Aalto-yliopisto
Perustieteiden
korkeakoulu

Scheduling in a ride-sharing scenario (results presentation)

Aapo Laakkio

16.9.2026

Advisor: *Fernando Dias*

Supervisor: *Fernando Dias*

Työn saa tallentaa ja julkistaa Aalto-yliopiston avoimilla verkkosivuilla. Muilta osin kaikki oikeudet pidätetään.

Aalto-yliopisto
Perustieteiden
korkeakoulu

The problem

- **Two groups to connect**
- Customer accepts only some drivers
  - ➢ age, gender, vehicle, trust
- Driver works a fixed shift
- Must decide who drives whom, and in what order and when
  - ➢ decisions constrain each other
- Customer wants a fast pickup, driver wants a full shift
  - ➢ conflicting goals
- Hundreds of customers waiting, tens of drivers and vast amount of locations



Why?

- Matching and scheduling are both combinatorial
  - Exponential in problem size
- Real platforms must answer in minutes or even seconds, for possibly thousands customers

- **Solution**: split into smaller subproblems and accept a non-optimal solution
- A solution is worthless if either side refuses it
  - trust, convenience and incentives decide participation
- Preferences are usually left to the users to screen

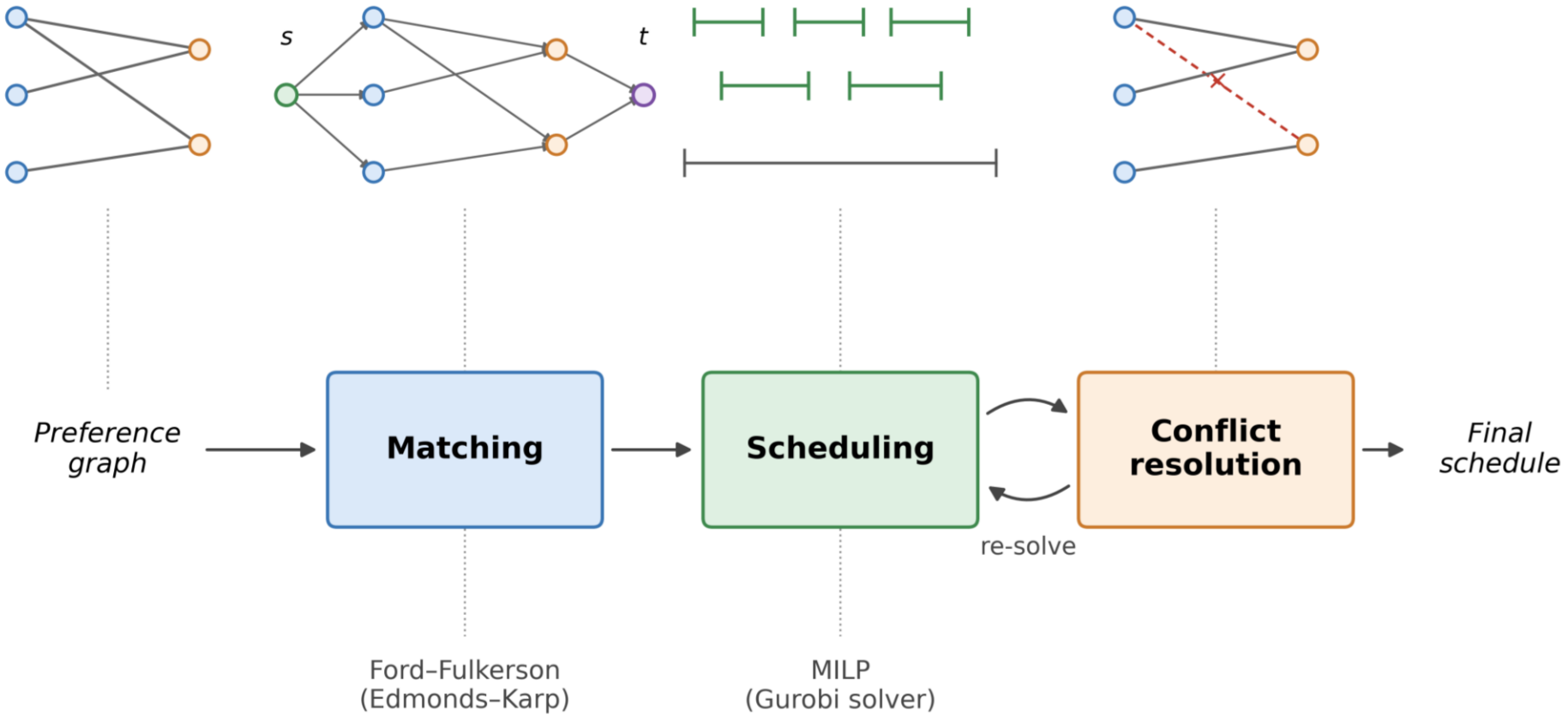
- Technology is no longer the constraint
  - For example: carpool share of US commuters fell from 20.4 % (1970) to 10.7 % (2008) despite ~638 matching programs (Chan and Shaheen, 2012)
- The open question is **whom to connect with whom**

Objectives

- Determine can the ride-sharing problem be decomposed into a multi-driver problem solved as many **single-driver** problems?

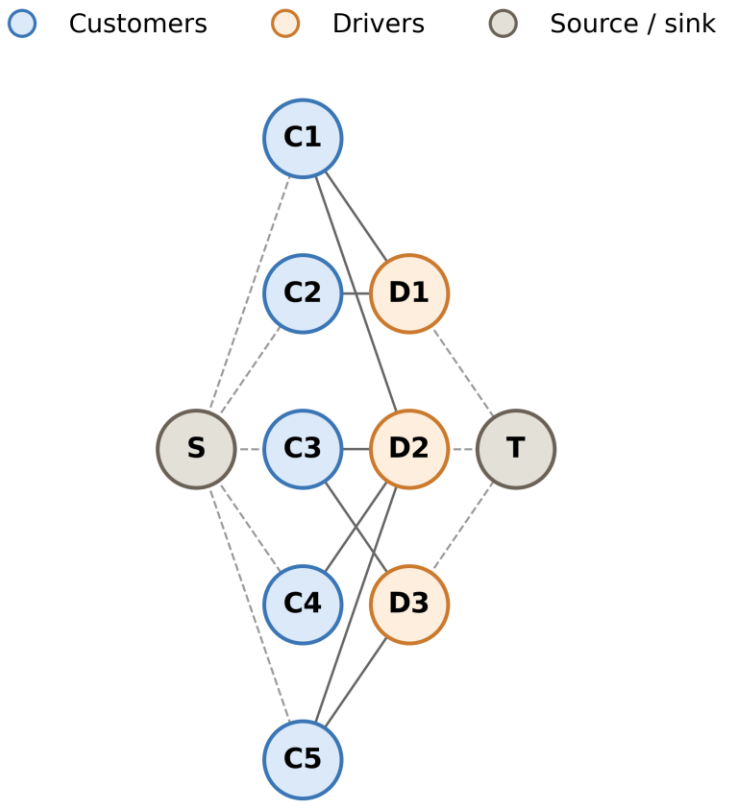
- Three subproblems to solve
  - Match customers to drivers based on preferences (Matching)
  - Schedule each driver independently (Scheduling)
  - Resolve conflicts between drivers (Conflict resolution)

Pipeline


s
t
Preference graph
Matching
Scheduling
Conflict resolution
Final schedule
re-solve
Ford–Fulkerson (Edmonds–Karp)
MILP (Gurobi solver)

Matching

- A **bipartite** graph is built using the preferences of customers
- Each customer is allowed to match to at most 5 drivers
  - Drivers have no maximum
- **Maximum flow** is calculated using Ford-Fulkerson algorithm (Ford and Fulkerson, 1957)
  - Edmonds-Karp variant (Edmonds and Karp, 1972)



Scheduling 1/2

- Feasible time slots calculated **before** optimizing

- Objective
  - **Delay term**: minutes late versus the customer's wanted pickup time
  - **Penalty term**: 300 per unserved customer
  - **Idle term**: shift length minus time actually driving

- Constraints
  - **Assign**: each customer starts at most one ride
    - Leaving them out is allowed, costs the penalty
  - **Overlap**: one ride per time slot
    - Breaks between each ride to simulate travel
  - **Successor**: a ride must be followed within the maximum break, unless the shift is nearly over
    - No idle gaps
  - **Integrality**: rides cannot be split

Scheduling 2/2

Scheduling ILP

$$\begin{aligned}\min \quad & D \sum_{i \in A_k} \sum_{t \in T_i} (t \cdot \Delta t - U_i) x_{i,t} \\ + \quad & P \sum_{i \in A_k} (1 - \sum_{t \in T_i} x_{i,t}) \\ + \quad & I((H_k - S_k) - \sum_{i \in A_k} \sum_{t \in T_i} R_i x_{i,t})\end{aligned}$$

Feasible time slots

$$\begin{aligned} t \cdot \Delta t &\geq \max(U_i, S_k), \\ t \cdot \Delta t + R_i &\leq H_k, \\ t \cdot \Delta t - U_i &\leq E_{\max}. \end{aligned}$$

$$\text{s.t.} \quad \sum_{t \in T_i} x_{i,t} \leq 1, \qquad \forall i \in A_k,$$

$$\sum_{\substack{i \in A_k, t \in T_i \\ t \leq \tau < t + r_i + b}} x_{i,t} \leq 1, \qquad \forall \text{ slot } \tau,$$

$$x_{i,t} \leq \sum_{(j,t') \in N_{i,t}} x_{j,t'}, \qquad \forall (i,t) \text{ s.t. } e_{i,t} + B_{\max} < H_k,$$

$$x_{i,t} \in \{0,1\}, \qquad \forall i \in A_k,\ t \in T_i.$$

Conflict resolution

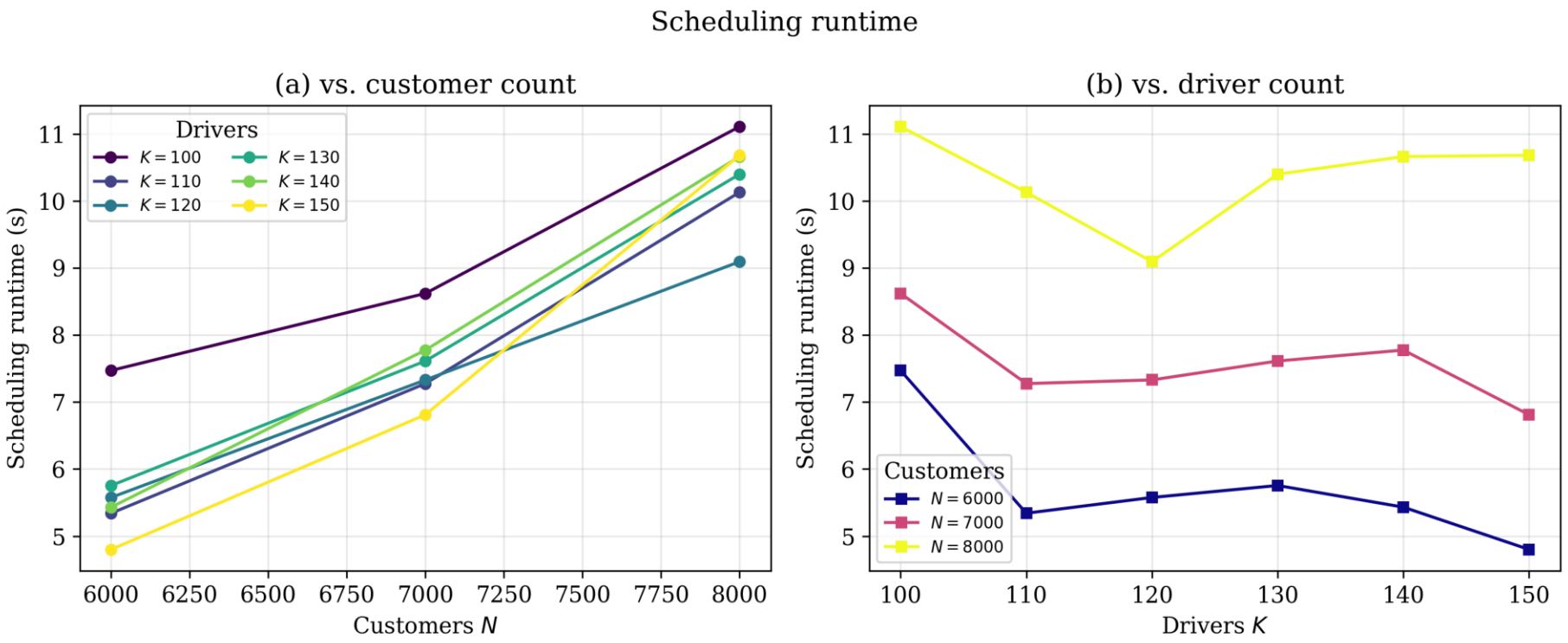
- Solving the schedule for drivers separately causes assignment conflicts
  - ILPs are solved again in a greedy resolve loop
- The resolve first looks at the pickup delays
  - Choose the best one for each customer
- After this some drivers will have free slots in their schedules
  - More customers might be possible to assign
- Drivers are looped through in five different orders
  - Lowest index, number of assignments and number of conflicts
- The ILP is solved again without the already reserved conflicted drivers
  - Unassigned customers are greedily assigned to free slots
- The loop is iterated through five times or until the amount of assigned customer doesn't increase

Experiments

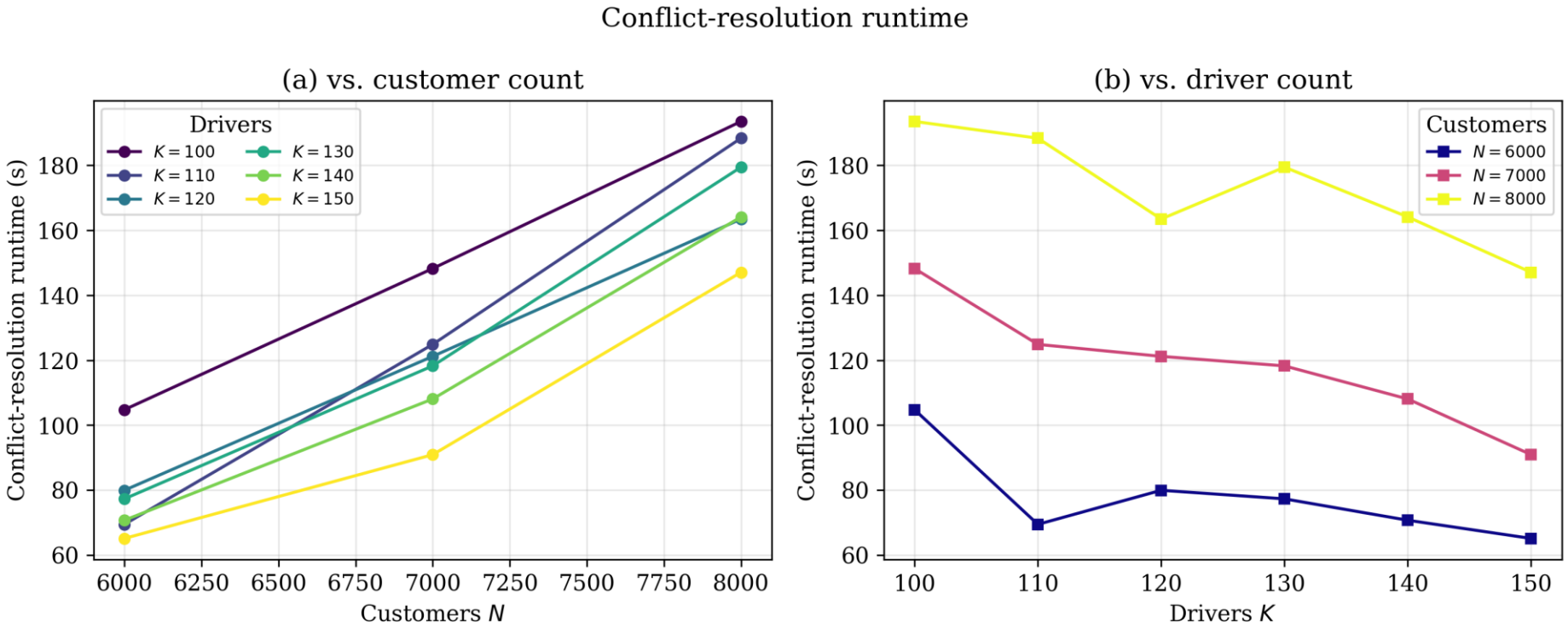
- Pipeline run with 18+1 different synthetic scenarios
  - 6000-8000 customers, 100-150 drivers
  - 1000 customers, 200 drivers in reference
- Customer
  - wanted pickup: 0–600 min
  - ride length: 5–30 min
  - 3–6 preferred drivers
- Driver
  - shift starts at 0–60 min
  - ends at 240–600 min

- Parameters
  - At most 5 drivers per customer
  - Max pickup delay: 15 min
  - Break between rides: 10 min minimum, 60 min maximum
  - Time granularity: 1 min
  - Weights: delay 1, idle 1, unserved penalty 300
  - At most 5 resolve rounds

Results: Runtime 1/2

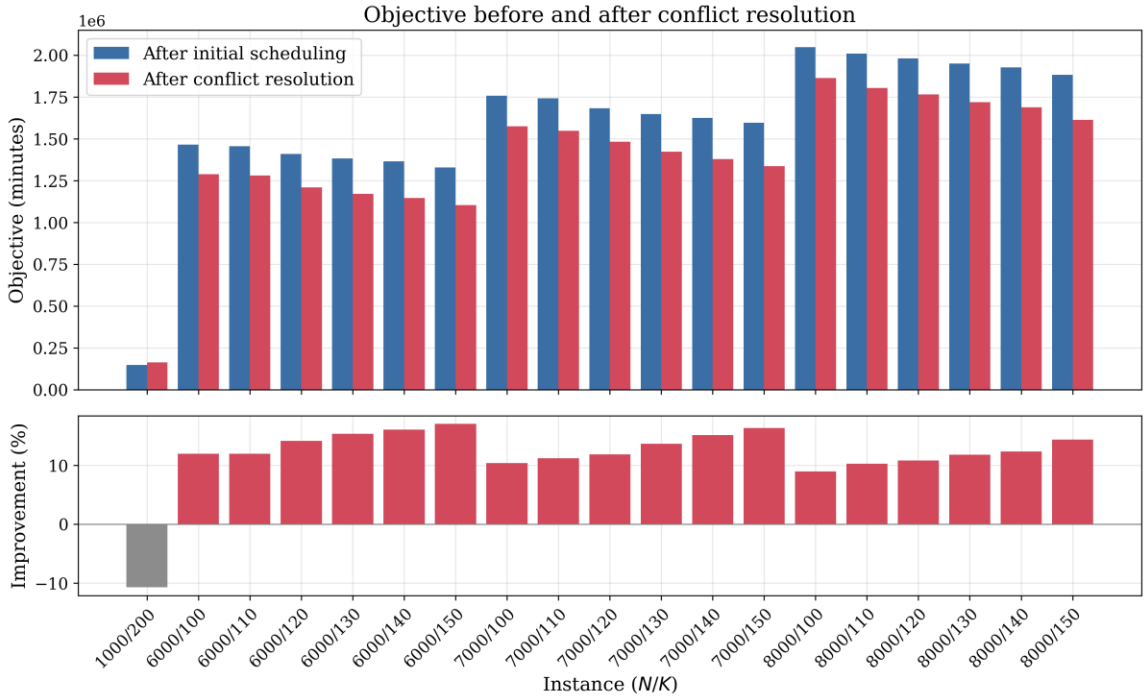


Results: Runtime 2/2



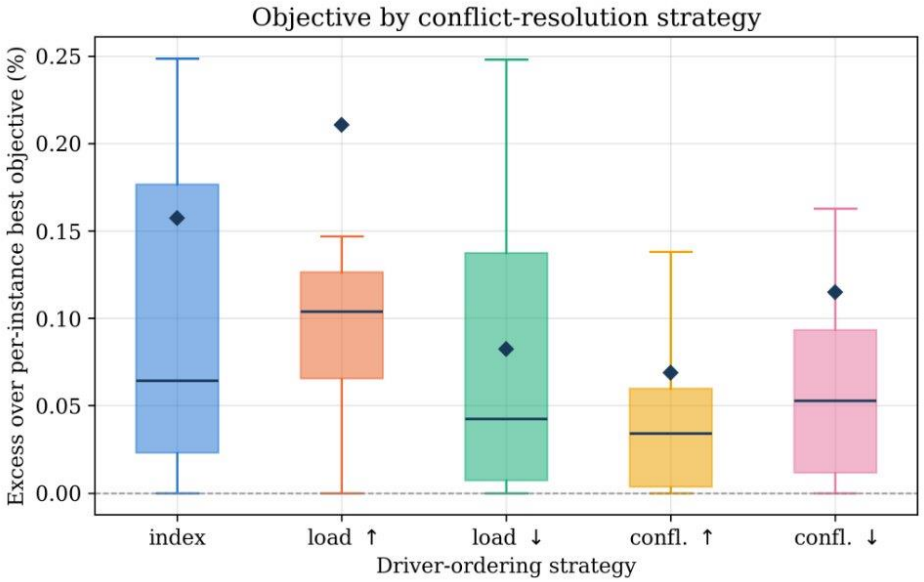
Results: Benefit of the resolving

- Resolve yields about 9-17% improvement
  - -11% worse for the reference



Results: Driver order in resolve

- No driver ordering stands out clearly
- Conflict-based orderings are slightly better
- Median excess over per-scenario best < 0.11 %



Conclusion

- Decomposing into matching, per-driver scheduling and conflict resolution works
- Largest scenario solved in a bit over five minutes
- Runtime grows superlinearly with customers, slightly decreases with drivers
- Scheduling is cheap (≤11 s), matching ~120 s, conflict resolution up to ~190 s
- Conflict resolution improved the objective by 9–17 %, but made the reference ~11 % worse
- Driver ordering hardly matters, at least with the used parameters

Limitations

- Only the time and coverage dimensions modelled
- Conflict resolution is greedy and only optimal per driver
- Resolve decides on pickup delay alone
- All tested scenarios are customer-dense

References

- Ford, L. R., Jr., & Fulkerson, D. R. (1957). A simple algorithm for finding maximal network flows and an application to the Hitchcock problem. Canadian Journal of Mathematics, 9, 210–218.
- Edmonds, J., & Karp, R. M. (1972). Theoretical improvements in algorithmic efficiency for network flow problems. Journal of the ACM, 19(2), 248–264.
- Psaraftis, H. N. (1980). A dynamic programming solution to the single vehicle many-to-many immediate request dial-a-ride problem. Transportation Science, 14(2), 130–154.
- Dailey, D. J., Loseff, D., & Meyers, D. (1999). Seattle smart traveler: Dynamic ridematching on the World Wide Web. Transportation Research Part C: Emerging Technologies, 7(1), 17–32.
- Chaube, V., Kavanaugh, A. L., & Pérez-Quiñones, M. A. (2010). Leveraging social networks to embed trust in rideshare programs. 2010 43rd Hawaii International Conference on System Sciences, 1–8. IEEE.
- Chan, N. D., & Shaheen, S. A. (2012). Ridesharing in North America: Past, present, and future. Transport Reviews, 32(1), 93–112.
- Agatz, N., Erera, A., Savelsbergh, M., & Wang, X. (2012). Optimization for dynamic ride-sharing: A review. European Journal of Operational Research, 223(2), 295–303.
- Furuhata, M., Dessouky, M., Ordóñez, F., Brunet, M.-E., Wang, X., & Koenig, S. (2013). Ridesharing: The state-of-the-art and future directions. Transportation Research Part B: Methodological, 57, 28–46.
- Mitropoulos, L., Kortsari, A., & Ayfantopoulou, G. (2021). A systematic literature review of ride-sharing platforms, user factors and barriers. European Transport Research Review, 13(1), Article 61.
- Pigalle, E., & Aguiléra, A. (2023). Ridesharing in all its forms – Comparing the characteristics of three ridesharing practices in France. Journal of Urban Mobility, 3, Article 100045.
- Gurobi Optimization, LLC. (2026). Gurobi Optimizer reference manual. https://www.gurobi.com

Thank you!

Questions?