On the effect of batching for on-demand high-capacity micro-transit services

Lynn Fayed

Gustav Nilsson

Nikolas Geroliminis

STRC conference paper 2024 **May 2, 2024**

STRC | **24th Swiss Transport Research Conference**
Monte Verità / Ascona, May 15-17, 2024

On the effect of batching for on-demand high-capacity micro-transit services

Lynn Fayed
Urban Transport Systems Laboratory (LUTS)
École polytechnique fédérale de Lausanne (EPFL)
lynn.fayed@epfl.ch

Gustav Nilsson
Urban Transport Systems Laboratory (LUTS)
École polytechnique fédérale de Lausanne (EPFL)
gustav.nilsson@epfl.ch

Nikolas Geroliminis
Urban Transport Systems Laboratory (LUTS)
École polytechnique fédérale de Lausanne (EPFL)
nikolas.geroliminis@epfl.ch

May 2, 2024

Abstract

The surge of on-demand ride-hailing services provides network users with a convenient and flexible transportation alternative. While ride-hailing and two-passenger ride-splitting services have been extensively studied, much less is known about the efficiency and operation of high-capacity on-demand services. In this work, we aim to improve our understanding of the operation of these services through casting them into a queuing theory framework. We do so for the purpose of theoretically investigating the impact of batching on the micro-transit service level. Given the fast-growing complexity of this model with the network structure, we resort to a micro-simulation environment to reproduce the deployment of on-demand micro-transit in large-scale networks. In this simulation framework, we analyze the request waiting time and request detour under different batching strategies and assess the service performance compared to a simple greedy vehicle-request matching. The results show that batching has significant potential to improve the passenger detour, yet the waiting time is rarely reduced when batching is involved.

Keywords

batching; dynamic matching; on-demand micro-transit

Contents

List of Tables

List of Figures

1 Introduction

Ride-hailing travel alternatives have quickly gained momentum among network users because of their potential to provide a flexible door-to-door service with affordable fares. However, their negative externalities surfaced shortly after, in particular with respect to their traffic impact to passenger serviceability ratio as idling ride-hailing vehicles are shown to significantly deteriorate traffic conditions without efficiently delivering any passengers in urban areas Tirachini and Gómez-Lobo (2019); Beojone and Geroliminis (2021). Ride-splitting services partially mitigate the unfavorable outcomes of ride-hailing because they can pair passengers with close origins and destinations within the same route, therefore serving more requests with a lower total VKT and a better service level compared to fixed bus routes. However, their operational features are not yet clearly defined, especially for high-capacity on-demand services where the number of trip-sharing passengers exceeds two. Despite its potential to be a very efficient transportation option, our understanding of micro-transit is still underdeveloped to the best of our knowledge. Consequently, examining its characteristics is a crucial step for identifying its market potential or operational efficiency in multi-modal networks.

The macroscopic modeling area for ride-hailing or two-passenger ride-splitting services is well-established as we have observed in Ke *et al.* (2020); Zhang and Nie (2021). The objective of their work was to come up with adequate pricing schemes under static equilibrium settings, including settings where idle ride-hailing vehicles are subject to congestion tolls. As for the dynamic realm, an extensive body of literature focused on designing fast and efficient real-time exact methods Alonso-Mora *et al.* (2017), heuristics Santos and Xavier (2013); Jung *et al.* (2016), or meta-heuristic Ali *et al.* (2019), to achieve a high-quality matching and routing for high-capacity dynamic micro-transit services on the microscopic level. The main challenge here is to properly deal with the dynamic aspect of the problem, i.e., the fact that an optimal vehicle-request match can potentially become suboptimal with the arrival of new requests. This myopic assignment problem has been addressed in Ke *et al.* (2022), where the matching of a request is delayed if a historically better assignment can be found. Service pricing under time-varying demand for ride-hailing or ride-splitting is accounted for in Nourinejad and Ramezani (2020); Fayed *et al.* (2023), where the goal was to use macroscopic traffic and service modelling functions to assess service price variation in multi-modal settings. The advantage of this aggregate modelling framework is that it circumvents the need to bind the pricing optimization framework to an exhaustive low-level vehicle-passenger matching module.

Reproducing this same analysis for high-capacity on-demand micro-transit services is

hindered by the lack of a comprehensive high-level understanding of the structure and operation of these services. Therefore, The main contributions of this work are twofold. First, we carve the problem of high-capacity on-demand micro-transit services into a queuing theory framework, and show some analytical findings on the usefulness of batching for these services on a small scale. Second, on a larger scale, we use a simulation framework for the city of Glyfada in Greece to showcase the effect of an event- and time-based batching on the waiting time, detour time, and passenger acceptance rate.

2 Methodology

In the following section, we develop a macroscopic framework for the modelling of on-demand micro-transit vehicles rooted in a queuing theory approach, and we use this model to assess whether for simple network graphs, batching is beneficial for improving micro-transit service level.

2.1 A macroscopic modelling framework for micro-transit services

In this work, we model an on-demand micro-transit service with a fixed number of micro-transit vehicles N, each with a capacity C such that $N, C \in \mathbb{Z}^+$. The micro-transit vehicles operate in a network defined as a complete graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ such that $\mathcal{V}$ is the set of station nodes where micro-transit vehicles stop to board or alight passengers, and $\mathcal{E}$ is the set of edges capturing the transit time between stations. We note that the pick-up and drop-off operations only happen at these nodes as the service we assess is station-based rather than door-to-door. We denote by $T \in \mathbb{R}_{\geq 0}^{\mathcal{V}\times\mathcal{V}}$ the matrix whose elements T_{ij} defines the travel time on the edge e_{ij}, where $e_{ij} \in \mathcal{E}$ is the edge between two stations i and j, $i, j \in \mathcal{V}$. The travel time between any two stations is strictly positive. As a consequence, together with the completeness assumption of the graph, for all $i, j \in \mathcal{V}$, $T_{ij} = 0$ if and only if $i = j$. We also make the natural assumption that the matrix T satisfies triangular inequality such that $T_{ij} \leq T_{ik} + T_{kj}$ for $i, j, k \in \mathcal{V}$. The arrival of passenger requests covers a time period $\xi > 0$, and the average hourly demand rate between any two pair of nodes is given by $\lambda \in \mathbb{R}_{\geq 0}^{\mathcal{V}\times\mathcal{V}}$ such that λ_{ij} defines the demand rate between any two stations $i, j \in \mathcal{V}$.

Figure 1: A two-node graph structure for the analysis of on-demand micro-transit

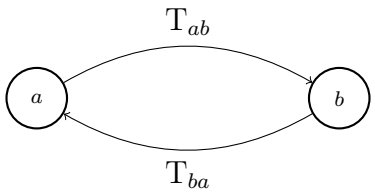
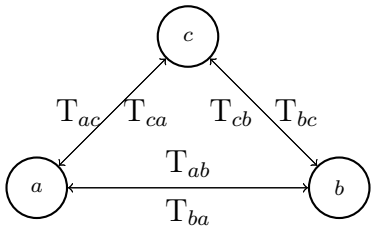


Figure 2: A three-node graph structure for the analysis of on-demand micro-transit



2.2 Theoretical analysis on the effect of batching in simple settings

2.2.1 Two-node example

Consider the two-node graph displayed in Figure 1 and a single vehicle only delivering passengers from station a to b such that $\lambda_{ab} > 0$ and is equal to 0 otherwise. We denote by β the batching time window and we set $\alpha = \lfloor \frac{\xi}{2T_{ab}+\beta} \rfloor$. We furthermore make the assumption that the demand is time-invariant and that $T_{ab} = T_{ba}$.

Proposition 1 *For a two-station network with a single vehicle and constant demand λ_{ab} and $\alpha > 1$, the total delay over a duration ξ is minimized for $\beta = 0$ when $\lambda_{ab}(2T_{ab}+\beta) \leq C$, i.e., when all passengers are getting served.*

Proof When $\lambda_{ab}(2T_{ab} + \beta) \leq C$, the cumulative waiting time δ for all requests before being served, expressed in request.hr, is given by

$$\delta = \frac{1}{2}\lambda_{ab}\alpha(2T_{ab} + \beta)^2 + \frac{1}{2}\lambda_{ab}(\xi - \alpha(2T_{ab} + \beta))^2 \tag{1}$$

Since $\alpha = \lfloor \frac{\xi}{2T_{ab}+\beta} \rfloor$, then there exists ϵ such that if $0 \leq \epsilon < 1$, we have $\alpha = \frac{\xi}{2T_{ab}+\beta} - \epsilon$.

Replacing α with $\frac{\xi}{2T_{ab}+\beta} - \epsilon$ in (1), we get that

$$\delta = \frac{1}{2}\lambda_{ab}\left(\frac{\xi}{2T_{ab}+\beta} - \epsilon\right)(2T_{ab}+\beta)^2 + \frac{1}{2}\lambda_{ab}\left(\xi - \left(\frac{\xi}{2T_{ab}+\beta} - \epsilon\right)(2T_{ab}+\beta)\right)^2 = \frac{1}{2}\lambda_{ab}\xi(2T_{ab}+\beta) - \frac{1}{2}\lambda_{ab}\epsilon(2T_{ab}+\beta)^2 + \frac{1}{2}\lambda_{ab}\epsilon^2(2T_{ab}+\beta)^2$$

Now, let

$$f(\beta) = \xi(2T_{ab}+\beta) + (\epsilon^2 - \epsilon)(2T_{ab}+\beta)^2 .$$

Clearly,

$$\frac{df}{d\beta} = \xi + 2(\epsilon^2 - \epsilon)(2T_{ab}+\beta) .$$

Hence, for $f(\beta)$ to be decreasing, it must hold that

$$\xi + 2(\epsilon^2 - \epsilon)(2T_{ab}+\beta) < 0$$

which is equivalent to

$$\alpha + \epsilon < 2 - 2\epsilon^2$$

or

$$\alpha < \epsilon - 2\epsilon^2 < 1$$

which contradicts the fact that $\alpha > 1$.

2.2.2 Three-node example

Let us now consider the three-node graph shown in Figure 2. For the scenario under consideration, we only have demand going from c to a or c to b such that $\lambda_{ca} > 0$ and $\lambda_{cb} > 0$. Also, to simplify the analysis, we set the vehicle capacity C to ∞ which is an assumption we will relax in future work. The micro-transit vehicle always serves all the passengers waiting at node c since we assume that $T_{cb} + T_{ba} + T_{ac} < T_{cb} + T_{bc} + T_{ca} + T_{ac}$

and $T_{ca} + T_{ab} + T_{bc} < T_{cb} + T_{bc} + T_{ca} + T_{ac}$.

Let R_1 and R_2 be the route *cba* and *cab* respectively with travel time $T_1 = T_{cb} + T_{ba} + T_{ac}$ and $T_2 = T_{ca} + T_{ab} + T_{bc}$. The detour for passengers going from c to b that we denote by $\Delta_{cb}^{R_2} = T_{ca} + T_{ab} - T_{cb}$ if R_2 is chosen, and $\Delta_{cb}^{R_1} = 0$ otherwise. Similarly, the detour for passengers going from c to a is $\Delta_{ca}^{R_1} = T_{cb} + T_{ba} - T_{ca}$ if R_1 is chosen, and $\Delta_{ca}^{R_2} = 0$ otherwise. Furthermore, we assume that the micro-transit vehicle always takes the route that minimizes passengers' detour.

Proposition 2 *For a three-station network with a single vehicle and constant demand λ_{ca} and λ_{cb}, the total delay over a duration ξ is minimized for $\beta = 0$.*

Proof Under the assumption that the micro-transit vehicle chooses the route with the minimal passenger detour, then at any time t_k, the route R_{π_k} chosen by the vehicle, $\pi_k \in \{1, 2\}$ for $1 \leq k \leq \alpha$ is such that π_k is given by

$$\pi_k = \begin{cases} 1 & \text{if } \lambda_{cb}(t_k - t_{k-1})(T_{ca} + T_{ab} - T_{cb}) < \lambda_{ca}(t_k - t_{k-1})(T_{cb} + T_{ba} - T_{ca}), \\ 2 & \text{otherwise.} \end{cases}$$

Knowing π_k, the the service time for the next time interval $t_{k+1} = t_k + T_{\pi_k} + \beta$. The cumulative number of serviced requests at any time t_k is

$$S_c(t_k) = (\lambda_{ca} + \lambda_{cb})(t_k - t_{k-1}),$$

for $1 \leq k \leq \alpha$. The total delays, which are in this case a sum between the waiting and detour time, are given by

$$\delta = \xi(\lambda_{ca} + \lambda_{cb}) - \left[\sum_{k=1}^{\alpha-1}(t_{k+1} - t_k)S_c(t_k) + (\xi - t_\alpha)S_c(t_\alpha)\right]$$
$$+ \sum_{k=1}^{\alpha}(t_k - t_{k-1})(\Delta_{ca}^{R_{\pi_k}}\lambda_{ca} + \Delta_{cb}^{R_{\pi_k}}\lambda_{cb}) + (\xi - t_\alpha)(\Delta_{ca}^{R_{\pi_{\alpha+1}}}\lambda_{ca} + \Delta_{cb}^{R_{\pi_{\alpha+1}}}\lambda_{cb}),$$

where α in this case is the last integer such that $\xi - t_\alpha < T_{\pi_\alpha}$.

Without loss of generality, let us assume that $\lambda_{cb} > \lambda_{ca}$, then the vehicle will always serve route R_1 since $\pi_k = 1$ and $t_k = k(T_1 + \beta)$ for $1 \leq k \leq \alpha$. Clearly, the waiting time formulation is similar to what we observed in Proposition 1, and therefore is minimized

for $\beta = 0$. As for the detour, we that $\Delta_{cb}^{R_1} = 0$, and we are therefore left with $\xi \Delta_{cb}^{R_1} \lambda_{ca}$, which is independent of β.

3 Numerical Case Study

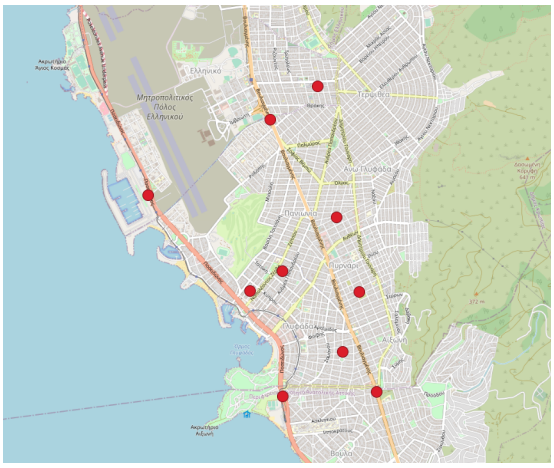
The theoretical framework that we developed so far quickly increases in complexity when the number of stations grows. This justifies the need to perform large-scale simulations to validate whether our conclusion on the effect of batching scales well in more realistic networks. As a result, we perform in the following section a numerical study in a simulation environment to replicate real network settings. Given the flexibility of the simulation environment, we furthermore study instances with time-varying demand to identify whether batching is potentially useful under non-uniform demand profiles.

3.1 Simulation settings

In the following part, we elaborate on the main characteristics of the generic simulation environment that we developed, and we illustrate the main outcomes of our simulations. In particular, we consider the city of Glyfada in Greece, shown in Figure 3. We design a service of variable fleet N of micro-transit vehicles with a uniform passenger capacity C. The network under consideration has 10 stations distributed over the center and the city's outskirts. The total arrival of requests happens over a span of $\xi = 3$ hr. The simulation time itself can potentially extend beyond ξ as long as micro-transit vehicles continue to pick up and drop off pre-assigned requests. Assuming that the hourly micro-transit demand is uniform for all trips, with demand rate $\bar{\lambda}$, then the matrix $\lambda = \frac{\bar{\lambda}}{|\mathcal{V}|(|\mathcal{V}|-1)}(\mathbf{1}^T\mathbf{1} - I)$. The variable $n(t)$, which defines the number of passengers having arrived at every node during a period $[0, t]$, $t < D$, follows a Poisson process such that $n(t) \sim \text{Poisson}(\bar{\lambda}t)$. The origin O and the destination D of arriving requests are uniformly generated from the set of stations $\mathcal{V}$ such that $O, D \in \mathcal{V}$.

With regard to the vehicle-passenger matching and vehicle dispatching, we implement an efficient and fast algorithm within the simulation to achieve desired objectives within reasonable time frames. For this reason, we design a straightforward greedy matching with the aim of minimizing a combined objective function that consists of the waiting time of

Figure 3: Glyfada network with the station locations



requests and the detour time of traveling passengers. For every request assignment, and every potential vehicle schedule, we make sure that the output solution is feasible, i.e., it abides by the capacity, maximum waiting time, and maximum detour time constraints. The baseline greedy matching algorithm is instantaneous and assigns the first coming request to the feasible vehicle route with the least amount of wait and detour cost increment, and this assignment is definitive in the sense that any changes in the assignment are forbidden despite the new arrival of requests. A summary of the greedy algorithm logic that we utilize in our simulations is displayed in Algorithm 1.

Algorithm 1 Greedy matching and routing algorithm

```
procedure GREEDY(vehicles, O, D)  ▷ For a copy of the vehicles structure, find the best vehicle for a given OD pair
    best_vehicle ← ∅, best_cost ← +∞
    for vehicle ∈ vehicles do
        origin_indices ← enumerate_origin_index(vehicle.route)
        for i_O ∈ origin_indices do
            destination_indices ← enumerate_destination_index(i_O, vehicle.route)
            for i_D ∈ destination_indices do
                vehicle.route.insert_stop(O, i_O)
                vehicle.route.insert_stop(D, i_D)
                if feasible(vehicle) and cost(vehicle) < best_cost then
                    best_vehicle ← vehicle, best_cost ← cost(vehicle)
                end if
            end for
        end for
    end for
    return best_vehicle
end procedure
```

Upon the arrival of a request with an origin station O and destination station D, Algorithm 1 iterates over the set of operational vehicles, and the functions `enumerate_origin_index` and `enumerate_destination_index` return the comprehensive set of origin and destination indices i_O and i_D respectively, always ensuring that the destination index follows the origin index in a vehicle route. Next, we insert the stations in the vehicle route according to their candidate positional index, and check for violations of capacity constraints. If the route is feasible, and the cost increment in terms of additional detour and waiting times incurred is the lowest, the current vehicle and its respective route becomes the best candidate assignment. The algorithm continues running up until it covers all available vehicles and their respective routes.

Note that so far, this algorithm deals with instantaneous request arrivals. To integrate batching into our framework, we implement two batching strategies: a time-based batching, and an event-based batching. The difference between the two is that the time-based batching accumulates the requests over a fixed time interval, and the assignment of requests to vehicles is triggered once this time has elapsed. On the contrary, the event-based batching triggers the request to vehicle assignment once a pre-defined number of requests have been accumulated. Whether prompted by a time-based or event-based batching, the assignment of the queued requests to available vehicles is done through repeatedly calling Algorithm 1, but batching guarantees that the iteration over all requests in the batch is completed before proceeding with the rest of the simulation. Moreover, instead of assigning the requests based on a first come first served basis, we shuffle the queue of requests multiple times, and keep the solution with the lowest objective function value. Next, we elucidate the results of our simulations for two different scenarios, one with fixed demand rate $\bar{\lambda}$, and another scenario for a variable demand rate $\bar{\lambda}$.

3.2 Numerical results

In the following part, we display the different simulation outputs for cases with constant and time-varying demand. Although the theoretical analysis we previously performed applies only for fixed demand, the simulation framework is flexible such as it allows to assess the efficiency of batching irrespective of the demand profile.

3.2.1 Cases with constant demand

For the purpose of showing the influence of batching on the performance of the on-demand micro-transit service, we assess the case for two different fleet sizes, $N = 8$ and $N = 10$ vehicles. The demand rate $\bar{\lambda}$ for this part of the results is set to 180 requests/hr. Moreover, we consider the cases when the waiting time constraint is equal to 10 min, but also when the waiting time is unconstrained. Irrespective of the scenario under consideration, we always set the detour time constraints to infinity. We point out here that for the time- and event-based situations, we perform the simulations for different batching time windows and various number of requests for the time-based and event-based batching, respectively. However, we only report the values for the instances returning the lowest average waiting time for the accepted requests. Moreover, for demonstrative purposes, we display the results for the greedy time-based and greedy event-based to observe the effect of solely delaying the request assignment without any attempt at reoptimizating through reshuffling. When batching is performed but the assignment is based on a first-come first-served basis (FCFS), we denote these scenarios by FCFS time-based and FCFS event-based. When the assignment however is performed after a batching round with reshuffling, we denote these scenarios by Shuffle time-based, and Shuffle event-based. Every value reported in this section represents the average over 10 simulation runs. Also, for the optimization framework, we randomly shuffle the requests 50 times before selecting the best solution.

Tables 1 and 2 summarize the simulation results for a fleet of 8 vehicles under a constrained versus unconstrained waiting time scenarios respectively. Clearly, when the maximum waiting time constraint is set to infinity, we accept all requests at the expense of a higher wait delays. The values of detour are however smaller, indicating that a higher trip efficiency is achieved when the number of serviced users is larger. When comparing the different scenarios in each table independently, it is clear that time-based or event-based batching does not significantly improve waiting time, yet it reduces the detour and increases the acceptance rate in constrained settings. This improvement is less significant in the unconstrained settings. Similar results are observed when the fleet size is set to 10 in Tables 3 and 4. Moreover, a larger fleet size results in a lower waiting and detour time, and a higher acceptance rate due to the increase of the overall service capacity. Nevertheless, the results here reproduce what we observed with the scenarios of $N = 8$. The minor difference however, is that batching scenarios have relatively worse waiting times compared to the instantaneous greedy assignment scenario. In unconstrained settings, the need for batching is actually unsubstantiated, given that the greedy assignment is returning good enough solutions.

Table 1: Simulation results with constant demands for $N = 8$ and waiting time constraints

Scenario	**Waiting [hr]**	**Detour [hr]**	**Acceptance**
Greedy	0.110	0.205	0.639
FCFS time-based	0.113	0.198	0.639
FCFS event-based	0.115	0.204	0.630
Shuffle time-based	0.106	0.142	0.680
Shuffle event-based	0.106	0.152	0.691

Table 2: Simulation results with constant demands for $N = 8$ and no waiting time constraints

Scenario	**Waiting [hr]**	**Detour [hr]**	**Acceptance**
Greedy	0.259	0.078	1
FCFS time-based	0.259	0.077	1
FCFS event-based	0.262	0.076	1
Shuffle time-based	0.247	0.073	1
Shuffle event-based	0.244	0.072	1

3.2.2 Cases with time-varying demand

Batching has not shown to improve the waiting under constant demand rates, whether in the simulation environment or the analytical framework. Motivated by this, we analyze the output of the simulations for the time-varying case to verify whether a non-uniform demand profile justifies the need for request batching. For this case, therefore, we assume that for every 30 request arrivals, we regenerate the demand rate $\bar{\lambda}$ such that $\bar{\lambda} = 100$ requests/hr for 80% of the time and $\bar{\lambda} = 450$ requests/hr for the remaining time. Elsewise, all the other parameters are kept the same as in the case with constant demand. The results for a fleet size of 8 for a constrained and unconstrained waiting times are displayed in Tables 5 and 6, while the results for a fleet of 10 are shown in Tables 7 and 8. The conclusions that we reach here are similar to those observed in the case of constant demand. Nevertheless, we point out that a more accentuated difference was observed in

Table 3: Simulation results with constant demands for $N = 10$ and waiting time constraints

Scenario	**Waiting [hr]**	**Detour [hr]**	**Acceptance**
Greedy	0.102	0.173	0.825
FCFS time-based	0.104	0.177	0.798
FCFS event-based	0.108	0.172	0.806
Shuffle time-based	0.099	0.127	0.845
Shuffle event-based	0.100	0.119	0.859

Table 4: Simulation with constant demands results for $N = 10$ and no waiting time constraints

Scenario	**Waiting [hr]**	**Detour [hr]**	**Acceptance**
Greedy	0.136	0.057	1
FCFS time-based	0.142	0.060	1
FCFS event-based	0.155	0.058	1
Shuffle time-based	0.143	0.057	1
Shuffle event-based	0.145	0.056	1

Table 5: Simulation results with time-varying demands for $N = 8$ and waiting time constraints

Scenario	**Waiting [hr]**	**Detour [hr]**	**Acceptance**
Greedy	0.109	0.202	0.658
FCFS time-based	0.110	0.201	0.649
FCFS event-based	0.112	0.191	0.655
Shuffle time-based	0.106	0.154	0.714
Shuffle event-based	0.105	0.150	0.721

Table 3 compared to Table 7 between the two scenarios Shuffle time-based and Shuffle event-based, indicating that the event-based scenario performs better under time-varying demand. This finding however requires further justification in other instances with various demand profiles.

Table 6: Simulation results with time-varying demands for $N = 8$ and no waiting time constraints

Scenario	**Waiting [hr]**	**Detour [hr]**	**Acceptance**
Greedy	0.230	0.074	1
FCFS time-based	0.216	0.073	1
FCFS event-based	0.260	0.077	1
Shuffle time-based	0.226	0.069	1
Shuffle event-based	0.234	0.070	1

Table 7: Simulation results with time-varying demands for $N = 10$ and waiting time constraints

Scenario	**Waiting [hr]**	**Detour [hr]**	**Acceptance**
Greedy	0.102	0.166	0.823
FCFS time-based	0.099	0.159	0.863
FCFS event-based	0.107	0.157	0.801
Shuffle time-based	0.098	0.145	0.881
Shuffle event-based	0.101	0.124	0.838

Table 8: Simulation results with time-varying demands for $N = 10$ and no waiting time constraints

Scenario	**Waiting [hr]**	**Detour [hr]**	**Acceptance**
Greedy	0.129	0.057	1
FCFS time-based	0.145	0.058	1
FCFS event-based	0.161	0.060	1
Shuffle time-based	0.144	0.056	1
Shuffle event-based	0.150	0.056	1

4 Conclusions

In this work, we focused on investigating the influence of request batching on the service level for on-demand high-capacity micro-transit services. We modeled the problem structure within a queuing theory framework and analyzed the effect of request batching on total delays for simple network graph with constant demand. We then tested our findings in a more realistic simulation environment with a larger service scale with uniform and time-dependent demand profiles. In both approaches, we concluded that batching has little improvement in waiting time but has the potential to reduce the passenger detour.

In the future, we plan to further develop our analytical findings to handle more complex scenarios. Furthermore, we aim to come up with comprehensive scaling laws for the structure and operation of high-capacity micro-transit under different exogenous factors shaping this particular service. The ultimate goal is to integrate the aggregate relationships we uncover to investigate the pricing of on-demand micro-transit in multi-modal networks.

5 References

Ali, A., B. Kaltenhäuser, I. Gaponova and K. Bogenberger (2019) Asynchronous adaptive large neighborhood search algorithm for dynamic matching problem in ride hailing services, 3006–3012, 10 2019.

Alonso-Mora, J., S. Samaranayake, A. Wallar, E. Frazzoli and D. Rus (2017) On-demand high-capacity ride-sharing via dynamic trip-vehicle assignment, *Proceedings of the National Academy of Sciences*, **114**, 201611675.

Beojone, C. V. and N. Geroliminis (2021) On the inefficiency of ride-sourcing services towards urban congestion, *Transportation Research Part C: Emerging Technologies*, **124**, 102890, ISSN 0968-090X.

Fayed, L., G. Nilsson and N. Geroliminis (2023) A macroscopic modelling framework for the dynamic pricing of pool ride-splitting vehicles in bus lanes. Accepted to IEEE 26th International Conference on Intelligent Transportation Systems (ITSC) 2023.`https://www.dropbox.com/s/a45wvuzp24m06d9/ITSC2023_0642_MS.pdf?dl=0`.

Jung, J., R. Jayakrishnan and J. Y. Park (2016) Dynamic shared-taxi dispatch algorithm with hybrid-simulated annealing, *Computer-Aided Civil and Infrastructure Engineering*, **31** (4) 275–291.

Ke, J., F. Xiao, H. Yang and J. Ye (2022) Learning to delay in ride-sourcing systems: A multi-agent deep reinforcement learning framework, *IEEE Transactions on Knowledge and Data Engineering*, **34** (5) 2280–2292.

Ke, J., H. Yang, X. Li, H. Wang and J. Ye (2020) Pricing and equilibrium in on-demand ride-pooling markets, *Transportation Research Part B: Methodological*, **139**, 411–431.

Nourinejad, M. and M. Ramezani (2020) Ride-sourcing modeling and pricing in non-equilibrium two-sided markets, *Transportation Research Part B: Methodological*, **132**, 340–357, ISSN 0191-2615. 23rd International Symposium on Transportation and Traffic Theory (ISTTT 23).

Santos, D. O. and E. C. Xavier (2013) Dynamic taxi and ridesharing: A framework and heuristics for the optimization problem, paper presented at the *Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence*, IJCAI '13, 2885–2891, ISBN 9781577356332.

Tirachini, A. and A. Gómez-Lobo (2019) Does ride-hailing increase or decrease vehicle kilometers traveled (VKT)? A simulation approach for Santiago de Chile, *International Journal of Sustainable Transportation*, **14**, 1–18.

Zhang, K. and Y. M. Nie (2021) To pool or not to pool: Equilibrium, pricing and regulation, *Transportation Research Part B: Methodological*, **151**, 59–90, ISSN 0191-2615.