Appointment Scheduler

- Motivation
- Modeling choices and assumptions
- Problem formulation
- Results
- Conclusion and Remarks

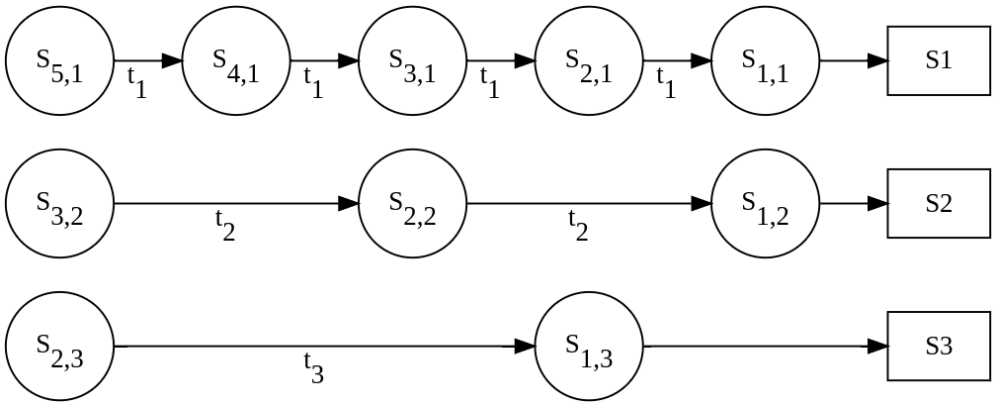
Motivation

At some area:

- Multiple **service providers**
- **User** wants to consume one or more services
- **Unnecessary** waiting times (queueing)

Modelling choices and assumptions

- Assign **user** to a **time-spot**
- Many users, few providers
- User visits **few (1-3)** providers
- Create only as many spots as needed



Notation

- K Service Providers
- N Users
- N_k users requested service k
- J_k spots for service k
- Position $j = 1, \ldots, J_k$ in service k
- $J_k = N_k$

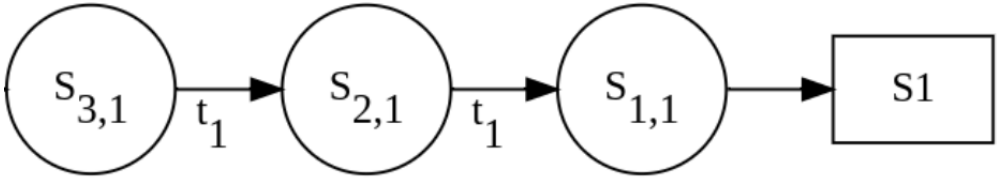
Timeslots for service

Concept: Timeslot for Service

Variable: $s_{j,k} \in \mathbb{R}$

($k = 1, \ldots, K \quad j = 1, \ldots, J_k$)

opening$_k$ and closing$_k$ from problem data



Properties

- Ordered, No Overlap
- Not before opening hour
- Not after closing hour

Constraints

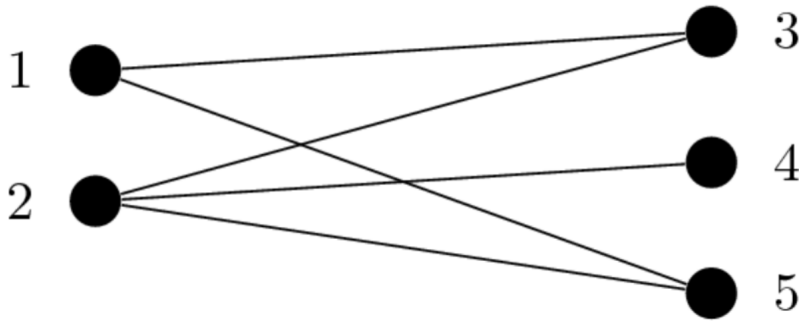
- $s_{j,k} + t_k \leq s_{j+1,k}$
- opening$_k \leq s_{1,k}$
- $s_{J_k,k} + t_k \leq$ closing$_k$

User to time-spot assignment

Concept: Assigning users to spots in service

Variable: $x_{ij,k} \in \{0, 1\}$

(for $k = 1, \ldots, K \quad j = 1, \ldots, J_k \quad i = 1, \ldots, N_k$)



Properties

- One user per spot
- One spot per user

Constraints

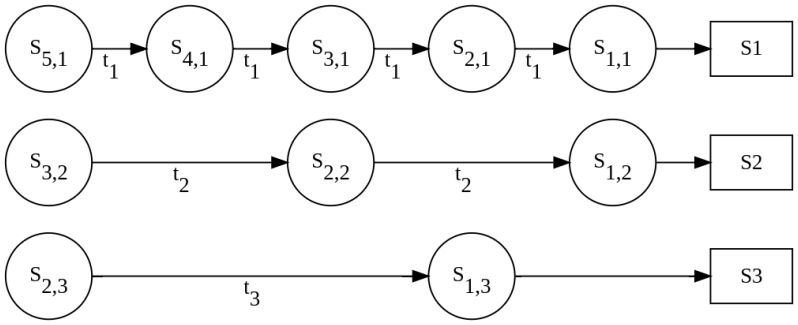
- $\sum_{i=1}^{N} x_{ij,k} = 1$
- $\sum_{j=1}^{N} x_{ij,k} = 1$

No Cross-Service overlaps

Concept: Spots between queues can not overlap if assigned to the same user

Variable: $b_{i,kk'} \in \{0,1\}$

$(k, k' = 1, \ldots, K \quad j, j' = 1, \ldots, J_k \quad i = 1, \ldots, N_k \quad k \neq k')$



Properties

- Antisymmetry
- Transitivity
- No cross-overlaps

Constraints

- $b_{i,kk'} = 1 - b_{i,k'k}$
- $b_{i,ab} + b_{i,bc} - 1 <= b_{i,ac}$
- $s_{j,k} + t_k \leq s_{j',k'} + M * (3 - x_{ij,k} - x_{ij',k'} - b_{i,kk'})$

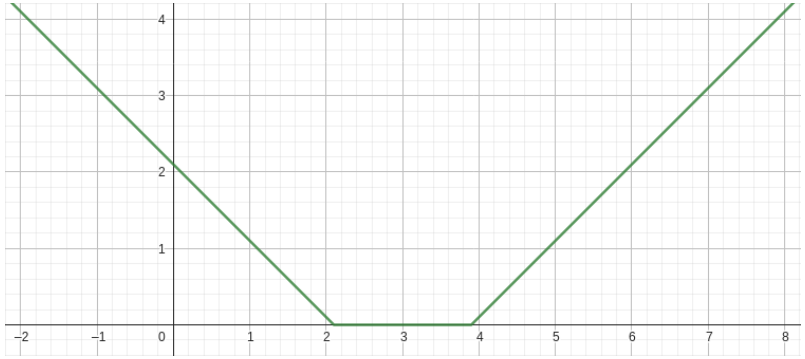
Cost

Concept: Fulfill user preferences as good as possible

Variable: $c_{ij,k} \in \mathbb{R}$

($k = 1, \ldots, K \quad j = 1, \ldots, J_k \quad i = 1, \ldots, N_k$)

$e_{i,k}$ and $l_{i,k}$ from problem data



Properties

- Positivity
- Penalty for too early
- Penalty for too late

Constraints

- $c_{ij,k} \geq 0$
- $c_{ij,k} \geq e_{i,k} - s_{j,k} - M * (1 - x_{ij,k})$
- $c_{ij,k} \geq s_{j,k} + t_k - l_{i,k} - M * (1 - x_{ij,k})$

Final Problem formulation

$\min_{x,s,c,b} \sum_{k=1}^{K} \sum_{j=1}^{J} \sum_{i=1}^{N} c_{ij,k}$

s.t.

(Timeslots for Service)

- $s_{j,k} + t_k \leq s_{j+1,k}$
- $\text{opening}_k \leq s_{1,k}$
- $s_{J_k,k} + t_k \leq \text{closing}_k$

(Assignment)

- $\sum_{i=1}^{N} x_{ij,k} = 1$
- $\sum_{j=1}^{N} x_{ij,k} = 1$

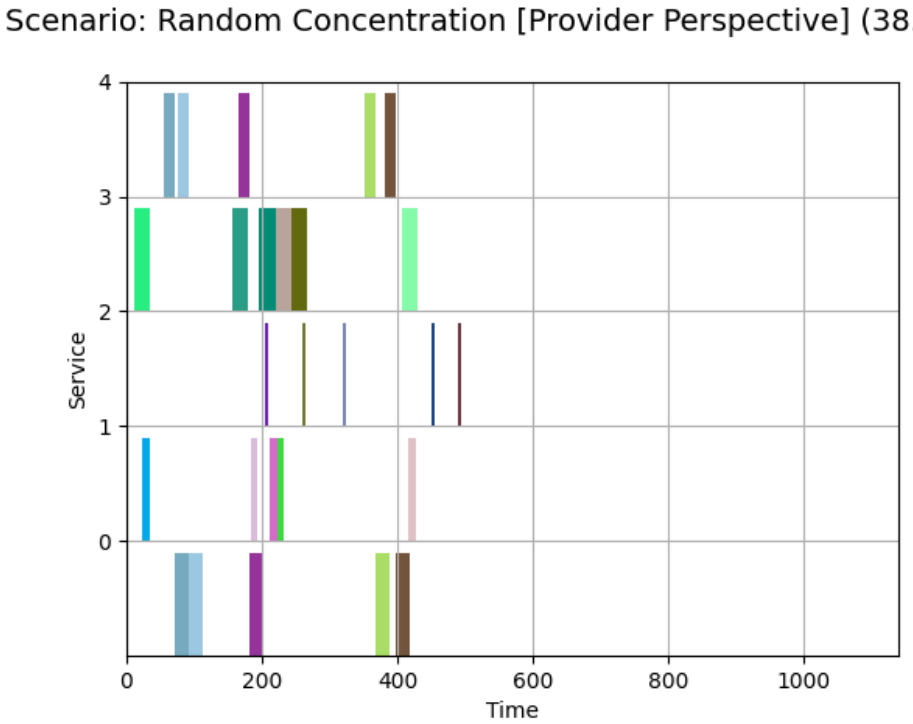
(No Cross-Service overlaps)

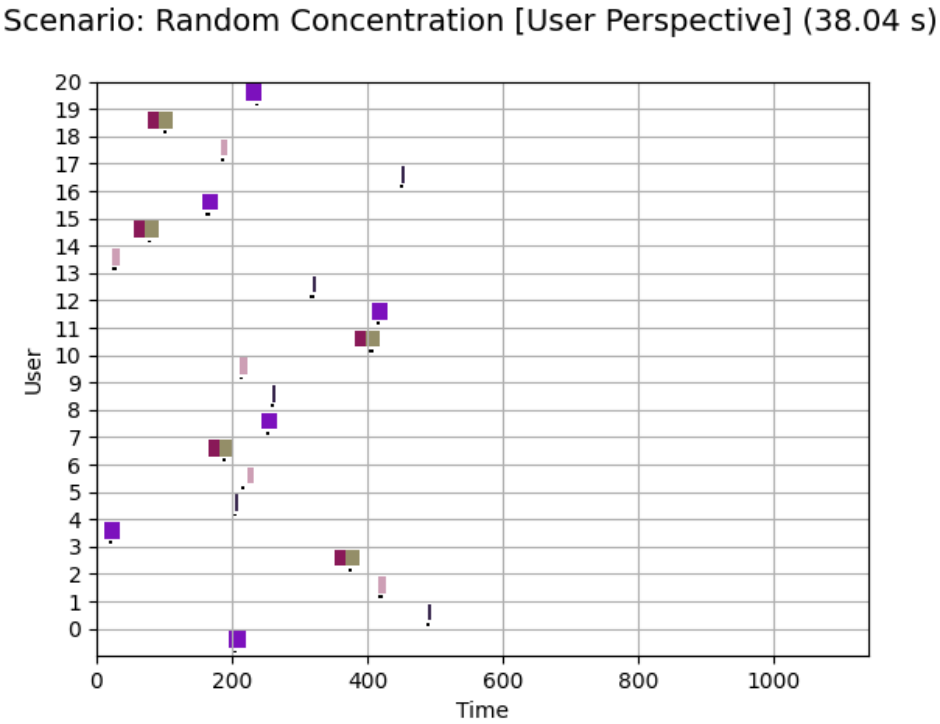
- $b_{i,kk'} = 1 - b_{i,k'k}$
- $b_{i,ab} + b_{i,bc} - 1 <= b_{i,ac}$
- $s_{j,k} + t_k \leq s_{j',k'} + M * (3 - x_{ij,k} - x_{ij',k'} - b_{i,kk'})$

(Cost)

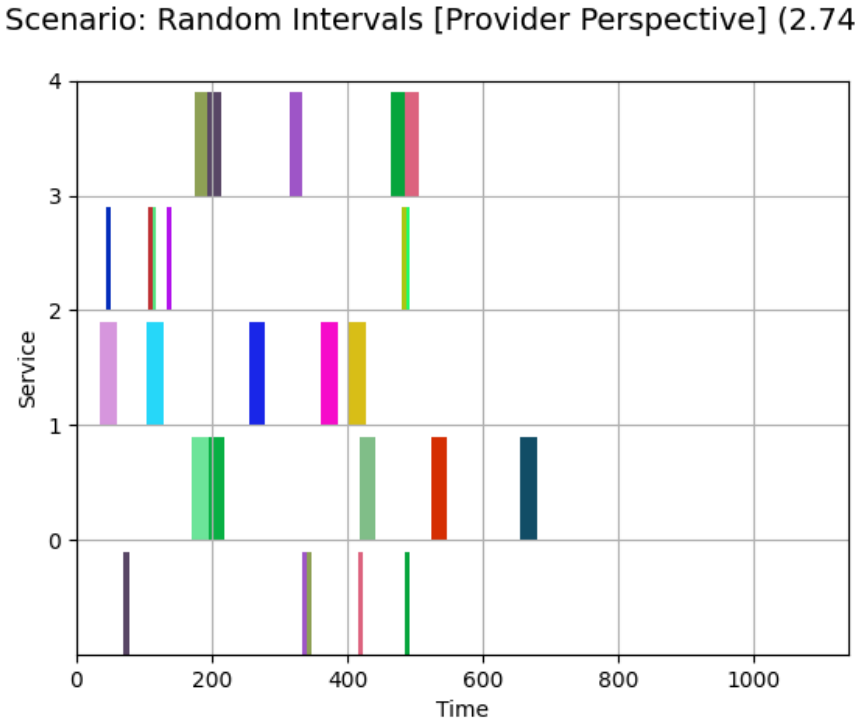
- $c_{ij,k} \geq 0$
- $c_{ij,k} \geq e_{i,k} - s_{j,k} - M * (1 - x_{ij,k})$
- $c_{ij,k} \geq s_{j,k} + t_k - l_{i,k} - M * (1 - x_{ij,k})$

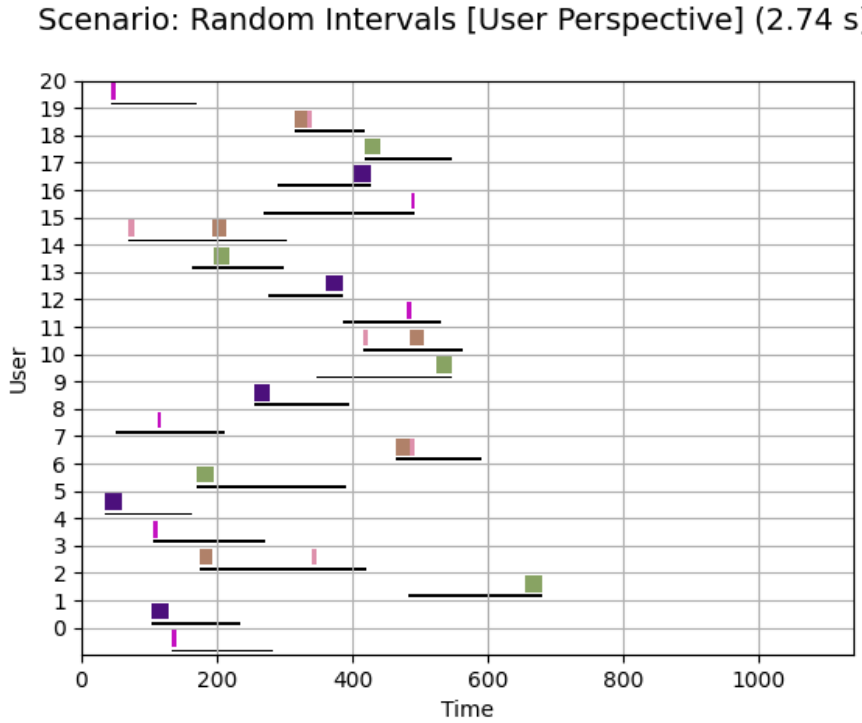
Results: Scenario “Random Concentration”



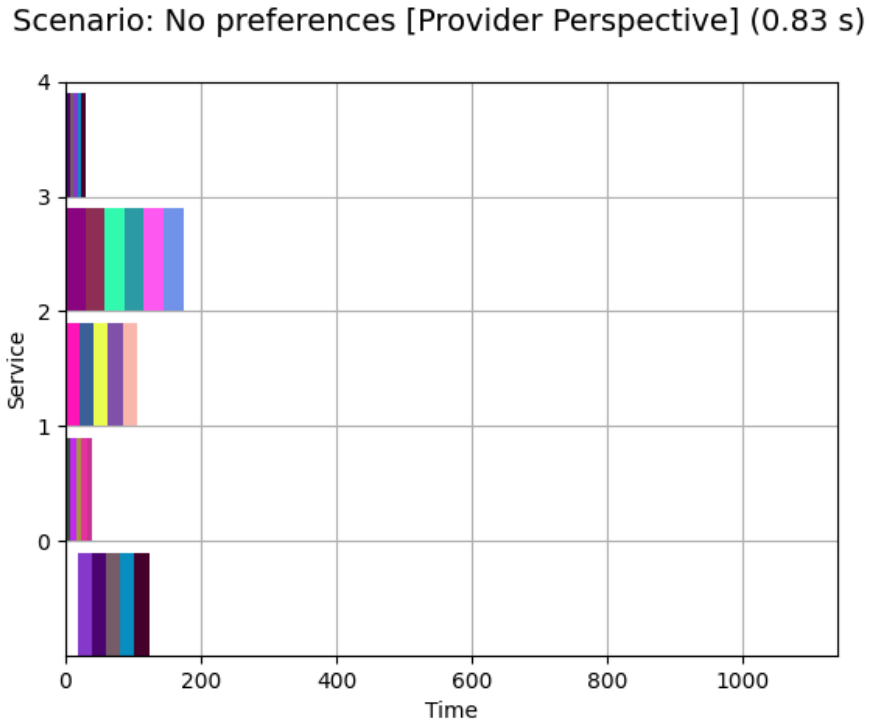


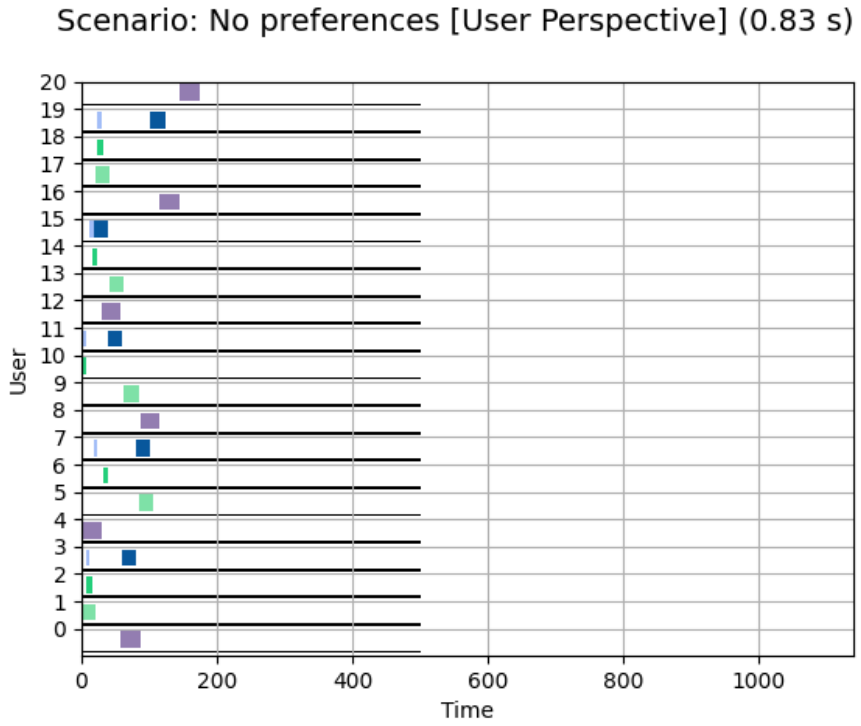
Results: Scenario “Random Intervals”



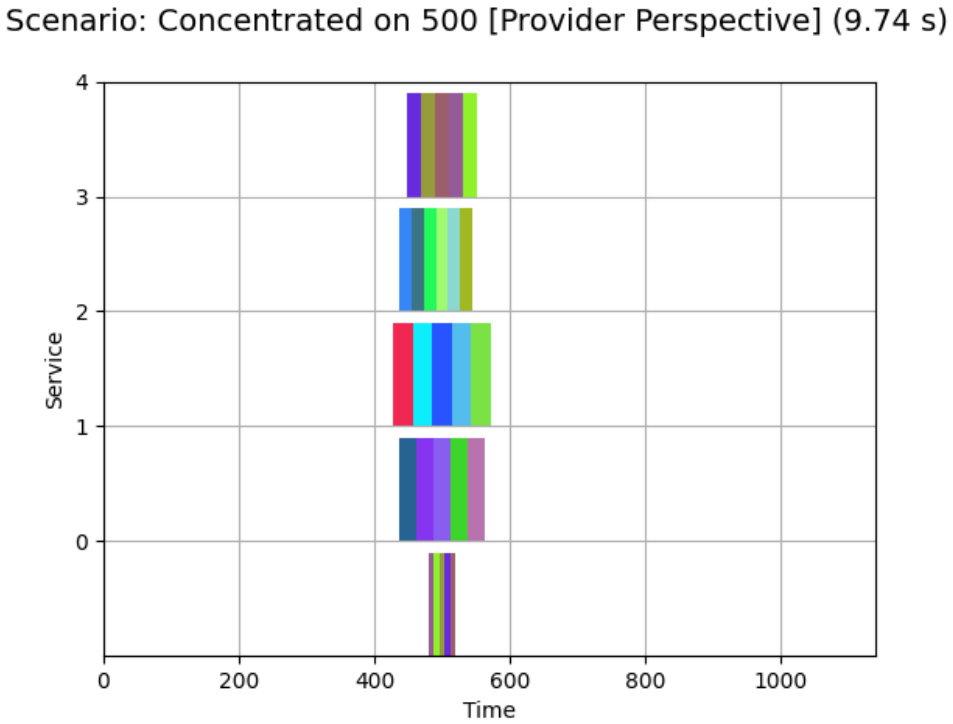


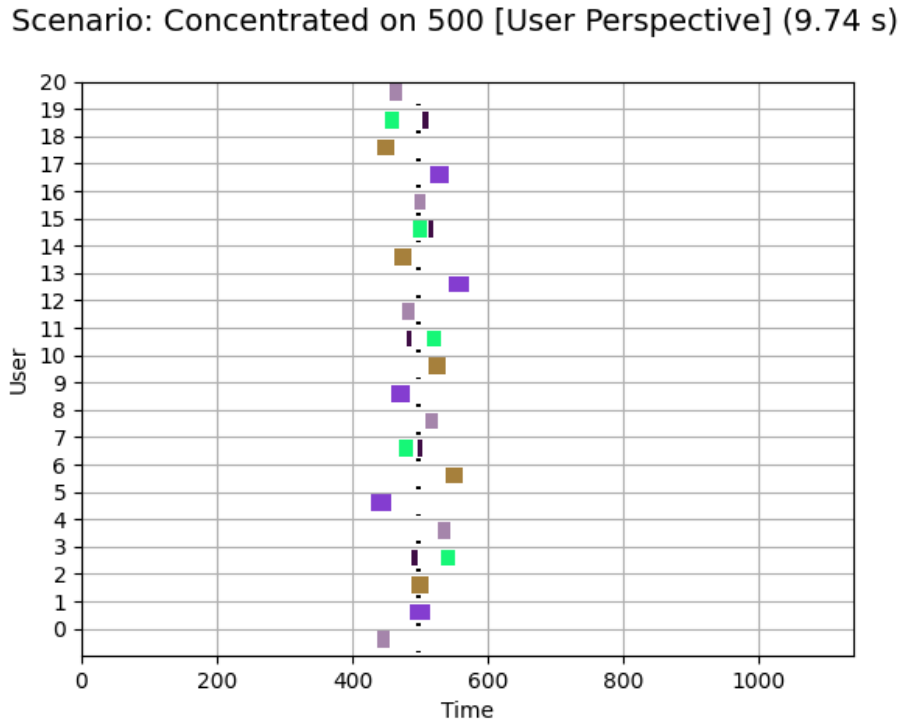
Results: Scenario “No Preferences”





Results: Scenario “Concentrated on 500”





Comparison of results

Scenario	Total Cost	Time Used
Random Concentration	308.49	38.03 s
Random Intervals	0.0	2.73 s
No Preferences	0.0	0.8 s
Concentrated on 500	921.49	9.7 s

Conclusion

Practicality

- Very flexible formulation.
- A lot of constraints, need to assume few service providers and few requests per user
- Each additional constraint helps!

Scalability

- Partitioning and Heuristics
- Improve formulation

Personal

- Intuition

Remarks

CVXPy

- **Great Library!!!**
- Cannot set parameters for all optimizers

General

- Sometimes big rounding erros (*math.isclose()* defaults not enough)
- **Gurobi is ~50 times faster**

End

Thank you for listening!