1609/aaai.v26i1.8252. [3]

Dickerson, John P., Ariel D. Procaccia, and Tuomas Sandholm. 2019. "Failure-Aware Kidney Exchange." *Management Science* 65 (4). https://doi.org/10.1287/mnsc.2018.3026. [3]

Erdős, Paul, and Alfréd Rényi. 1960. "On the evolution of random graphs." *Publ. math. inst. hung. acad. sci* 5 (1). https://doi.org/10.1515/9781400841356.38. [13, 16]

Farajallah, Mehdi, Robert G. Hammond, and Thierry Pénard. 2019. "What drives pricing behavior in Peer-to-Peer markets? Evidence from the carsharing platform BlaBlaCar." *Information Economics and Policy* 48. https://doi.org/https://doi.org/10.1016/j.infoecopol.2019.01.002. [34, 37]

Feng, Guiyun, Guangwen Kong, and Zizhuo Wang. 2021. "We Are on the Way: Analysis of On-Demand Ride-Hailing Systems." *Manufacturing & Service Operations Management* 23 (5). https://doi.org/10.1287/msom.2020.0880. [4]

Gale, David, and Lloyd S Shapley. 1962. "College admissions and the stability of marriage." *American Mathematical Monthly* 69 (1): 9–15. [35]

Goncharova, Masha. 2017. "Ride-Hailing Drivers Are Slaves to the Surge." Accessed May 23, 2024. https://www.nytimes.com/2017/01/12/nyregion/uber-lyft-juno-ride-hailing.html. [10, 20]

Harrison, J. Michael. 2000. “Brownian models of open processing networks: canonical representation of workload.” *Annals of Applied Probability* 10 (1). https://doi.org/10.1214/aoap/1019737665. [3]

Ivaldi, Marc, and Emil Palikot. 2023. “Sharing when stranger equals danger: Ridesharing during Covid-19 pandemic.” *Transport Policy* 141. https://doi.org/https://doi.org/10.1016/j.tranpol.2023.07.005. [1]

Kanoria, Yash. 2023. “Dynamic Spatial Matching.” arXiv: 2105.07329. [3]

Kirk, David S., Nicolo Cavalli, and Noli Brazil. 2020. “The implications of ridehailing for risky driving and road accident injuries and fatalities.” *Social Science & Medicine* 250. Accessed June 5, 2024. https://doi.org/10.1016/j.socscimed.2020.112793. [1]

Leshno, Jacob D. 2022. “Dynamic Matching in Overloaded Waiting Lists.” *American Economic Review* 112 (12). https://doi.org/10.1257/aer.20201111. [3]

Li, Zhuoshu, Kelsey Lieberman, William Macke, Sofia Carrillo, Chien-Ju Ho, Jason Wellen, and Sanmay Das. 2019. “Incorporating Compatible Pairs in Kidney Exchange: A Dynamic Weighted Matching Model.” In *Proceedings of the 2019 ACM Conference on Economics and Computation,* 349–67. EC ’19. https://doi.org/10.1145/3328526.3329619. [4]

Liu, Tracy, Zhixi Wan, and Chenyu Yang. 2024. “Dynamic Matching on a Commuter Carpooling Platform.” The paper waspreviously circulated under the title “The Efficiency of A Dynamic Decentralized Two-Sided Matching Market”, https://doi.org/10.2139/ssrn.3339394. [12, 20]

Özkan, Erhun, and Amy R. Ward. 2020. “Dynamic Matching for Real-Time Ride Sharing.” *Stochastic Systems* 10 (1): 29–70. https://doi.org/10.1287/stsy.2019.0037. [3–5, 20–32, 35, 37, 38, 40]

Plambeck, Erica L., and Amy R. Ward. 2006. “Optimal Control of a High-Volume Assemble-to-Order System.” *Mathematics of Operations Research* 31 (3). https://doi.org/10.1287/moor.1060.0196. [3]

Ranger, Matthieu. 2021. “Dynamic matching and exogenous thickness.” Accessed June 9, 2024. https://github.com/QuantEcon/MatchingMarkets.py. [35, 41]

Shimer, Robert, and Lones Smith. 2001. “Matching, Search, and Heterogeneity.” *Topics in Macroeconomics* 1 (1). https://doi.org/doi:10.2202/1534-6013.1010. [3]

Shortle, J.F., J.M. Thompson, D. Gross, and C.M. Harris. 2018. *Fundamentals of Queueing Theory.* Wiley Series in Probability and Statistics. Wiley. https://books.google.de/books?id=9MJcDwAAQBAJ. [23, 24]

The Economist. 2016. “A fare shake.” Accessed May 23, 2024. https://www.economist.com/finance-and-economics/2016/05/14/a-fare-shake. [10, 20]

Ünver, M. Utku. 2010. “Dynamic Kidney Exchange.” *Review of Economic Studies* 77 (1). https://doi.org/10.1111/j.1467-937X.2009.00575.x. [3]

Yan, Chiwei, Helin Zhu, Nikita Korolko, and Dawn Woodard. 2020. “Dynamic pricing and matching in ride-hailing platforms.” *Naval Research Logistics (NRL)* 67 (8). https://doi.org/https://doi.org/10.1002/nav.21872. [12]

Yu, Biying, Ye Ma, Meimei Xue, Baojun Tang, Bin Wang, Jinyue Yan, and Yi-Ming Wei. 2017. "Environmental benefits from ridesharing: A case of Beijing." *Applied Energy* 191: 141–52. https://doi.org/https://doi.org/10.1016/j.apenergy.2017.01.052. [1]

Zhong, Yueyang, Zhixi Wan, and Zuo-Jun Shen. 2020. "Queueing Versus Surge Pricing Mechanism: Efficiency, Equity, and Consumer Welfare." https://doi.org/10.2139/ssrn.3699134. [23]