Problem decomposition and approximation for shared mobility applications with endogenous congestion: integrated vehicle assignment and routing in capacitated transportation networks:

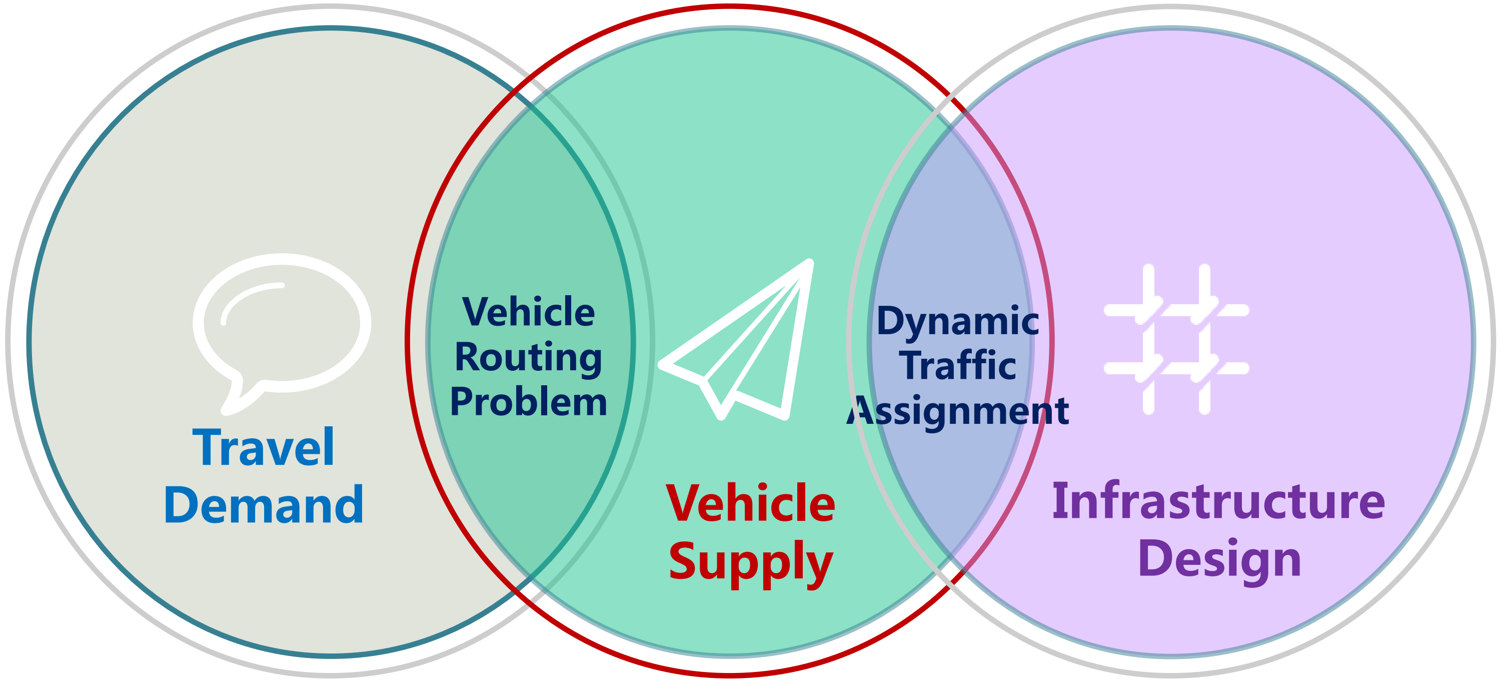
Jiangtao Liu (jliu215@asu.edu)
Xuesong Zhou (xzhou74@asu.edu)
School of Sustainable Engineering and the Built Environment

Pitu Mirchandani (pitu@asu.edu)
School of Computing, Informatics, and Decision Systems Engineering

Arizona State University

Outline

1. Introduction



How to optimize demand, **supply** and infrastructure?

1. Introduction

Vehicle Routing Problem:

Input:

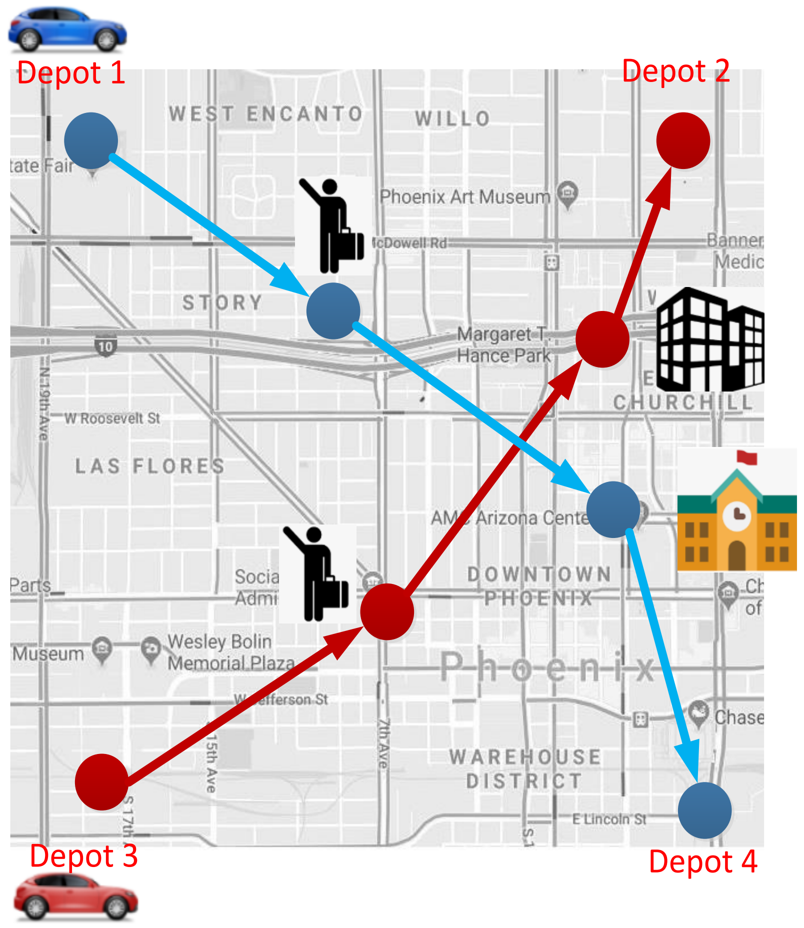
- ❑ **Network**: Virtual point-to-point network
- ❑ **Passenger trip request**: has specific pick-up and drop-off location with time windows
- ❑ **Vehicle carrying capacity:** considered

Goal:

- ❑ System Optimal

Output:

- ❑ **Vehicle-to-passenger assignment**: will be found
- ❑ **Variable**: discrete vehicle routing and scheduling



1. Introduction

Traffic Assignment Problem:

Input:

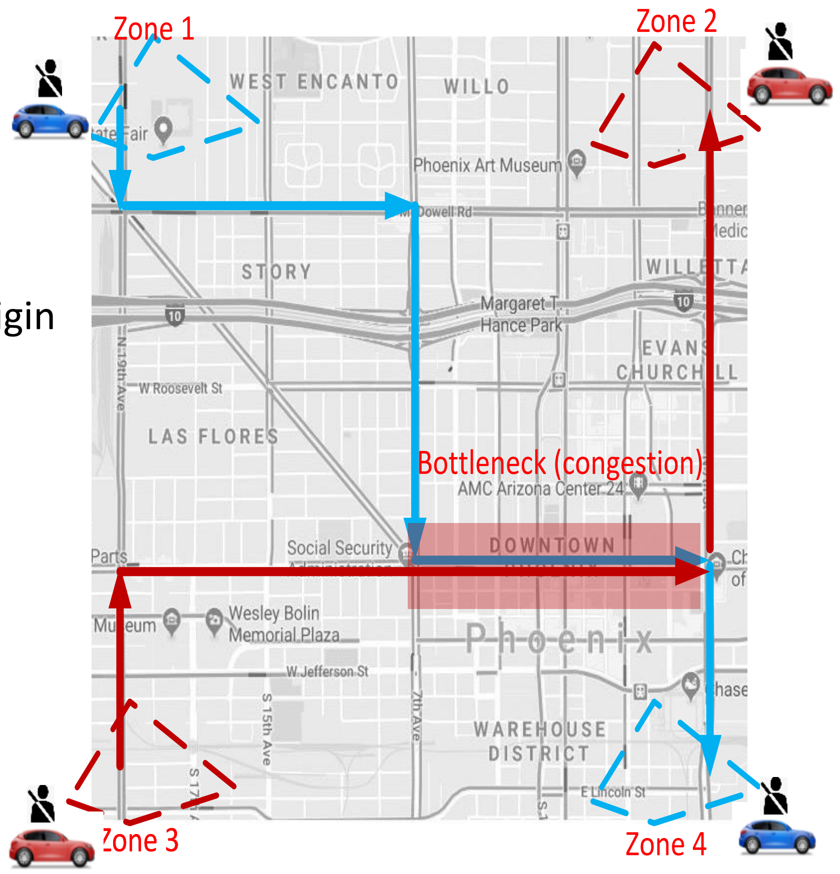
- ❑ **Network**: Physical traffic network
- ❑ **Road capacity**: capture road congestions
- ❑ **Origin and Destination**: vehicle has the same origin and destination with the assigned passengers

Goal:

- ❑ System Optimal or User Equilibrium

Output:

- ❑ continuous vehicle flow on links/paths



1. Introduction

Uncertain elements from the current to the future

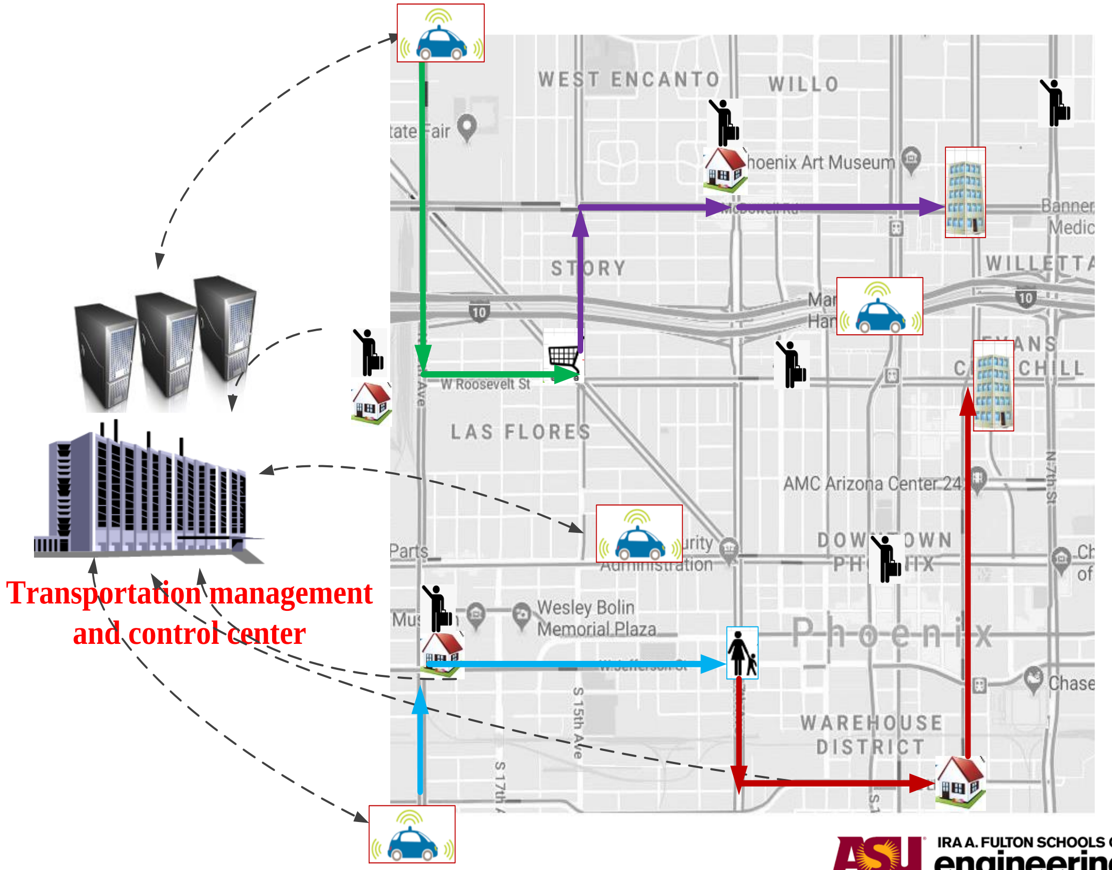
Travel Demand	Vehicle Supply	Infrastructure
• **Trip request**: origin and destination with self-owned vehicles or submit pickup and drop-off locations with time windows for mobility providers • **Trip privacy**: alone or rideshare • **Trip mode**: single mode or multiple modes	• **Driving mode**: self-driving or human-driven • **Routing behavior**: selfish or coordinated • **Ownership**: household or mobility providers	• **Link/Lane capacity** change • **Sensor and communication** (V2V, V2I) • **Smart Transportation network** (road, parking, depots, bus line, rail transit line, etc.)

1. Introduction

Special scenario: integration of travel demand, vehicle supply and infrastructure capacity

Keywords:

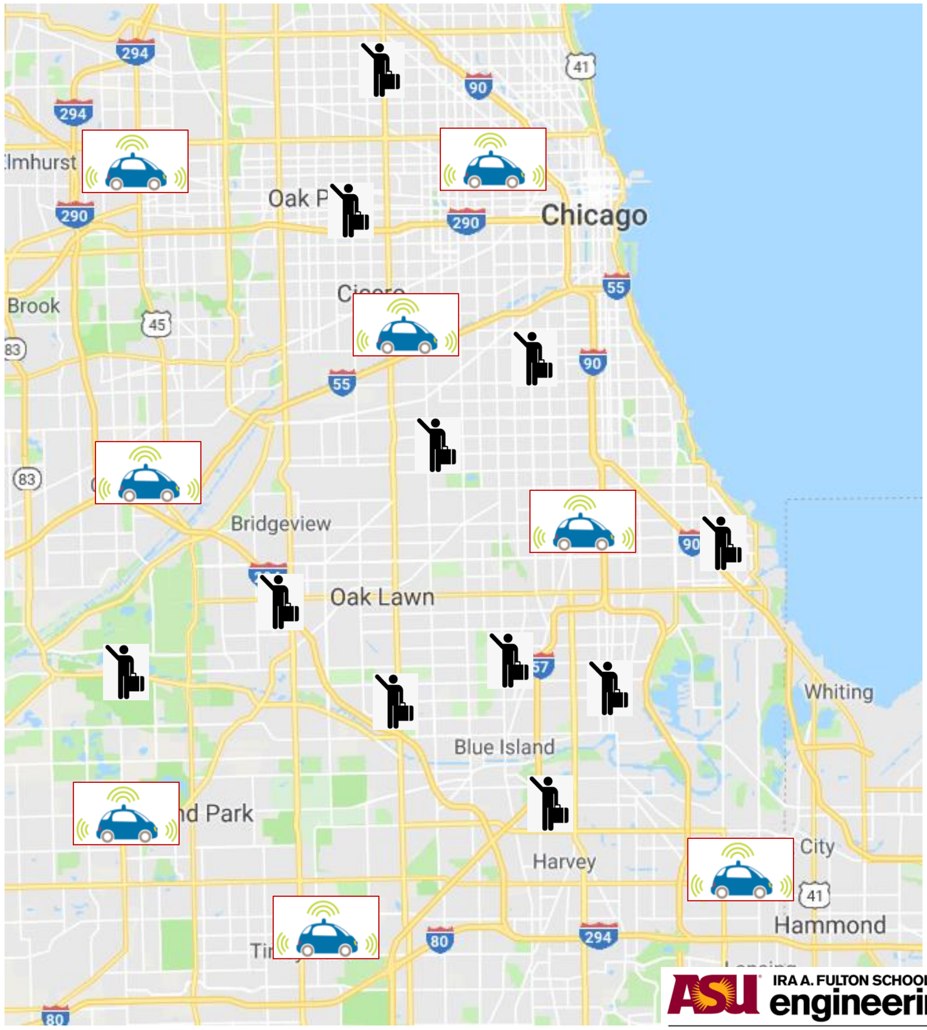
- **Physical traffic network** to consider **traffic congestion**
- **Trip requests** with Pickup and delivery with time windows
- **Autonomous vehicles with carrying capacity** for ride sharing
- **Central control (System optimal)**



1. Introduction

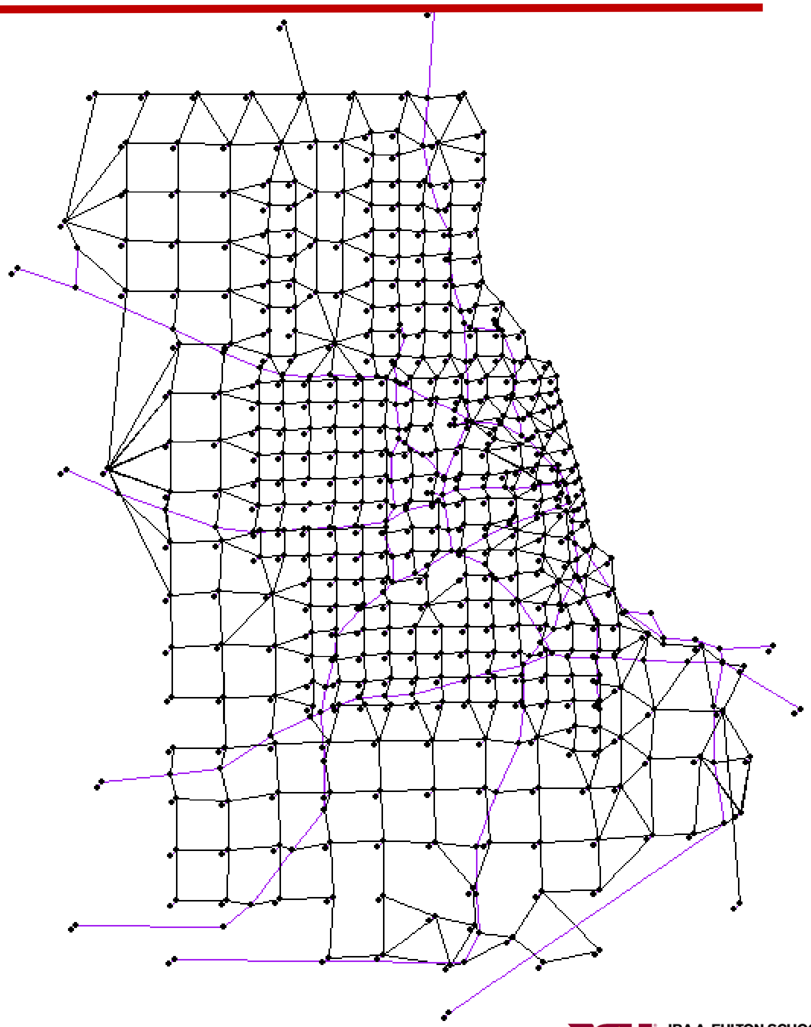
Key questions:

- How many autonomous vehicles do we need?
- How many passengers can we serve?
- How to capture the new traffic congestion?
- What is the best vehicle routing and vehicle-to-passenger assignment solution?



1. Introduction

# of nodes	1320
# of links	5431
optimization time horizon (min)	60
arc capacity each min (vehicle)	35
# of passenger groups with same OD and departure time	2226
Total trip requests	4452
# of vehicle depots	243

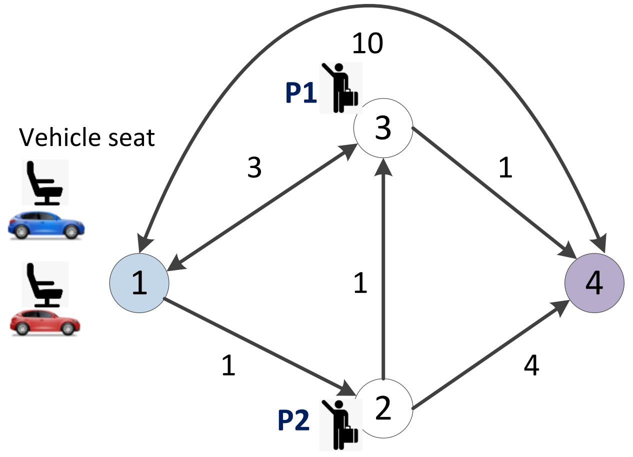

Chicago Sketch Network

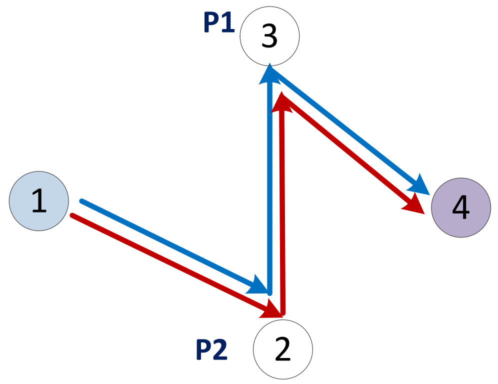
IRA A. FULTON SCHOOLS OF engineering ARIZONA STATE UNIVERSITY

2. Key Elements

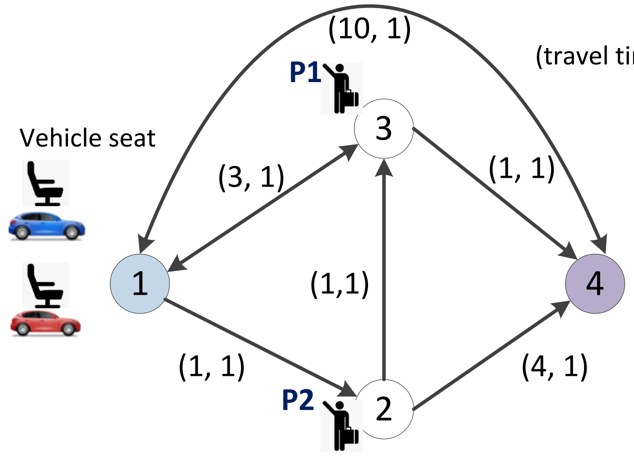
Link Capacity

Without link capacity:
Total cost is 6

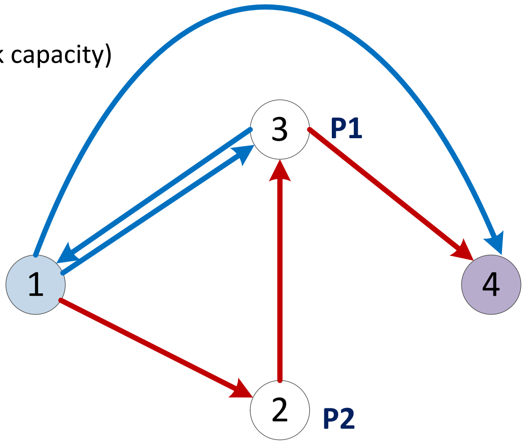




With link capacity:
Total cost is 15

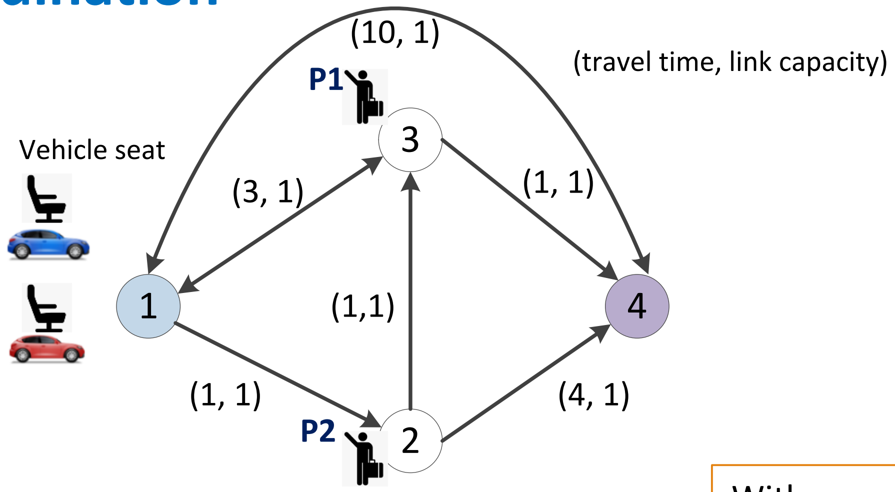


(travel time, link capacity)

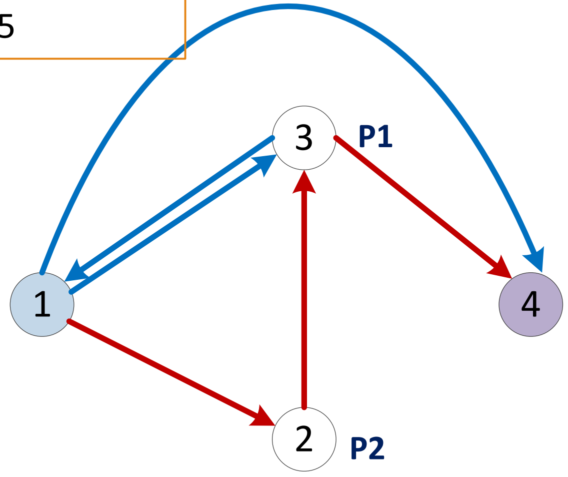


2. Key Elements

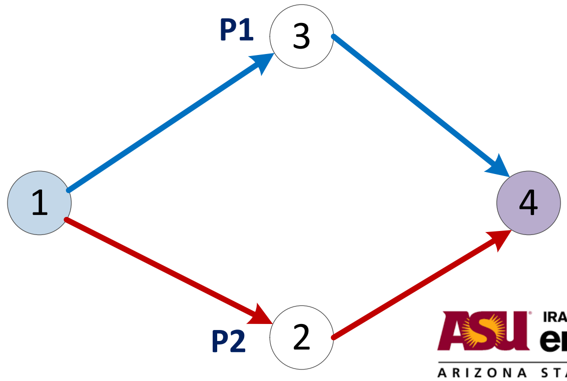
System Optimal Coordination



Without coordination (selfish routing):
Total cost is 15

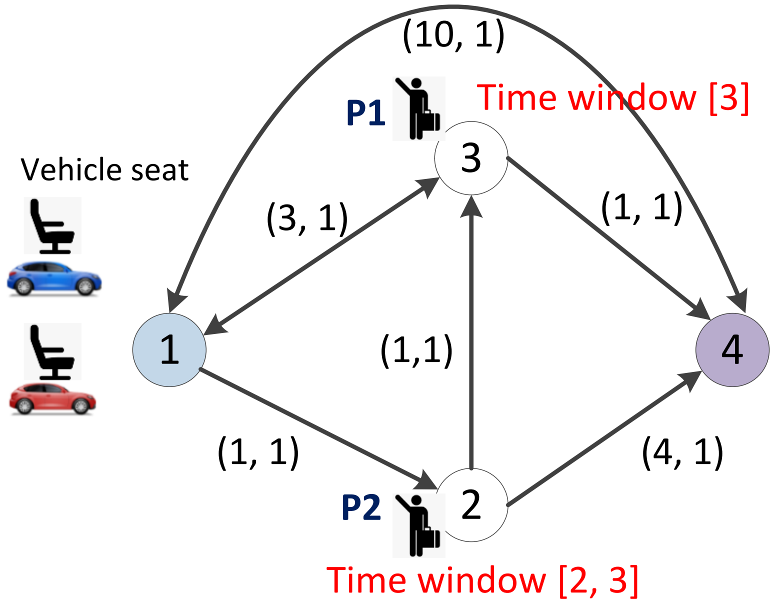


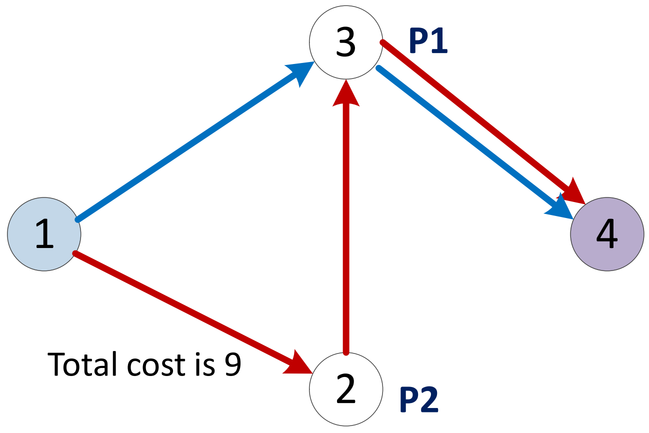
With coordination:
Total cost is 9



2. Key Elements

Time windows

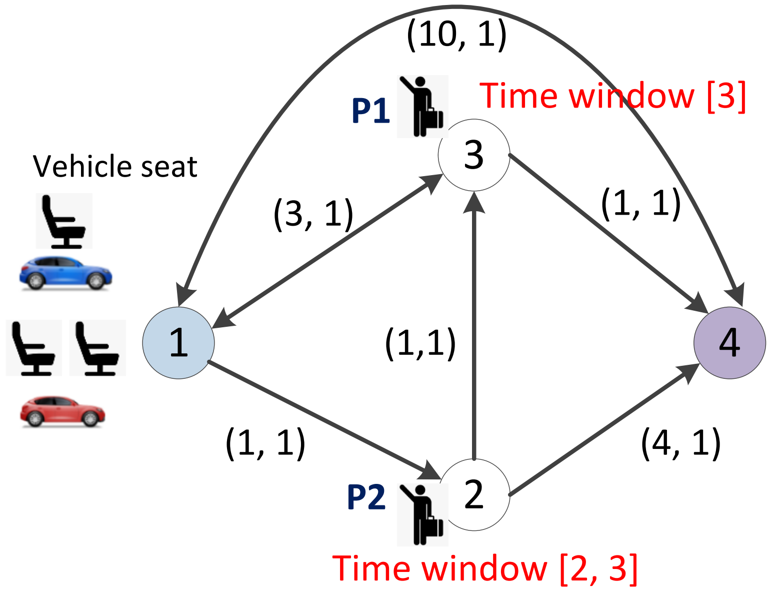


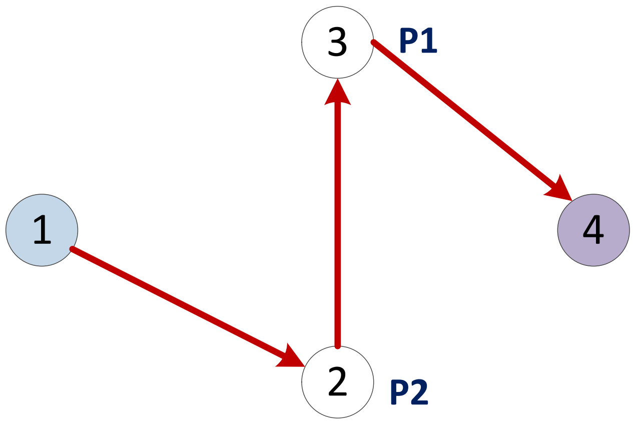


- ❑ The red vehicle can wait until time 3 to pick up passenger 2, so the blue vehicle can pick up passenger 1 at exact time 3.
- ❑ The optimal result doesn’t only optimize the vehicle routing, but also the departure time of picked up passengers.

2. Key Elements

Ridesharing (vehicle carrying capacity)



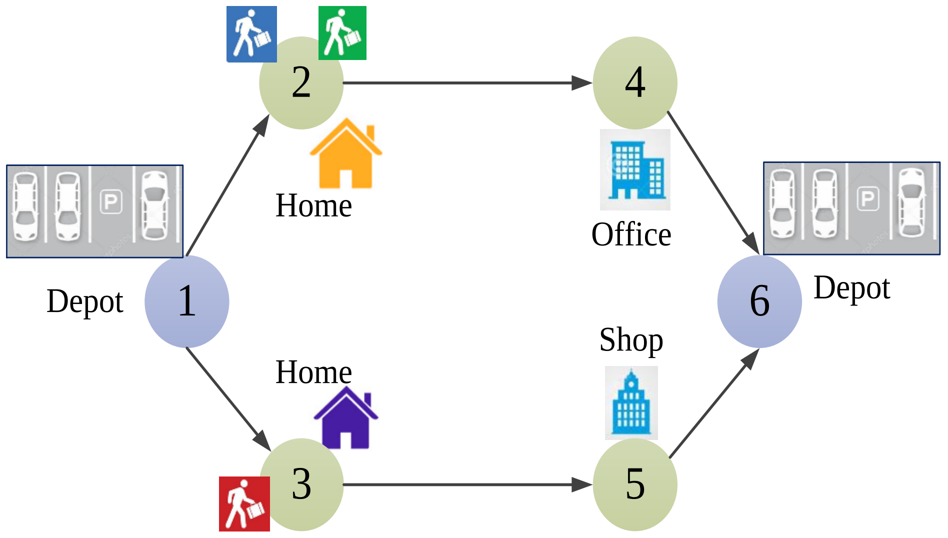


Total cost is 4

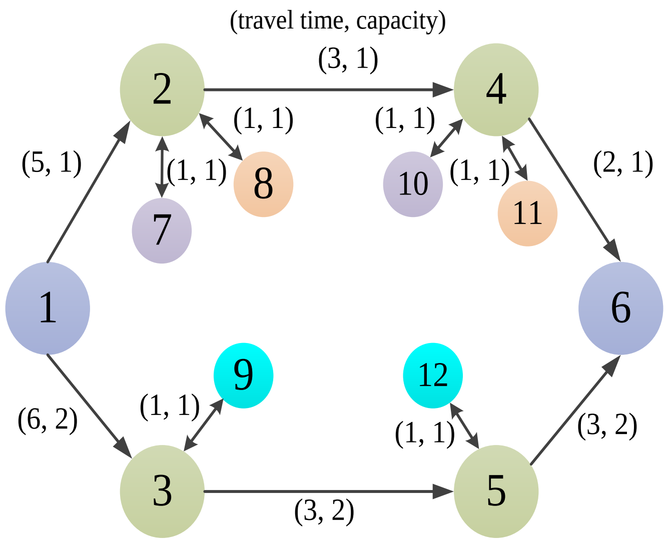
- When the red vehicle's carrying capacity is increased to 2, the total cost is reduced to 4 from 9;
- Only the red vehicle is required to serve the trip requests.

3. Space-time-state network and model formulation

Passenger trip requests: pickup and drop-off locations and time windows



(a) Physical transportation network with vehicles and trip requests

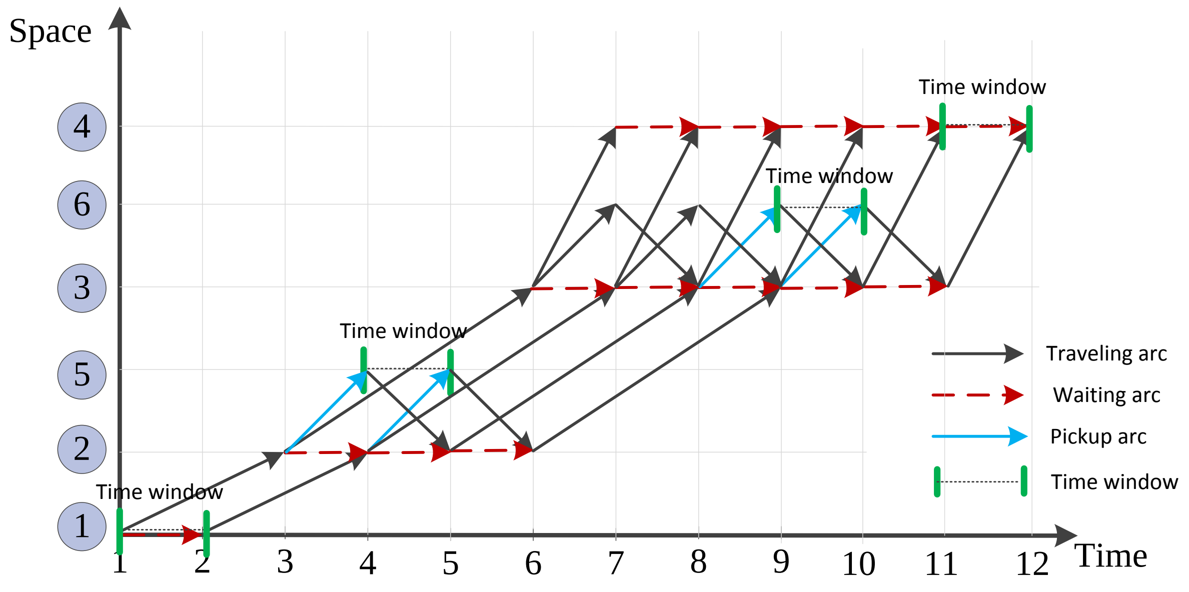


(b) Modified transportation network with virtual pickup and delivery nodes and links

Add virtual pick-up and drop-off nodes and links for each passenger

3. Space-time-state network and model formulation

Time-extended Space-time network construction for physical path $1 \to 2 \to 4 \to 6$



Space-time network

Arc (i,j,t,s) with capacity

Vertex (i,t), (j,s)

Passenger pickup time windows and locations are embedded in this network

3. Space-time-state network and model formulation

Vehicle Carrying States for passenger pickup and drop-off:

which passengers are being carried by this vehicle:

To record the **passenger service status**:

0: the passenger is not served;

1: under served (picked up but not delivered);

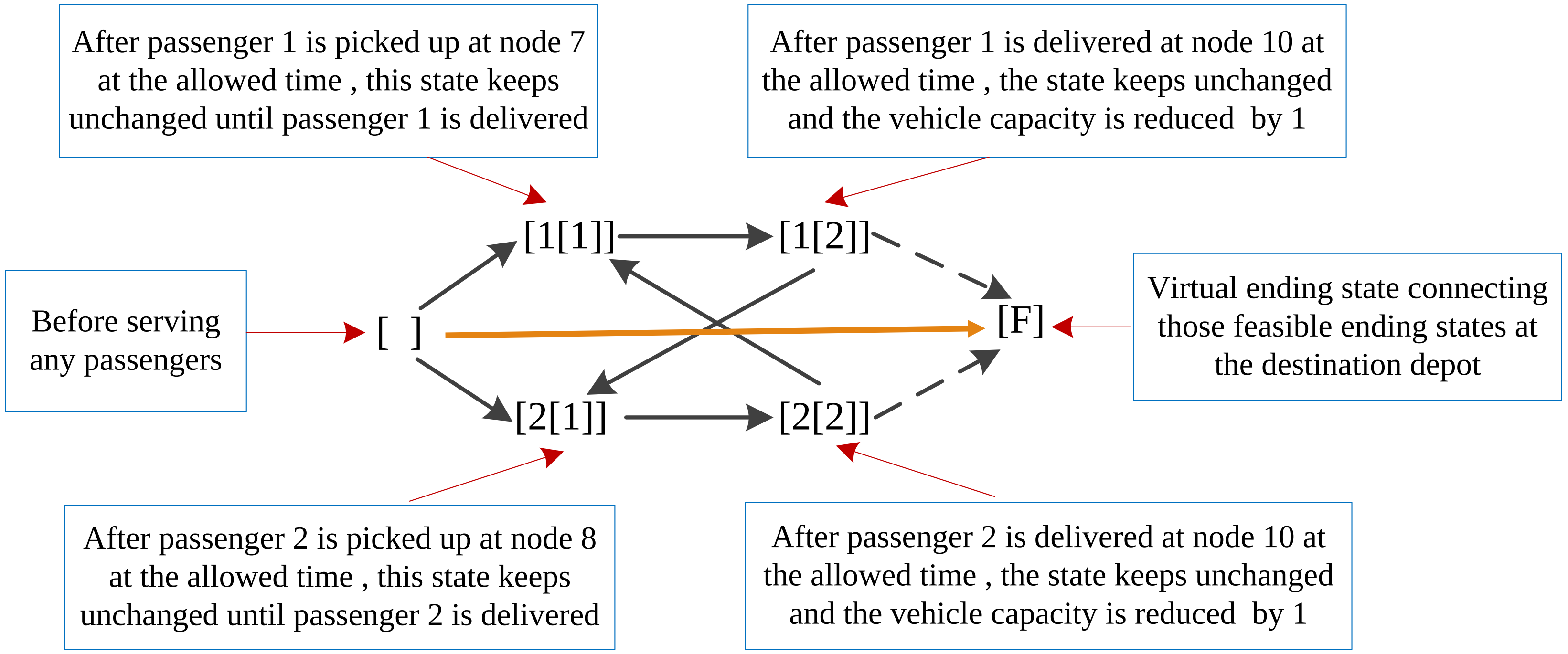
2: finished (delivered)

Example: In the case: if vehicle capacity is 1 and 2 passengers trip requests,

All possible states: [ ], [1[1]], [1[2]], [2[1]] or [2[2]]

3. Space-time-state network and model formulation

Vehicle State Transition with specific pickup and drop-off locations within time windows


After passenger 1 is picked up at node 7 at the allowed time , this state keeps unchanged until passenger 1 is delivered
After passenger 1 is delivered at node 10 at the allowed time , the state keeps unchanged and the vehicle capacity is reduced by 1
[1[1]]
[1[2]]
Before serving any passengers
[ ]
[F]
Virtual ending state connecting those feasible ending states at the destination depot
[2[1]]
[2[2]]
After passenger 2 is picked up at node 8 at the allowed time , this state keeps unchanged until passenger 2 is delivered
After passenger 2 is delivered at node 10 at the allowed time , the state keeps unchanged and the vehicle capacity is reduced by 1

Vehicle carrying state transition graph

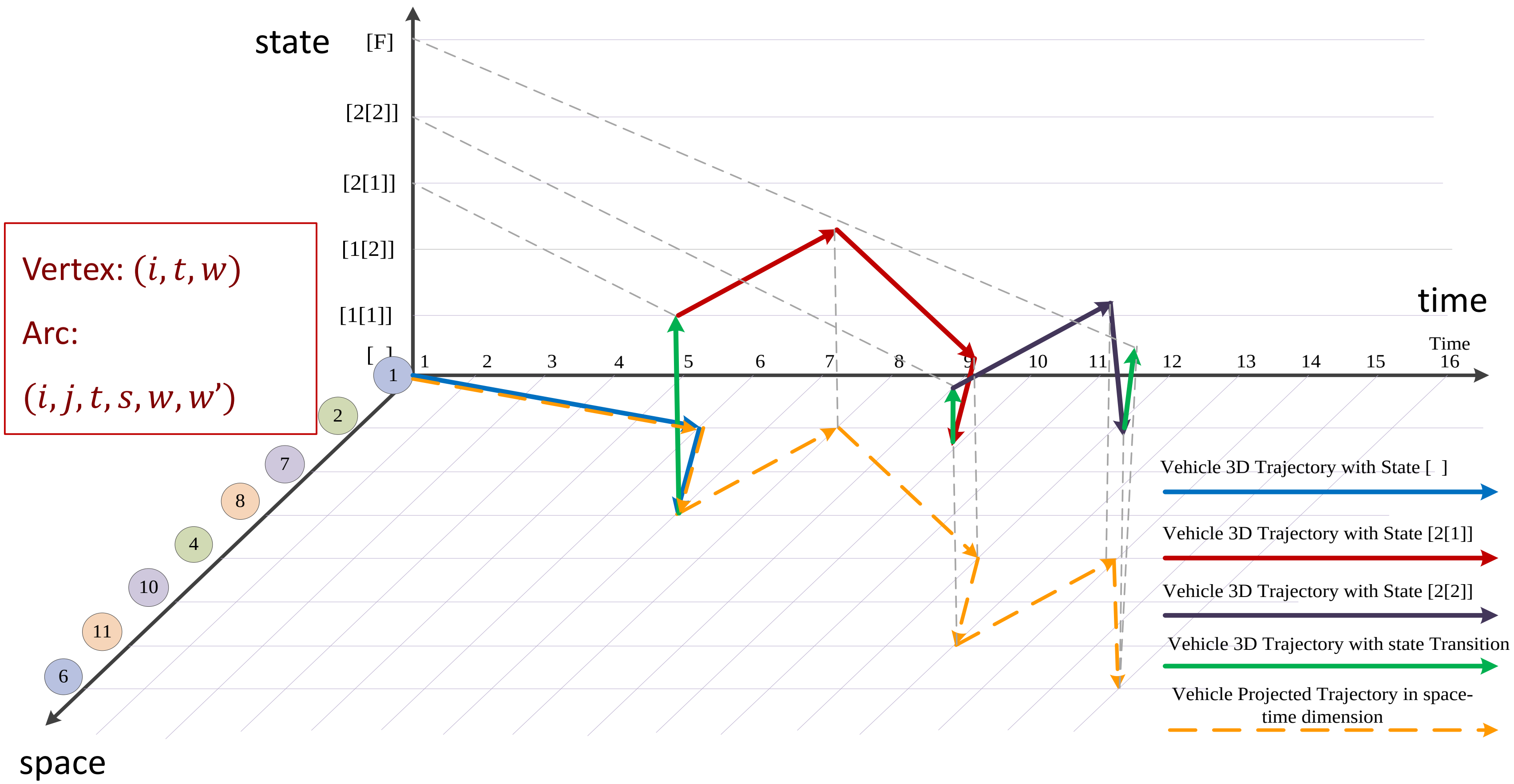
3. Space-time-state network and model formulation

One possible vehicle trajectory in a space-time-state network

Vertex: (i, t, w)

Arc:

(i, j, t, s, w, w')


state
[F]
[2[2]]
[2[1]]
[1[2]]
[1[1]]
[ ]
time
Time
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
1
2
7
8
4
10
11
6
space
Vehicle 3D Trajectory with State [ ]
Vehicle 3D Trajectory with State [2[1]]
Vehicle 3D Trajectory with State [2[2]]
Vehicle 3D Trajectory with state Transition
Vehicle Projected Trajectory in space-time dimension

3. Space-time-state network and model formulation

Arc-based agent-based formulation:

Objective function:

$$Min\ Z = \sum_a \sum_{(i,j,t,s,w,w')} (c^a_{i,j,t,s,w,w'} \times x^a_{i,j,t,s,w,w'}) \qquad (1)$$

Subject to,

(i) **Vehicle supply**: Arc-based flow balance constraint for each vehicle

$$\sum_{i,t,w:(i,j,t,s,w,w')} x^a_{i,j,t,s,w,w'} - \sum_{i,t,w:(j,i,s,t,w',w)} x^a_{j,i,s,t,w',w} = \begin{cases} -1 & j = O(a), s = DT(a), w = [0,0,\dots,0] \\ 1 & j = D(a), s = T, w = [0,0,\dots,0] \\ 0 & otherwise \end{cases} ,\forall a \qquad (2)$$

(ii) **Travel demand**: Passenger p's pick-up request constraint

$$\sum_a \sum_{i,t,s:(i,j,t,s,w,w') \in A(p)} x^a_{i,j,t,s,w,w'} = 1, \forall p \qquad (3)$$

(iii) **Infrastructure supply**: Tight road capacity constraint (endogenous congestion)

$$\sum_a \sum_w x^a_{i,j,t,s,w,w'} \le cap_{i,j,t,s}, \forall\ (i,j,t,s) \qquad (4)$$

(iv) Binary definitional constraint

$$x^a_{i,j,t,s,w,w'} \in \{0,1\} \qquad (5)$$

Remark: state definition *w* differs for passenger pickup only and passenger pickup and drop-off

3. Space-time-state network and model formulation

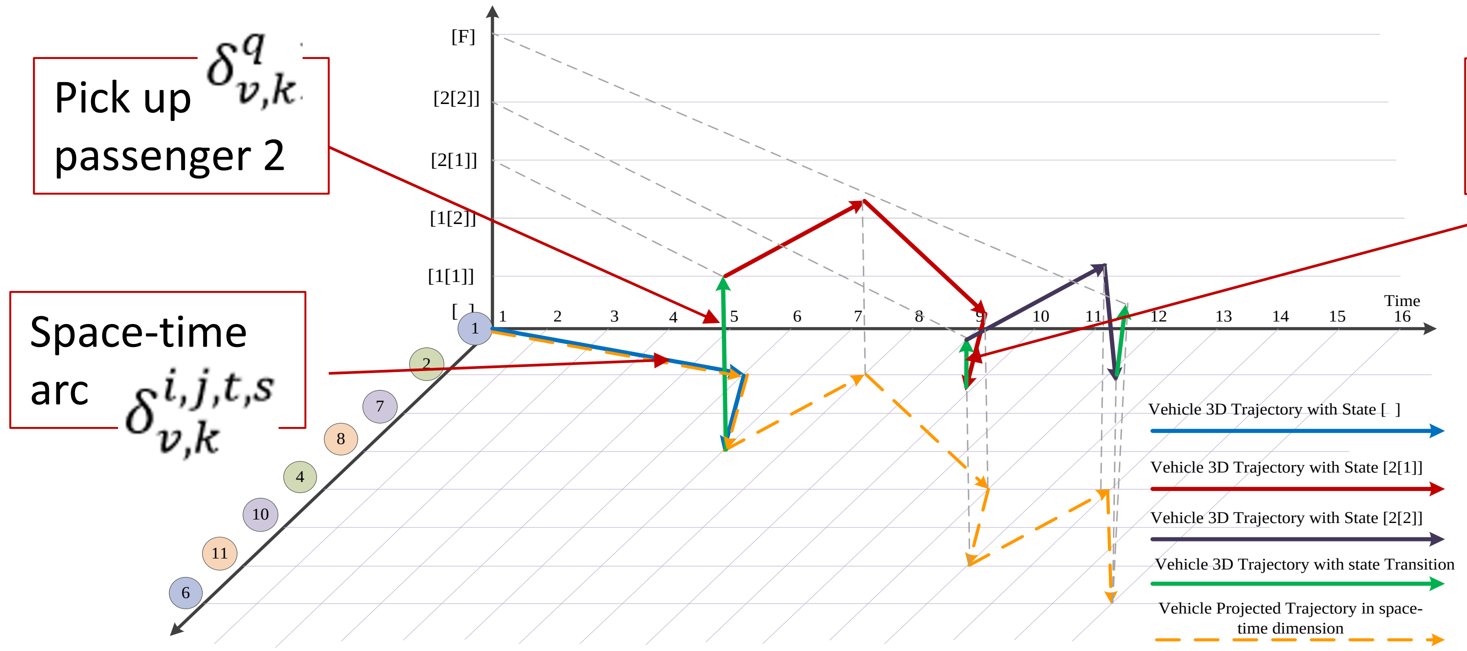
Path-based flow-based formulation:

Assumptions:

(1) **Vehicles** and **Passengers** can be grouped by its origin, destination and required service time period

(2) **All possible paths of vehicle groups** can be **enumerated** in advance.

(3) The **space-time-state** path of each vehicle group has the **mapping** information about **vehicle-to-passenger** and **vehicle-to-arc** relation.



Pick up passenger 2 $\delta^{q}_{v,k}$
Space-time arc $\delta^{i,j,t,s}_{v,k}$
Drop off passenger 2
[F]
[2[2]]
[2[1]]
[1[2]]
[1[1]]
[ ]
Time
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
1
2
7
8
4
10
11
6
Vehicle 3D Trajectory with State [ ]
Vehicle 3D Trajectory with State [2[1]]
Vehicle 3D Trajectory with State [2[2]]
Vehicle 3D Trajectory with state Transition
Vehicle Projected Trajectory in space-time dimension

3. Space-time-state network and model formulation

Path-based flow-based formulation:

$$\min \sum_{(v,k)} (c_v^k \times y_v^k) \tag{6}$$

Subject to

(i) **Vehicle supply:** Path-based vehicle group flow balance constraint:

$$\sum_k y_v^k = d(v), \forall v \tag{7}$$

(ii) **Travel demand**: Pickup requests on passenger group q:

$$\sum_{(v,k)} (y_v^k \times \delta_{v,k}^q) = g(q), \forall q \tag{8}$$

(iii) **Infrastructure supply**: Road capacity constraints (endogenous congestion):

$$\sum_{(v,k)} (y_v^k \times \delta_{v,k}^{i,j,t,s}) \leq cap_{i,j,t,s}, \forall (i,j,t,s) \tag{9}$$

(iv) Positive continuous variable:

$$y_v^k \geq 0 \tag{10}$$

3. Space-time-state network and model formulation

Simplified format of our models

1. Primal problem

$$\min cx$$
$$Ax = b$$
$$Bx \leq d$$

Flow-balance constraint

Coupling constraint (trip request, link capacity)

2. Lagrangian Relaxation

$$\min cx + \mu(d - Bx)$$
$$Ax = b$$

3. Cutting plane method

$$max_{\mu} \min_{x}[cx + \mu(d - Bx)]$$
$$Ax = b$$

4. Column generation/DW

Restricted master problem

$$\min c(\sum_i \lambda_i \times x_i)$$
$$B\sum_i \lambda_i \times x_i \leq d$$
$$\sum_i \lambda_i = 1$$
$$\lambda_i \geq 0$$

5. Augmented Lagrangian

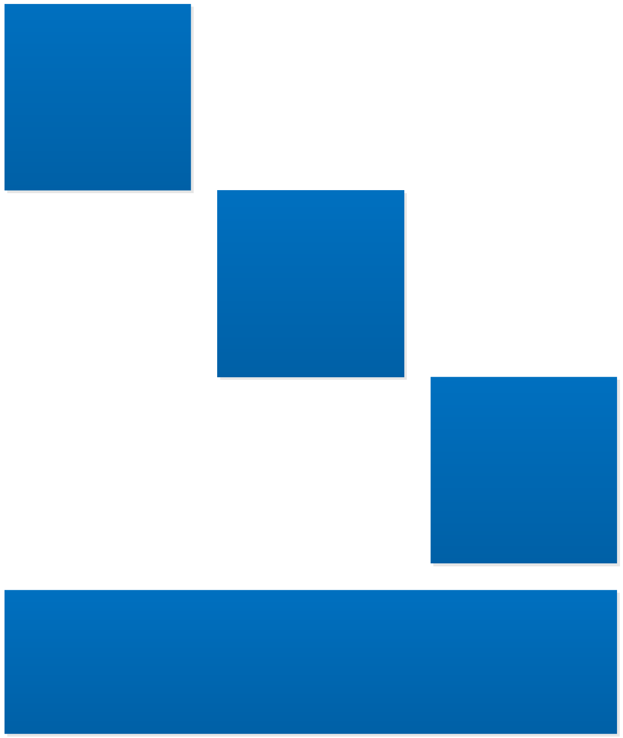
$$\min cx - \mu Bx - \pi$$
$$Ax = b$$

Subproblems

4. Dantzig-Wolfe Decomposition

$\min c_1 x_1 + c_2 x_2 + c_3 x_3$

Objective function



$a_1 x_1 = b_1$

$a_2 x_2 = b_2$

Flow-balance constraint for each vehicle

$a_3 x_3 = b_3$

$a_4 x_1 + a_5 x_2 + a_6 x_6 \leq b_4$

Coupling constraints (passenger pickup, road capacity)

4. Dantzig-Wolfe Decomposition

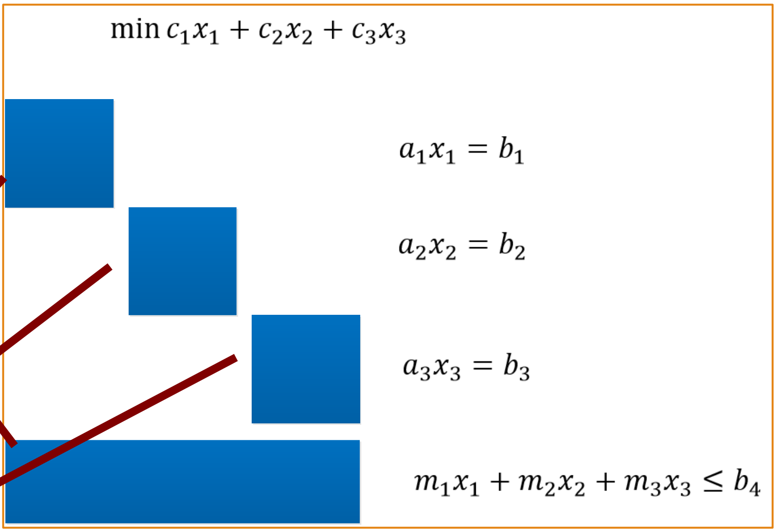
Restricted Master Problem

$$\min c_1 \sum_k (\lambda_1^k \times x_1^k) + c_2 \sum_k (\lambda_2^k \times x_2^k) + c_3 \sum_k (\lambda_3^k \times x_3^k)$$

$$m_1 \sum_k (\lambda_1^k \times x_1^k) + m_2 \sum_k (\lambda_2^k \times x_2^k) + m_3 \sum_k (\lambda_3^k \times x_3^k) \le b_4$$

$$\sum_k \lambda_i^k = 1, \forall i$$

$$\lambda_i^k \ge 0$$



$$\min c_1 x_1 + c_2 x_2 + c_3 x_3$$

$$a_1 x_1 = b_1$$

$$a_2 x_2 = b_2$$

$$a_3 x_3 = b_3$$

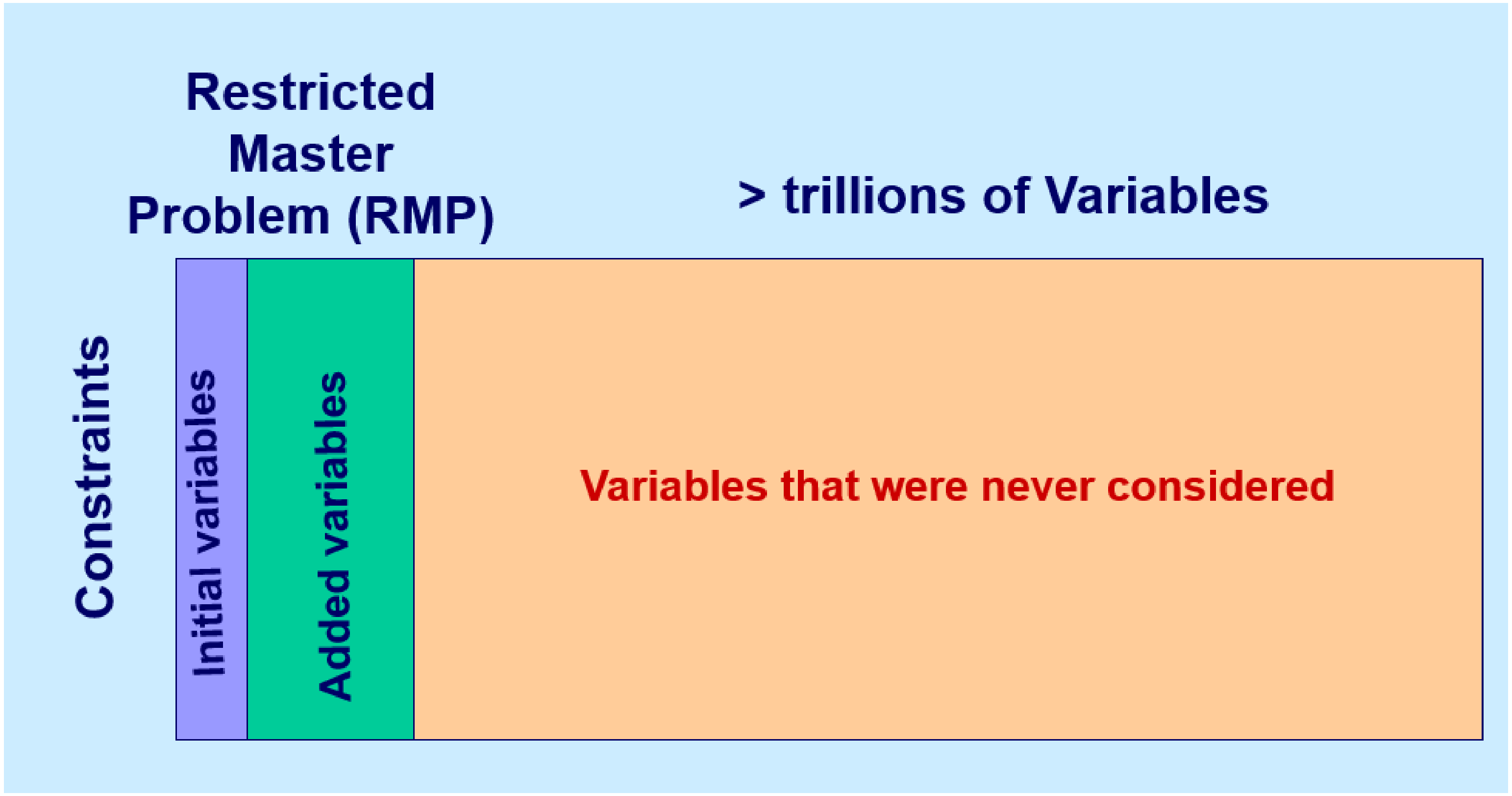
$$m_1 x_1 + m_2 x_2 + m_3 x_3 \le b_4$$

Sub-Problems

$$\min c_i x_i - \pi \times m_i x_i - \mu_i$$

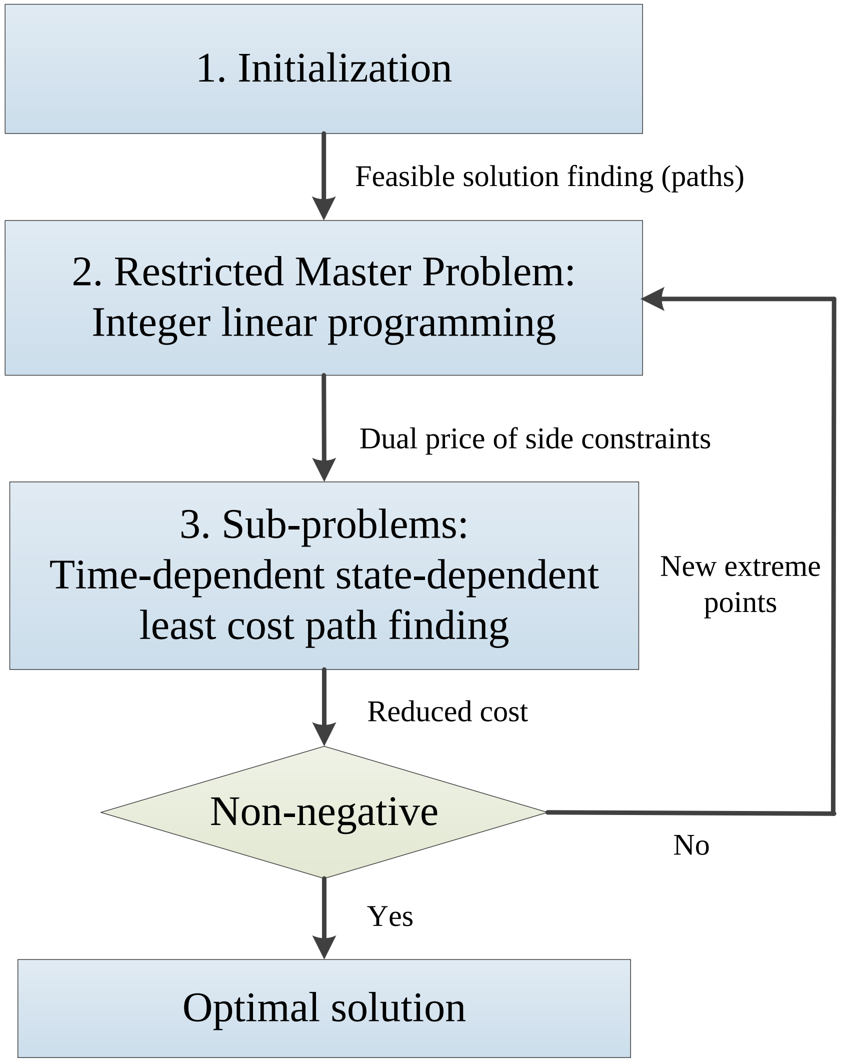
$$a_i x_i = b_i$$

4. Dantzig-Wolfe Decomposition


Restricted Master Problem (RMP)
> trillions of Variables
Constraints
Initial variables
Added variables
Variables that were never considered

https://ocw.mit.edu/courses/sloan-school-of-management/15-082j-network-optimization-fall-2010/lecture-notes/

4. Dantzig-Wolfe Decomposition


1. Initialization
Feasible solution finding (paths)
2. Restricted Master Problem: Integer linear programming
Dual price of side constraints
3. Sub-problems: Time-dependent state-dependent least cost path finding
New extreme points
Reduced cost
Non-negative
No
Yes
Optimal solution

4. Dantzig-Wolfe Decomposition

Restricted master problem

Objective function

$$Min \sum_v \sum_k (c^k_{a,o,d} \times \lambda^k_{a,o,d})$$

Passenger pickup constraint

$$\sum_v \sum_k (\lambda^k_{a,o,d} \times \delta^{a,o,d}_{k,p}) = 1, \forall\, p$$

Space-time arc capacity constraint

$$\sum_{a,o,d} \sum_k (\lambda^k_{a,o,d} \times \beta^k_{a,o,d,(i,j,t,s)}) \leq cap_{i,j,t,s}, \;\; \forall (i,j,t,s)$$

Path weight constraint

$$\sum_k \lambda^{a,o,d}_k = 1, \forall (a,o,d)$$

$$\lambda^{a,o,d}_k = \{0,1\}$$

4. Dantzig-Wolfe Decomposition

Subproblems (TDSDSP)

The sub-problem for each vehicle a:

$$Min\ Z' = \sum_{(i,j,t,s,w,w')}\left(c^a_{i,j,t,s,w,w'} \times x^a_{i,j,t,s,w,w'}\right) - \sum_p \sum_{(i,j,t,s,w,w')\in A(p)}\left(\boxed{\pi_p} \times x^a_{i,j,t,s,w,w'}\right) - \sum_{(i,j,t,s)}\left(\boxed{\mu_{i,j,t,s}} \times \sum_w x^a_{i,j,t,s,w,w'}\right) - \boxed{\omega_a}$$

Flow balance constraint for each vehicle

$$\sum_{i,t,w:(i,j,t,s,w,w')} x^a_{i,j,t,s,w,w'} - \sum_{i,t,w:(j,i,s,t,w',w)} x^a_{j,i,s,t,w',w} = \begin{cases} -1 & j = O(a), s = DT(a), w = [0,0,\dots,0] \\ 1 & j = D(a), s = T, w = [0,0,\dots,0] \\ 0 & otherwise \end{cases} \quad ,\forall a$$

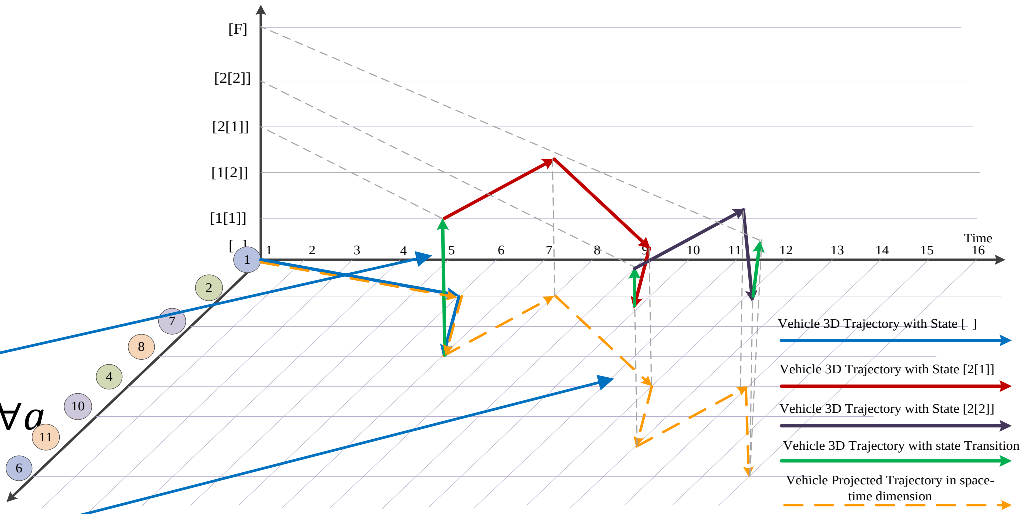
Dual price of of passenger pickup constraints

Dual price of congestion constraints

Dual price of path weight constraints

5. Column-pool based Approximation

Each column is a path with information and connection of passenger trip requests, and vehicle passed space-time arcs



[F]
[2[2]]
[2[1]]
[1[2]]
[1[1]]
[ ]
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16
Time
1
2
7
8
4
10
11
6
Vehicle 3D Trajectory with State [ ]
Vehicle 3D Trajectory with State [2[1]]
Vehicle 3D Trajectory with State [2[2]]
Vehicle 3D Trajectory with state Transition
Vehicle Projected Trajectory in space-time dimension

$$\min \sum_{(o,d,k)} (c^k_{o,d} \times y^k_{o,d})$$

(1) Pickup requests on passenger group q:

$$\sum_{(o,d,k)} (y^k_{o,d} \times \boxed{\delta^q_{o,d,k}}) = g(q), \forall q$$

(2) Road capacity constraints:

$$\sum_{(o,d,k)} (y^k_{o,d} \times \boxed{\delta^{i,j,t,s}_{o,d,k}}) \le cap_{i,j,t,s}, \forall (i,j,t,s)$$

(3) Positive continuous variable (path vehicle flow):

$$y^k_{o,d} \ge 0$$

Compared with arc-based formulation, column-based model greatly reduces the number of variables.

IRA A. FULTON SCHOOLS OF engineering ARIZONA STATE UNIVERSITY

5. Column-pool based Approximation

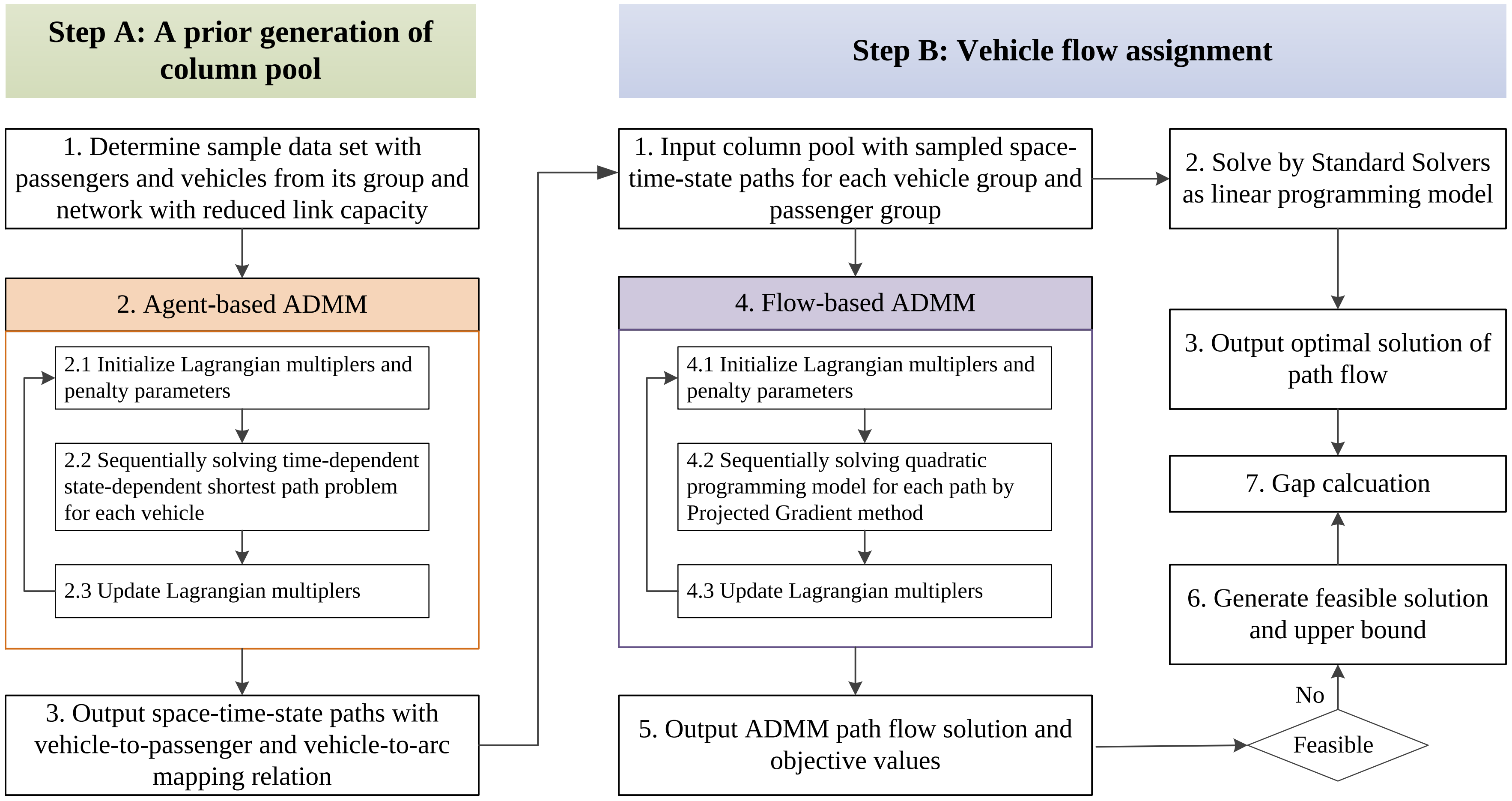
$\delta_{v,k}^{q}$ Incidence matrix of column with passenger trip requests

Column\ trip requests	p1	p2	p3	p4
y1	1	1	0	0
y2	0	1	0	1
y3	0	0	1	1

$\delta_{v,k}^{i,j,t,s}$ Incidence matrix of column with arc

Column\space-time arc	arc 1(i,j,t,s)	arc 2(i,j,t,s)	arc 3(i,j,t,s)	arc 4(i,j,t,s)
y1	1	0	0	1
y2	0	1	1	0
y3	1	0	0	1

5. Column-pool based Approximation

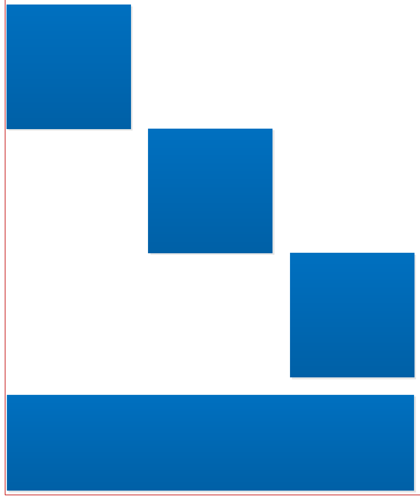

Step A: A prior generation of column pool
1. Determine sample data set with passengers and vehicles from its group and network with reduced link capacity
2. Agent-based ADMM
2.1 Initialize Lagrangian multiplers and penalty parameters
2.2 Sequentially solving time-dependent state-dependent shortest path problem for each vehicle
2.3 Update Lagrangian multiplers
3. Output space-time-state paths with vehicle-to-passenger and vehicle-to-arc mapping relation
Step B: Vehicle flow assignment
1. Input column pool with sampled space-time-state paths for each vehicle group and passenger group
2. Solve by Standard Solvers as linear programming model
3. Output optimal solution of path flow
4. Flow-based ADMM
4.1 Initialize Lagrangian multiplers and penalty parameters
4.2 Sequentially solving quadratic programming model for each path by Projected Gradient method
4.3 Update Lagrangian multipliers
5. Output ADMM path flow solution and objective values
Feasible
No
6. Generate feasible solution and upper bound
7. Gap calcuation

5. Column-pool based Approximation

Alternating Direction Method of Multipliers (ADMM)

Primal Block Angular

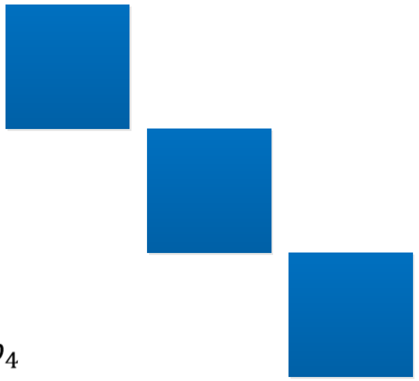
$$\min c_1x_1 + c_2x_2 + c_3x_3$$



$$a_1x_1 = b_1$$
$$a_2x_2 = b_2$$
$$a_3x_3 = b_3$$
$$m_1x_1 + m_2x_2 + m_3x_3 \le b_4$$

Dualization and Augmentation

$$min \quad c_1x_1 + c_2x_2 + c_3x_3 + \lambda\boxed{(m_1x_1 + m_2x_2 + m_3x_3 - b_4)} + \frac{\rho}{2}\boxed{(m_1x_1 + m_2x_2 + m_3x_3 - b_4)^2}$$



$$a_1x_1 = b_1$$
$$a_2x_2 = b_2$$
$$a_3x_3 = b_3$$

Objective function:

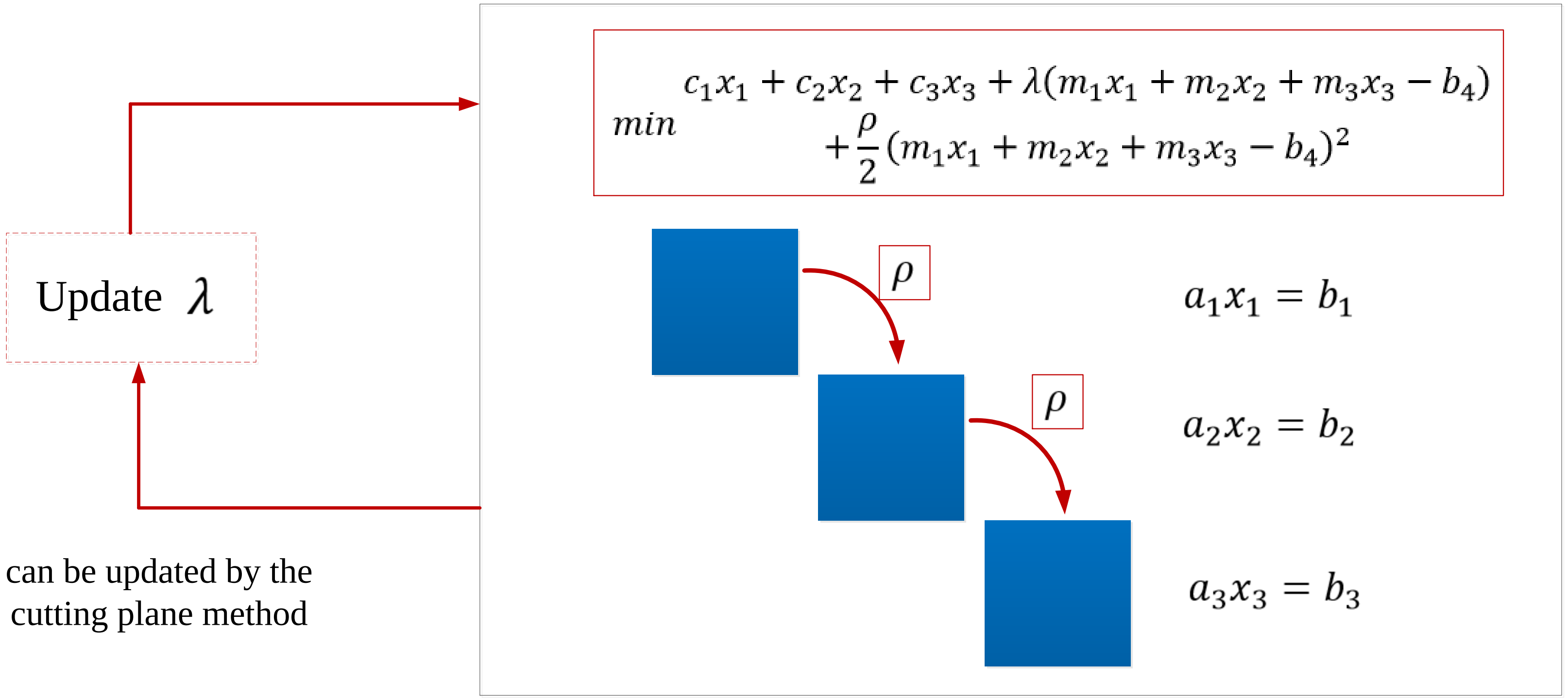
$$L(\boldsymbol{x}, \lambda, \rho) = c_1x_1 + c_2x_2 + c_3x_3 + \lambda(m_1x_1 + m_2x_2 + m_3x_3 - b_4) + \frac{\rho}{2}(m_1x_1 + m_2x_2 + m_3x_3 - b_4)^2$$

Solutions: $x_3^{n+1} = argminL(x_1^{n+1}, x_2^{n+1}, x_3, \lambda^n, \rho)$

Multiplier update: $\lambda^{n+1} = \lambda^n + \rho(m_1x_1^{n+1} + m_2x_2^{n+1} + m_3x_3^{n+1} - b_4)$

5. Column-pool based Approximation

Sequentially solving each subproblem



$min \; c_1x_1 + c_2x_2 + c_3x_3 + \lambda(m_1x_1 + m_2x_2 + m_3x_3 - b_4) + \frac{\rho}{2}(m_1x_1 + m_2x_2 + m_3x_3 - b_4)^2$
Update λ
ρ
ρ
$a_1x_1 = b_1$
$a_2x_2 = b_2$
$a_3x_3 = b_3$

λ can be updated by the cutting plane method

5. Column-pool based Approximation

Arc-based Agent-based formulation

Objective function:

$$Min\ Z = L(\boldsymbol{x^a}, \boldsymbol{\pi_p}, \boldsymbol{\pi_{(i,j,t,s)}}) = \sum_a \sum_{(i,j,t,s,w,w')} \left(c_{i,j,t,s,w,w'} \times x^a_{i,j,t,s,w,w'}\right) + \sum_p [\pi_p \times (\sum_a \sum_{(i,j,t,s,w,w') \in A(p)} (x^a_{i,j,t,s,w,w'} \times \delta^a_{i,j,t,s}) - 1)] + \frac{\rho_1}{2} \sum_p \left[\sum_a \sum_{(i,j,t,s,w,w') \in A(p)} (x^a_{i,j,t,s,w,w'} \times \delta^a_{i,j,t,s}) - 1\right]^2 + \sum_{(i,j,t,s)} [\pi_{(i,j,t,s)} \times (\sum_a \sum_w x^a_{i,j,t,s,w,w'} - cap_{i,j,t,s})] + \frac{\rho_2}{2} \sum_{(i,j,t,s)} \left[\sum_a \sum_w x^a_{i,j,t,s,w,w'} - cap_{i,j,t,s}\right]^2$$

At each iteration:

- Solve each sub-problem sequentially for each vehicle
- Each sub-problem is a time-dependent state-dependent shortest path problem for each vehicle
- Update Lagrangian multipliers for passenger pickup constraints and arc capacity constraints

5. Column-pool based Approximation

ADMM for agent-based model

```
// initialization
Set up initial values for all Lagrangian multiplers and penalty parameters
for $n = 1$ to $n_{max}$ // total number of iterations
    for $a = 1$ to $a_{max}$ //total number of vehicles
        Find the time-dependent state-dependent shortest path for vehicle $a$ with the fixed solution of other vehicles
        Update the network-arc costs based on the new solution of vehicle $a$ for vehicle $a + 1$
    end // vehicle
    Update Lagrangian multipliers of passenger pickup constraints and arc capacity constraints
end // iterations
```

At iteration $n + 1$ of ADMM:

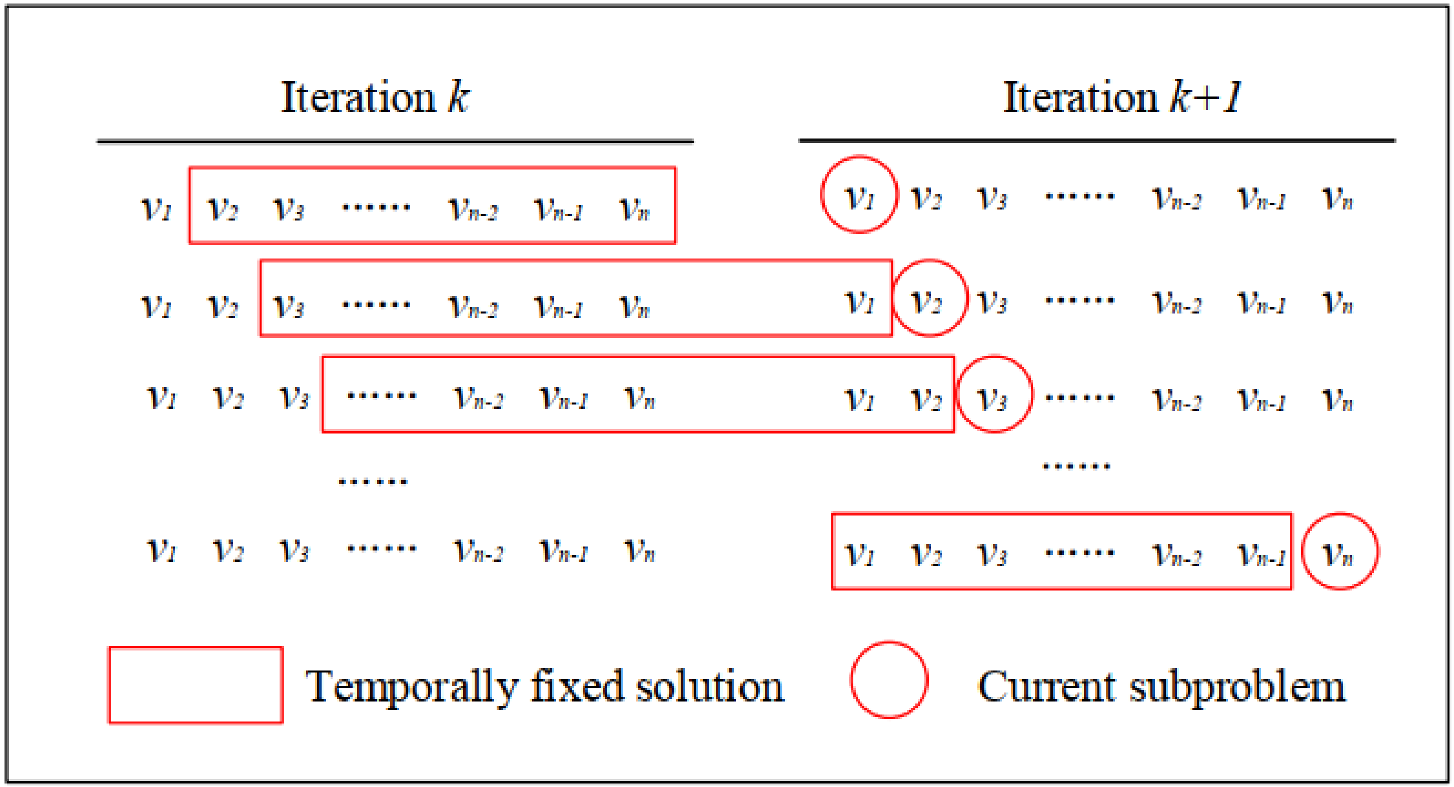
$$x_a^{n+1} = \arg\min \{L(x_1^{n+1}, x_2^{n+1}, \dots, x_a, x_{a+1}^n, \dots, x_{a_{\max}}^n, \boldsymbol{\pi}_p^n, \boldsymbol{\pi}_{i,j,t,s}^n)\}$$

$$\pi_p^{n+1} = \pi_p^n - \rho_1[\textstyle\sum_a \sum_{(i,j,t,s,w,w')\in A(p)}(x_{i,j,t,s,w,w'}^{a,n+1} \times \delta_{i,j,t,s}^a) - 1]$$

$$\pi_{i,j,t,s}^{n+1} = \max\{0, \pi_{i,j,t,s}^n - \rho_2[\textstyle\sum_a \sum_w x_{i,j,t,s,w,w'}^{a,n+1} - cap_{i,j,t,s}]$$

5. Column-pool based Approximation

At each iteration for each vehicle:



Iteration k
Iteration $k+1$
v_1 v_2 v_3 ...... v_{n-2} v_{n-1} v_n
Temporally fixed solution
Current subproblem

5. Column-pool based Approximation

Flow-based path-based ADMM:

Quadratic objective functions:

$$\min \sum_k (c^k \times y^k) + \sum_q (\lambda_q \times [(\sum_k (y^k \times \delta_q^k) - g(q)]) + \frac{\rho_1}{2} \sum_q \left(\left(\sum_k (y^k \times \delta_q^k \right) - g(q) \right)^2 +$$
$$\sum_{i,j,t,s} (\mu_{i,j,t,s} \times [\sum_k (y^k \times \delta^k_{i,j,t,s}) - cap_{i,j,t,s}]) + \frac{\rho_2}{2} \sum_{i,j,t,s} (\sum_k (y^k \times \delta^k_{i,j,t,s}) - cap_{i,j,t,s})^2$$

Its Hessian Matrix can be derived as,

$$H = \begin{vmatrix} \sigma_1 \sum_p \delta_p^1 + \sigma_2 \sum_{i,j,t,s} \delta^1_{i,j,t,s} & \sigma_1 \sum_p \delta_p^1 \delta_p^2 + \sigma_2 \sum_{i,j,t,s} \delta^1_{i,j,t,s} \delta^2_{i,j,t,s} & \cdots & \sigma_1 \sum_p \delta_p^1 \delta_p^{kmax} + \sigma_2 \sum_{i,j,t,s} \delta^1_{i,j,t,s} \delta^{kmax}_{i,j,t,s} \\ \sigma_1 \sum_p \delta_p^1 \delta_p^2 + \sigma_2 \sum_{i,j,t,s} \delta^1_{i,j,t,s} \delta^2_{i,j,t,s} & \sigma_1 \sum_p \delta_p^2 + \sigma_2 \sum_{i,j,t,s} \delta^2_{i,j,t,s} & \cdots & \sigma_1 \sum_p \delta_p^2 \delta_p^{kmax} + \sigma_2 \sum_{i,j,t,s} \delta^2_{i,j,t,s} \delta^{kmax}_{i,j,t,s} \\ \vdots & \vdots & \cdots & \vdots \\ \sigma_1 \sum_p \delta_p^1 \delta_p^{kmax} + \sigma_2 \sum_{i,j,t,s} \delta^1_{i,j,t,s} \delta^{kmax}_{i,j,t,s} & \sigma_1 \sum_p \delta_p^2 \delta_p^{kmax} + \sigma_2 \sum_{i,j,t,s} \delta^2_{i,j,t,s} \delta^{kmax}_{i,j,t,s} & \cdots & \sigma_1 \sum_p \delta_p^{kmax} + \sigma_2 \sum_{i,j,t,s} \delta^{kmax}_{i,j,t,s} \end{vmatrix}$$

Since it is difficult to directly obtain its inverse matrix H^-, especially in large-scale networks, we apply ADMM to decompose the primal problem to sequentially solve the subproblem for each column.

5. Column-pool based Approximation

Flow-based path-based ADMM:

- Solve each variable $y_{o,d}^{k}$ sequentially by ADMM

$$y_k^{n+1} = \arg\min\{L(y_1^{n+1}, y_2^{n+1}, \dots, y_k, y_{k+1}^{n}, \dots, y_{k_{\max}}^{n}, \boldsymbol{\lambda}_q^{n}, \boldsymbol{\mu}_{i,j,t,s}^{n})\}$$

At each iteration of ADMM, Lagrangian multipliers are updated as follows,

Passenger group trip requests:

$$\lambda_q^{n+1} = \lambda_q^{n} + \rho_1((\textstyle\sum_k (y_k^n \times \delta_q^k) - g(q))$$

Arc capacity constraints:

$$\mu_{i,j,t,s}^{n+1} = \max\{0, \mu_{i,j,t,s}^{n} + \rho_2(\textstyle\sum_k (y_k^n \times \delta_{i,j,t,s}^{k}) - cap_{i,j,t,s})\}$$

5. Column-pool based Approximation

Projected Gradient Method (Rosen, 1960) to solve each subproblem

$$y_k^{n+1} = \arg\min\{L(y_1^{n+1}, y_2^{n+1}, \dots, y_k, y_{k+1}^n, \dots, y_{k_{\max}}^n, \boldsymbol{\lambda}_q^n, \boldsymbol{\mu}_{i,j,t,s}^n)\}$$

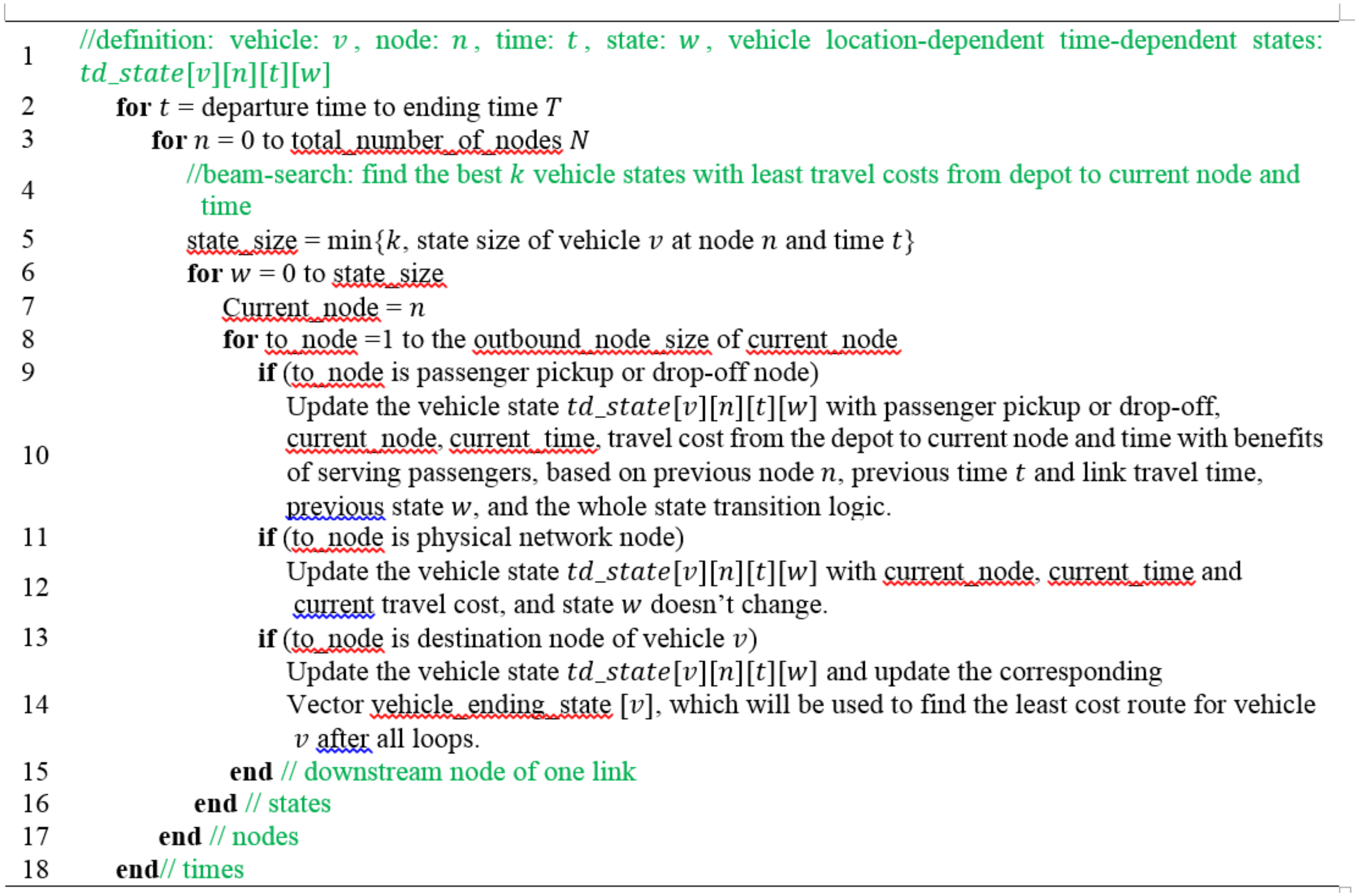
$$y_k^{n+1} = \max\{0, y_k^n - \frac{1}{s} \times L(y_k^n)'\}$$

Where $L(y_k^n)' = c^k + \sum_q \lambda_q \times \delta_q^k + \rho_1 \left(\sum_q \delta_q^k \left(\left(\sum_k (y_k^n \times \delta_q^k\right) - g(q)\right)\right) + \sum_{i,j,t,s} \mu_{i,j,t,s} \times \delta_{i,j,t,s}^k +$ $\rho_2(\sum_{i,j,t,s} \delta_{i,j,t,s}^k (\sum_k (y_k^n \times \delta_{i,j,t,s}^k) - cap_{i,j,t,s}))$, and $s = \frac{\partial^2 L(x)}{\partial x^2} = \rho_1 \sum_q \delta_q^k + \rho_2 \sum_{i,j,t,s} \delta_{i,j,t,s}^k$

Projected Gradient Method has been used in solving the path-based nonlinear programming models in equilibrium traffic assignment (Larsson and Patriksson, 1992; Jayakrishnan et al, 1994; Florian et al., 2009), and it is more efficient, compared with arc-based nonlinear programming models, but needs more memory use.

5. Column-pool based Approximation

Beam-search algorithm for finding the time-dependent state-dependent shortest paths:



```
//definition: vehicle: $v$, node: $n$, time: $t$, state: $w$, vehicle location-dependent time-dependent states: $td\_state[v][n][t][w]$
for $t$ = departure time to ending time $T$
    for $n$ = 0 to total_number_of_nodes $N$
        //beam-search: find the best $k$ vehicle states with least travel costs from depot to current node and time
        state_size = min{$k$, state size of vehicle $v$ at node $n$ and time $t$}
        for $w$ = 0 to state_size
            Current_node = $n$
            for to_node =1 to the outbound_node_size of current_node
                if (to_node is passenger pickup or drop-off node)
                    Update the vehicle state $td\_state[v][n][t][w]$ with passenger pickup or drop-off, current_node, current_time, travel cost from the depot to current node and time with benefits of serving passengers, based on previous node $n$, previous time $t$ and link travel time, previous state $w$, and the whole state transition logic.
                if (to_node is physical network node)
                    Update the vehicle state $td\_state[v][n][t][w]$ with current_node, current_time and current travel cost, and state $w$ doesn't change.
                if (to_node is destination node of vehicle $v$)
                    Update the vehicle state $td\_state[v][n][t][w]$ and update the corresponding Vector vehicle_ending_state [$v$], which will be used to find the least cost route for vehicle $v$ after all loops.
            end // downstream node of one link
        end // states
    end // nodes
end// times
```

6. Discussion and Preliminary Experiments

Discussion: path marginal cost calculation

System-impact of adjusting one vehicle routing:

- ❑ System marginal vehicle travel cost
- ❑ System marginal passenger service benefit/cost

[Time window]
[Time window]

In this queuing system:

❑ Waiting time for individual: 4 min

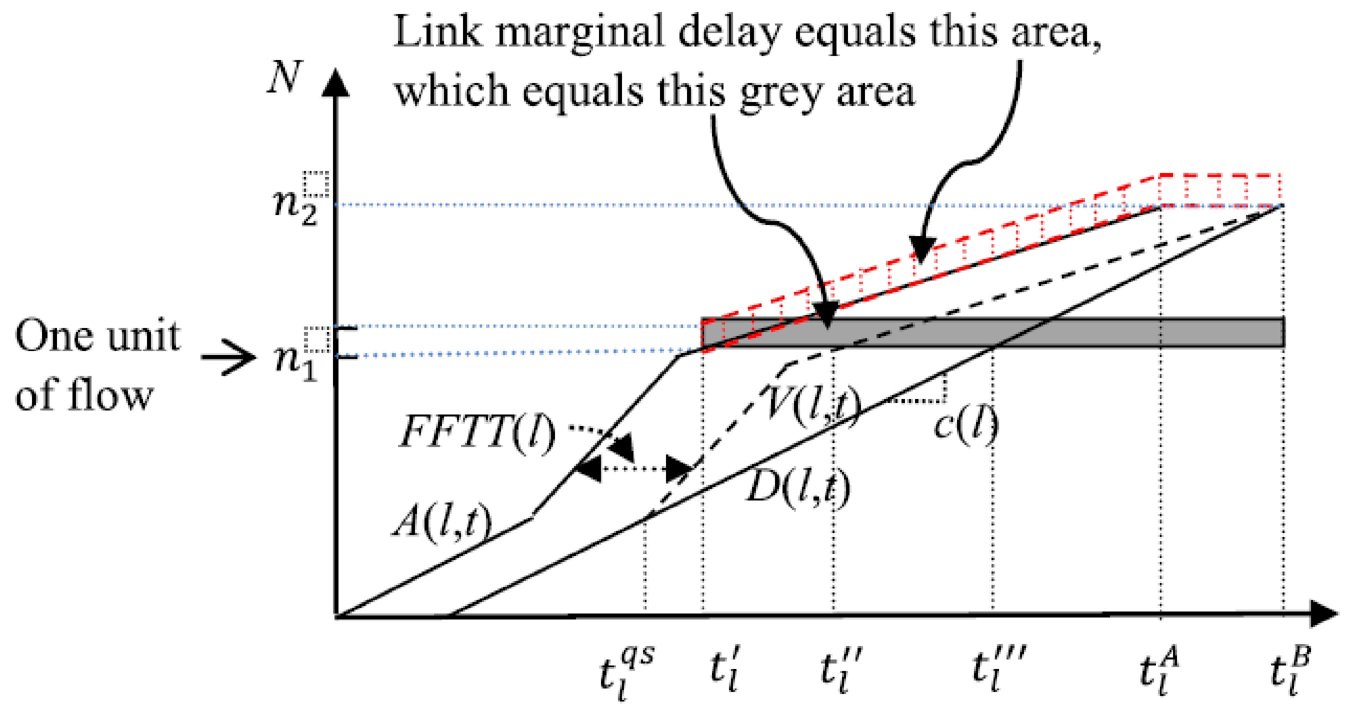
After adding one more person in the queue:

❑ Societal travel time: additional 4 min for each person behind: +16 min, and the waiting time of added person is 4 min, so the system marginal waiting time is 20 min.

❑ Societal service benefit: some persons may not be served in their preferred time window and it decreases the service benefit.

6. Discussion and Preliminary Experiments

Marginal cost calculation in system optimal dynamic traffic assignment (SODTA)



Ghali and Smith (1995)

Ghali, M.O. and Smith, M.J., 1995. A model for the dynamic system optimum traffic assignment problem. Transportation Research Part B: Methodological, 29(3), pp.155-170.

6. Discussion and Preliminary Experiments

Step 1: Build virtual pickup and drop-off links in physical traffic networks, and its service pricing is converted to generalized link travel cost

Step 2: find one initial solution as the input

Step 3: Perform network loading within a space-time-state network

3.1 use cumulative arrival and departure counts to derive the link marginal travel cost.

3.2 update the marginal service link benefit of passengers (not served or served by multiple vehicles)

Step 4: find the new least-marginal-cost route for each vehicles, and go to step 3; otherwise, stop.

Path marginal cost is probably related to the **Lagrangian multipliers** in ADMM and the **dual prices** in Dantzig-Wolfe decomposition.

IRA A. FULTON SCHOOLS OF engineering ARIZONA STATE UNIVERSITY

6. Discussion and Preliminary Experiments

Capture queue spillback:

Inflow arc capacity constraint:

$$\sum_w x_{i,j',t-FFTT_{i,j}+1,t,w,w'} \leq Cap_{i,j',t-FFTT_{i,j}+1,t}, \forall (i,j') \in L_{inflow}, \forall t \tag{11}$$

Outflow arc capacity constraint:

$$\sum_w x_{j',j,t,t+1,w,w'} \leq y_{j',j,t,t+1}, \forall (j',j) \in L_{outflow}, \forall t \tag{12}$$

Outflow arc capacity balance constraint at points without merger and diverge:

$$y_{j',j,t,t+1} \leq Cap_{j,i,t+1,s} \tag{13}$$

Outflow arc capacity balance constraint at merger points:

$$\sum_{(j',t)} y_{j',j,t,t+1} \leq Cap_{j,i,t+1,s}, \forall (j,t+1) \in A_m \tag{14}$$

Outflow arc capacity balance constraint at diverge points:

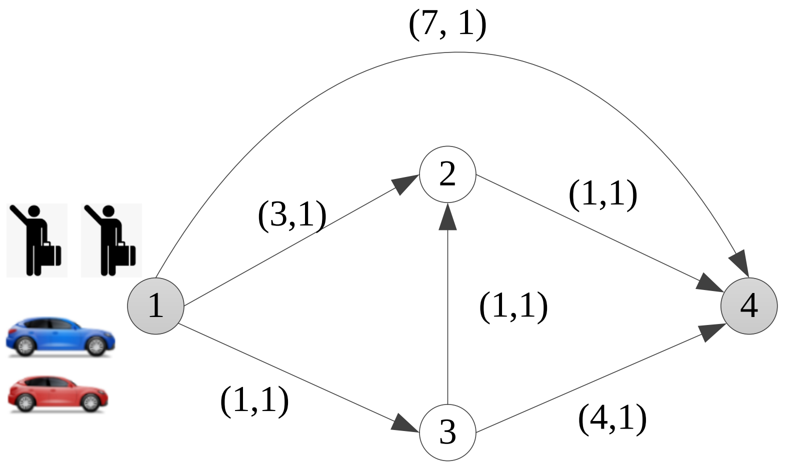
$$y_{j',j,t,t+1} \leq \sum_{(i,s)} Cap_{j,i,t+1,s}\ \forall (j,t+1) \in A_d \tag{15}$$

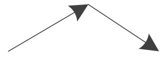
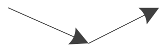
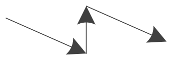
Link storage capacity constraint:

$$\sum_w x_{j',j',t-1,t,w,w'} + \sum_w \sum_{s=t-FFTT_{i,j'}}^{t-1} x_{i,j',s,t,w,w'} \leq Len_{i,j'} \times n_{i,j'} \times Jam_{i,j'}, \forall (i,j') \in L_{inflow}, \forall t \tag{16}$$

6. Discussion and Preliminary Experiments

- There are 2 vehicles, and each vehicle picks up one passenger from origin node 1 to destination node 4.
- Our goal is to minimize the total vehicle travel cost by using Dantzig-Wolfe decomposition approach.



Path ID	Node Sequence	Path Cost	Path Trajectory
Path 1	1→2→4	4	
Path 2	1→3→4	5	
Path 3	1→4	7	
Path 4	1→3→2→4	3	

(link cost, link capacity)

6. Discussion and Preliminary Experiments

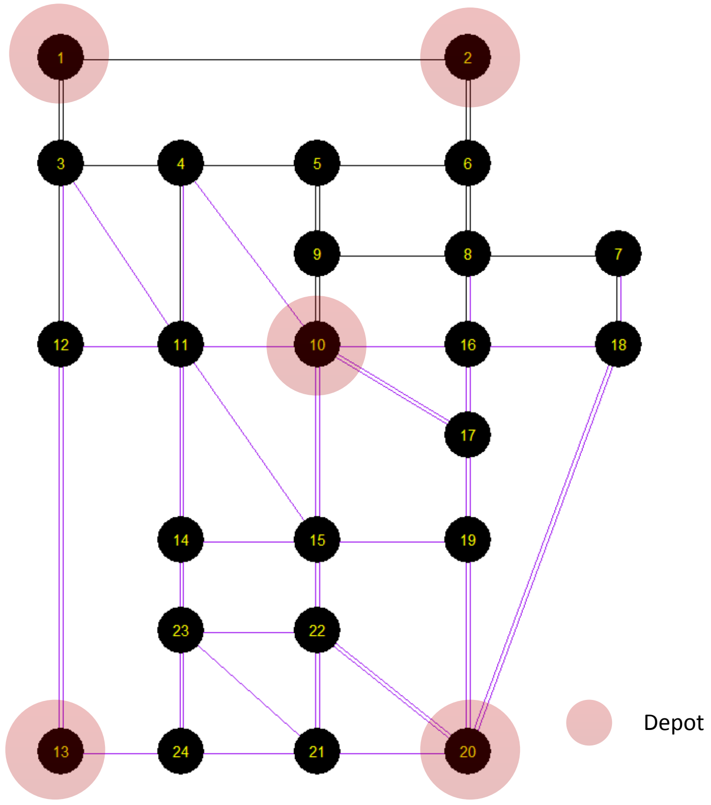
Variable λ_i is the weight of path i of total demand; $\mu_{i,j}$ is the dual price of capacity constraint of link (i, j); π is the dual price of path flow weight constraint.

	Sub problem	**New column: path 4**
Iteration 1	Master problem	$\lambda_4 = 1, \mu_{1,3} = -1, \mu_{1,2} = 0, \mu_{3,2} = 0, \mu_{2,4} = 0, \mu_{3,4} = 0, \mu_{1,4} = 0, \pi = 2$
	Sub problem	New column: path 1
Iteration 2	Master problem	$\lambda_1 = 1, \mu_{2,4} = -1, \mu_{1,2} = 0, \mu_{3,2} = 0, \mu_{1,3} = 0, \mu_{3,4} = 0, \mu_{1,4} = 0, \pi = 2$
	Sub problem	New column: path 3
Iteration 3	Master problem	$\lambda_4 = 0.5, \lambda_3 = 0.5, \mu_{1,3} = -1, \mu_{2,4} = -3, \mu_{1,2} = 0, \mu_{3,2} = 0, \mu_{3,4} = 0, \mu_{1,4} = 0, \pi = 14$
	Sub problem	New column: path 2
Iteration 4	Master problem	$\lambda_1 = 0.5, \lambda_2 = 0.5, \mu_{1,3} = -2, \mu_{2,4} = -2, \mu_{1,2} = -1, \mu_{3,2} = 0, \mu_{3,4} = 0, \mu_{1,4} = 0, \pi = 14$

The reduced cost is $4 + 5 - (-2) - (-2) - (-1) - 14 = 0$ and reach the optimal solution.

6. Discussion and Preliminary Experiments

Requests with pickup and drop-off and time windows under capacitated networks



Sioux Falls Network

# of nodes	24
# of links	84
# of trip requests (pickup and drop-off with time windows)	30
# of available autonomous vehicles	30
# of depots	5
optimization time horizon (time unit)	110
Vehicle capacity (person)	1

Source code: https://github.com/TonyLiu2015/VRPLite-DW

6. Discussion and Preliminary Experiments

Initial feasible solution

Vehicle_No	Passenger_No	Vehicle_No	Passenger_No	Vehicle_No	Passenger_No
1	[15]	11	[20]	21	[23]
2	[8]	12	[26]	22	[25]
3	[1]	13	[16]	23	[22]
4	[7]	14	[18]	24	[19]
5	[9]	15	[2]	25	[4]
6	[11]	16	[10]	26	[5]
7	[29]	17	[3]	27	[24]
8	[28]	18	[12]	28	[14]
9	[17]	19	[27]	29	[13]
10	[21]	20	[30]	30	[6]

6. Discussion and Preliminary Experiments

Dantzig-Wolfe decomposition algorithm solution:

- ❑ Each passenger has specific pickup and drop-off location and time windows
- ❑ The vehicle benefit of serving one passenger is 20
- ❑ The vehicle waiting cost is the half of the waiting time

	Number of required vehicles	Total travel cost
Initial solution	30	1096
vehicle carrying capacity is 1	27	967.5
vehicle carrying capacity is 2	25	869.5

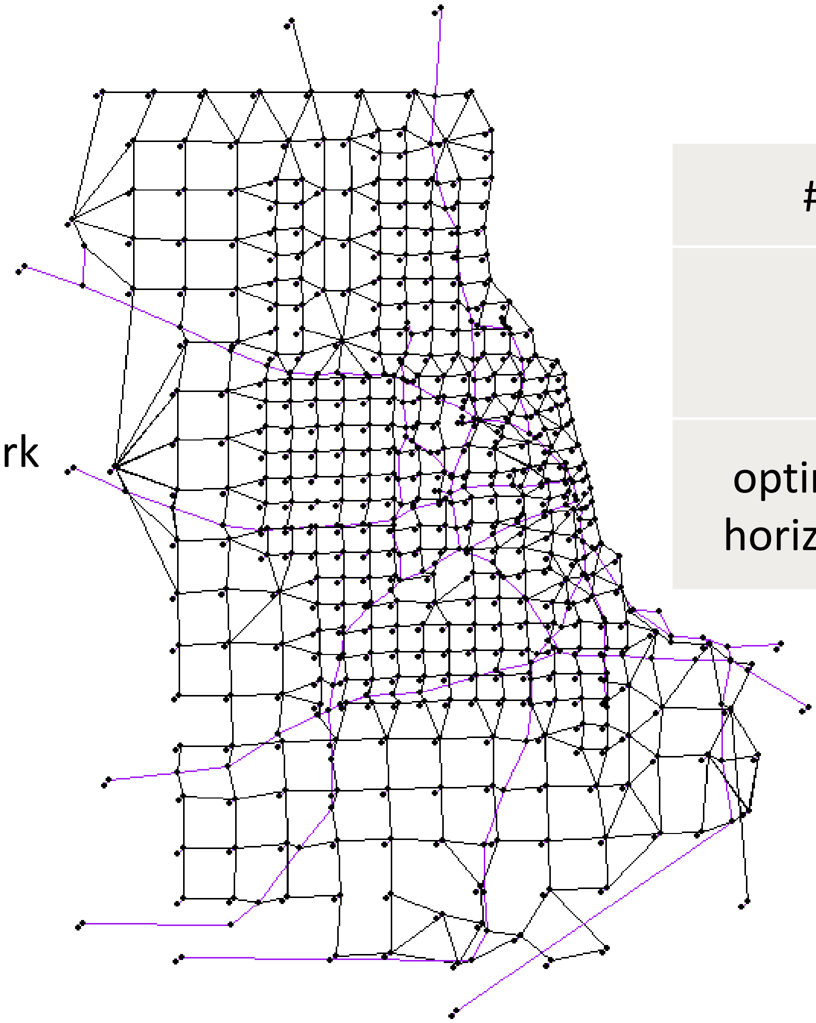
Take **vehicle 9** as an example:

- ❑ In initial solution: picks up passenger 17 -> drops off passenger 17;
- ❑ Vehicle carrying capacity is 1: picks up passenger 17 -> drops off passenger 17-> picks up passenger 29 -> drops off passenger 29
- ❑ Vehicle carrying capacity is 2: picks up passenger 17 -> drops off passenger 17-> picks up passenger 30-> picks up passenger 29-> drops off passenger 29-> drops off passenger 30

6. Discussion and Preliminary Experiments

Requests with pickup only and time windows under capacitated networks

Chicago network



# of nodes	1320
# of links	5431
optimization time horizon (time unit)	60

6. Discussion and Preliminary Experiments

Step A: Prior generation of column pool

Scenario 1: Sample data set

- 10 pairs of vehicle groups and passenger groups.
- Each pair has 243 vehicles and 387 passengers trip requests
- The space-time arc capacity in each minute is 5.
- Vehicle carrying capacity is 1
- 2430 binary variables and 332,160 constraints

Scenario 2: Sample data set

- 20 pair of vehicle groups and passengers groups.
- Each pair has 243 vehicles and 387 passengers trip requests
- The space-time arc capacity in each minute is 5.
- Vehicle carrying capacity is 1
- 4860 binary variables and 338,460 constraints

6. Discussion and Preliminary Experiments

Solution from Agent-based ADMM

Scenario 1:

- 1789 vehicles find their paths/columns to serve 1084 passengers
- 23,357 space-time arcs are generated based on vehicles' space-time paths
- Computation time: about 70 seconds each iteration

Scenario 2:

- 3686 vehicles find their paths/columns to serve 2226 passengers
- 36,454 space-time arcs are generated based on vehicles' space-time paths
- Computation time: about 140 seconds each iteration

Remark: each passenger has a specific pickup location, time window and destination, and vehicle can only pick up passengers within a same pair of groups.

6. Discussion and Preliminary Experiments

Step B: Flow-based ADMM implemented by C++

Experiment 1:

- 1084 passenger groups and each passenger group has 4 passenger trip requests
- The space-time arc capacity in each minute is 35
- 1789 positive continuous variables/columns and 24,441 constraints
- Computation time: 700 CPU seconds for running 250 iterations

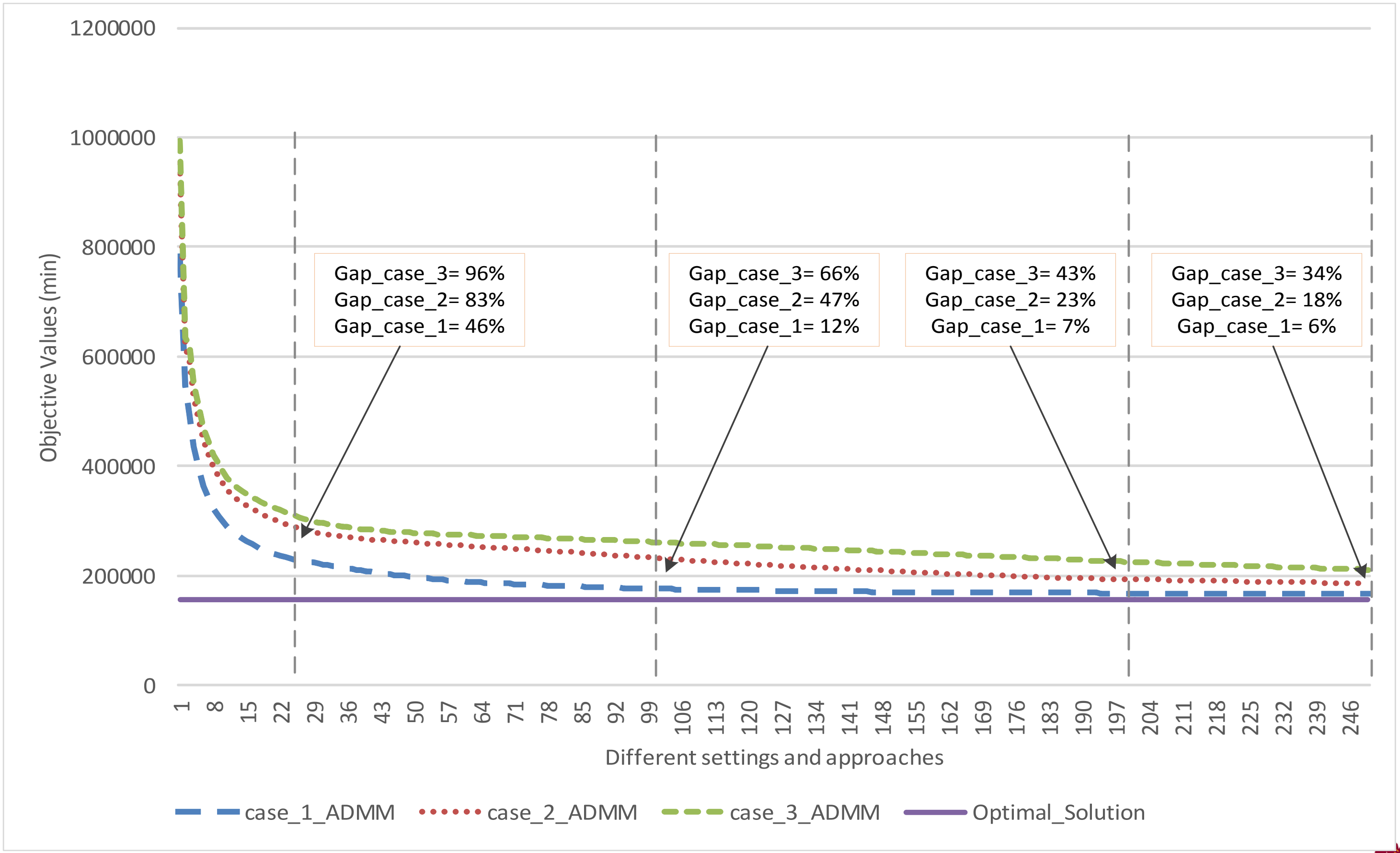
Experiment 2:

- 2226 passenger groups and each passenger group has 2 passenger trip requests
- The space-time arc capacity in each minute is 35
- 3686 positive continuous variables/columns and 38,680 constraints
- Computation time: 2735 CPU seconds to finish 250 iterations

6. Discussion and Preliminary Experiments

Experiment 1

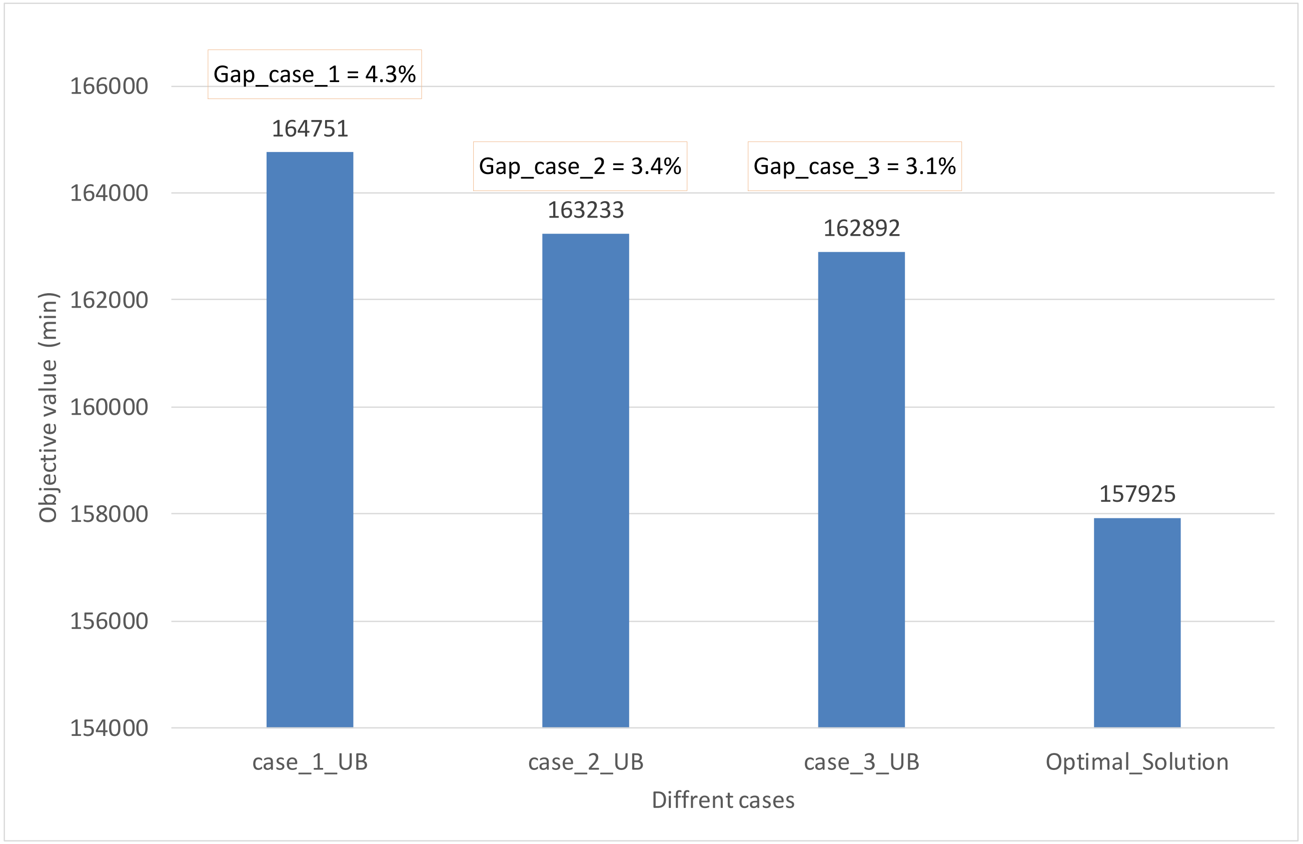
Case 1: $\rho_1 = 3$ and $\rho_2 = 1$; Case 2: $\rho_1 = 3$ and $\rho_2 = 3$; Case 3: $\rho_1 = 3$ and $\rho_2 = 5$.



Solution of each iteration of ADMM in three cases and CPLEX

6. Discussion and Preliminary Experiments

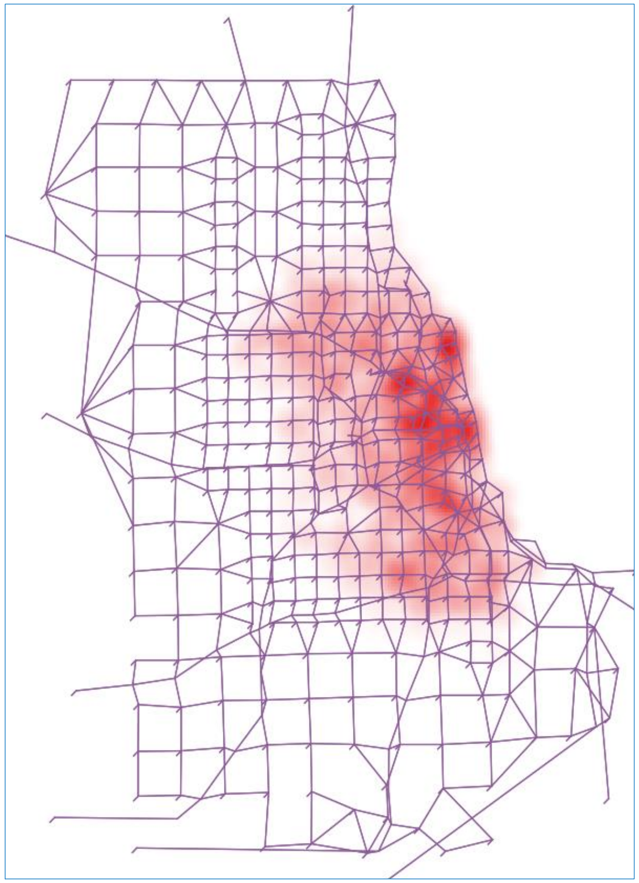
Experiment 1

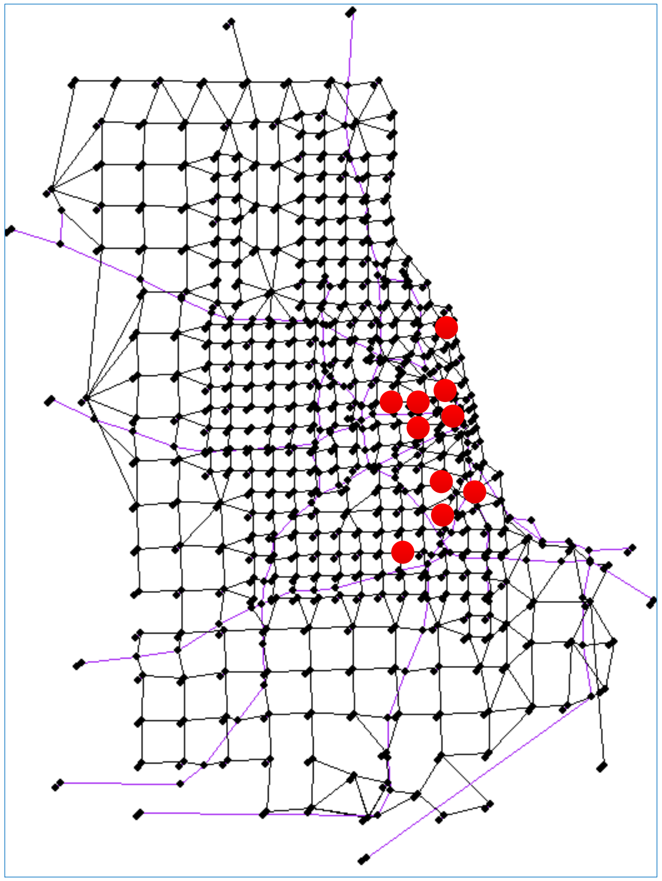


Comparison of objective values of upper bound and CPLEX

6. Discussion and Preliminary Experiments

Experiment 1


(a) The heat map on waiting flows in experiment 1


(b) Top 10 of congested nodes in experiment 1

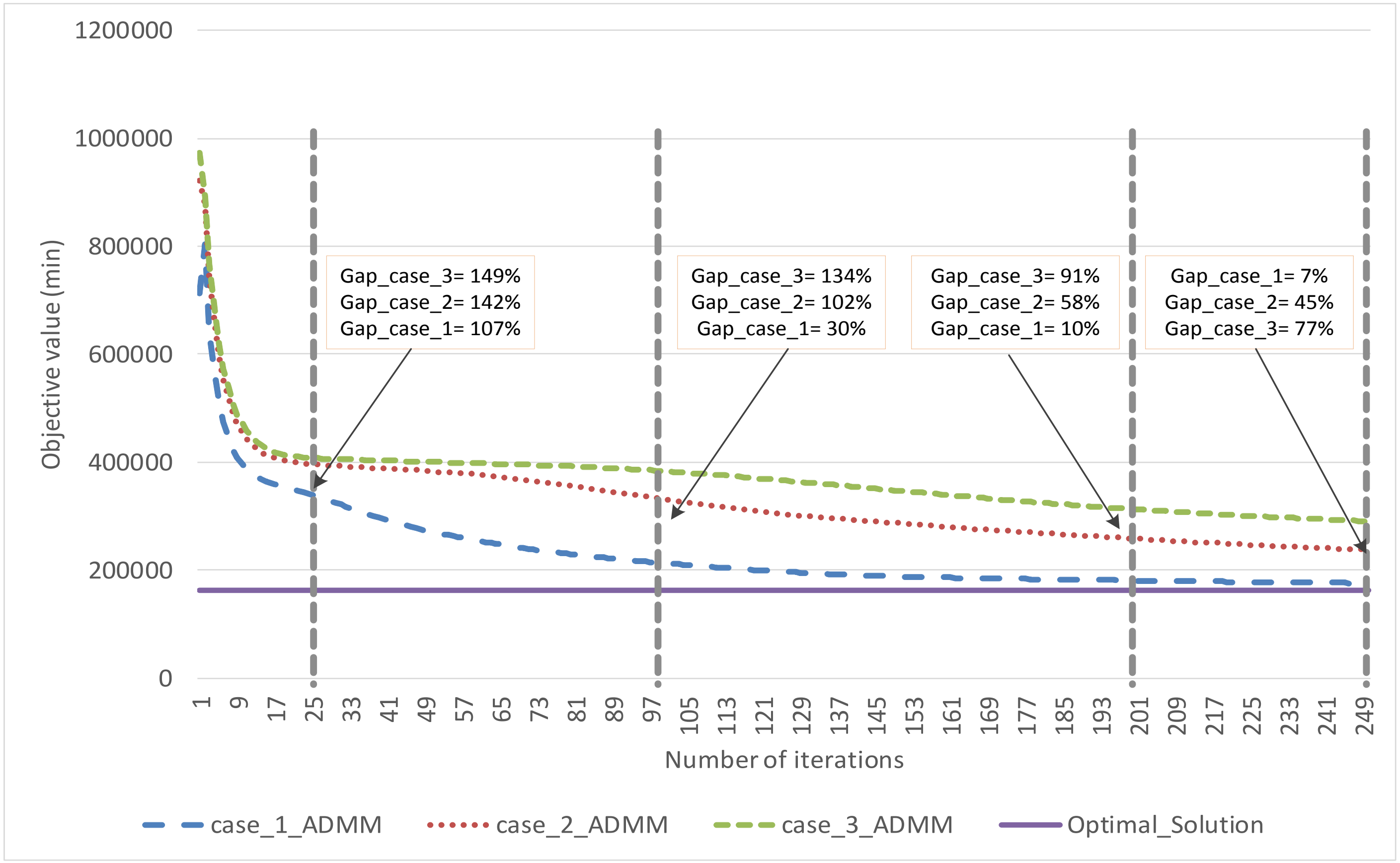
Visualization of congested nodes in experiment 1

IRA A. FULTON SCHOOLS OF engineering
ARIZONA STATE UNIVERSITY

6. Discussion and Preliminary Experiments

Experiment 2

Case 1: $\rho_1 = 3$ and $\rho_2 = 1$; Case 2: $\rho_1 = 3$ and $\rho_2 = 3$; Case 3: $\rho_1 = 3$ and $\rho_2 = 5$.

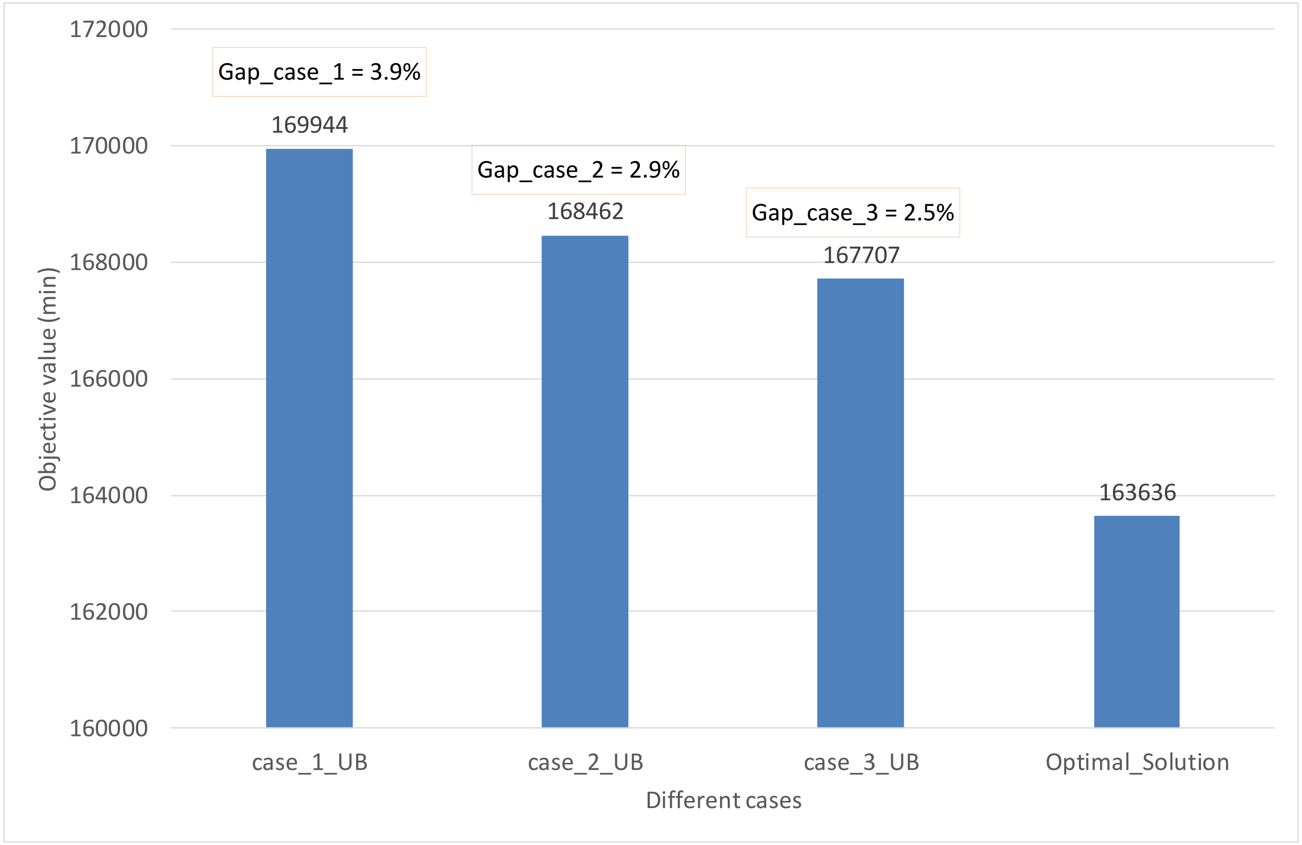


Solution of each iteration of ADMM in three cases and CPLEX

6. Discussion and Preliminary Experiments

Experiment 2

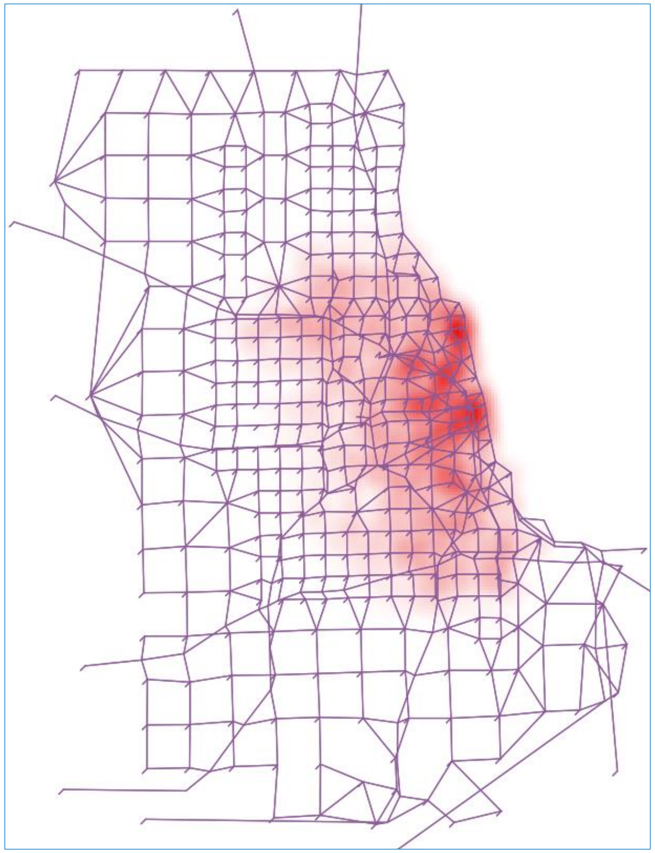
Case 1: $\rho_1 = 3$ and $\rho_2 = 1$; Case 2: $\rho_1 = 3$ and $\rho_2 = 3$; Case 3: $\rho_1 = 3$ and $\rho_2 = 5$.

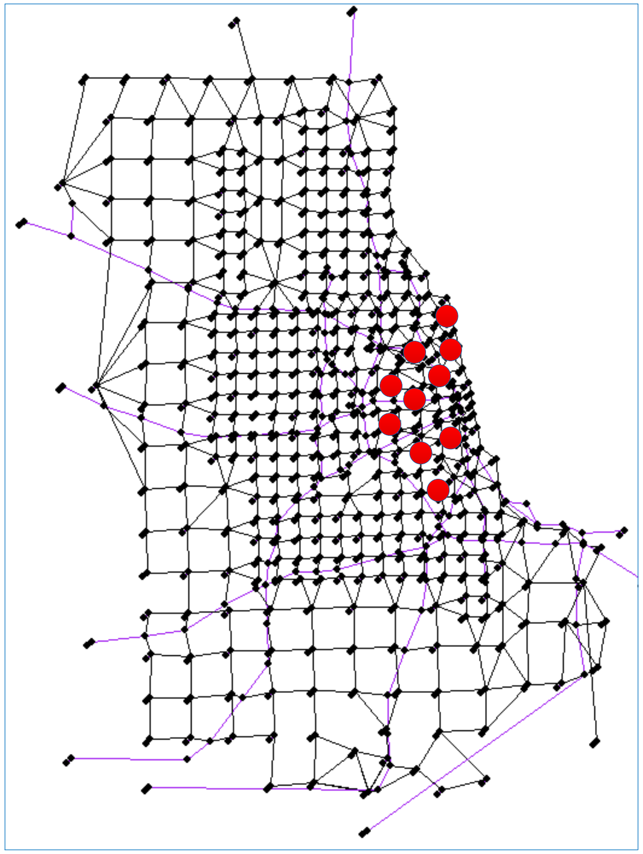


Comparison of objective values of upper bound and CPLEX

6. Discussion and Preliminary Experiments

Experiment 2

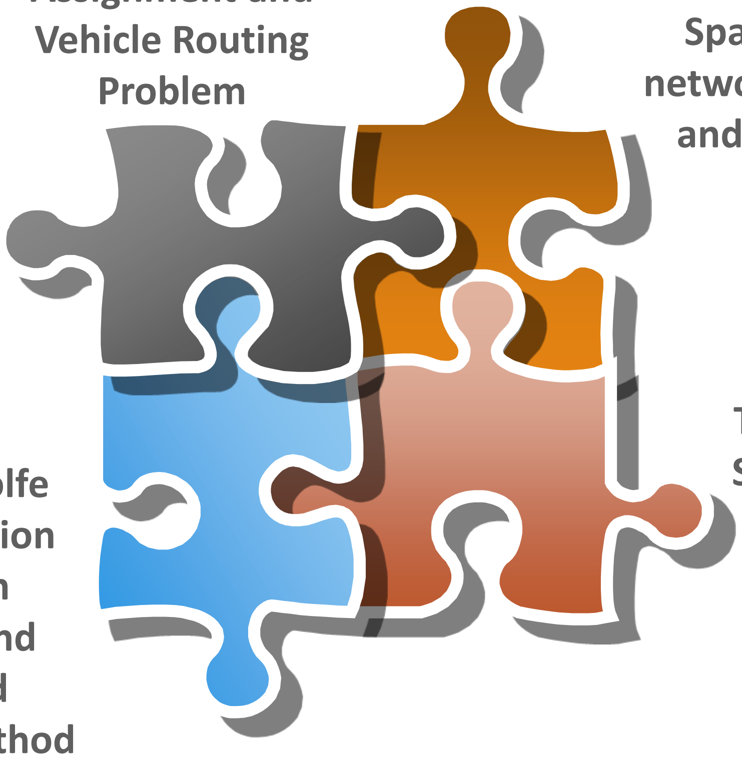

(a) The heat map on waiting flows in experiment 2


(b) Top 10 of congested nodes in experiment 2

Visualization of congested nodes in experiment 2

7. Summary

Required knowledge:



Dynamic Traffic Assignment and Vehicle Routing Problem

Space-time-state network construction and network flow model

Time-dependent State-dependent shortest path problem

Dantzig-Wolfe decomposition algorithm , ADMM and Projected Gradient method

Selected References

Space-time-state network flow models and vehicle routing problem:

1. Mahmoudi, M. and Zhou, X., 2016. Finding optimal solutions for vehicle routing problem with pickup and delivery services with time windows: A dynamic programming approach based on state–space–time network representations. Transportation Research Part B: Methodological, 89, 19-42.
2. Liu, J., Kang, J., Zhou, X., Pendyala, R., 2018. Network-oriented household activity pattern problem for system optimization. Transportation Research Part C 94, 250-269
3. Zhou, X., Tong, L., Mahmoudi, M., Zhuge, L., Yao, Y., Zhang, Y., Shang, P., Liu, J. and Shi, T., 2018. Open-source VRPLite Package for Vehicle Routing with Pickup and Delivery: A Path Finding Engine for Scheduled Transportation Systems. Urban Rail Transit, 4(2), 68-85.

Dynamic Traffic Assignment and Traffic flow model:

4. Lu, C.C., Liu, J., Qu, Y., Peeta, S., Rouphail, N.M. and Zhou, X., 2016. Eco-system optimal time-dependent flow assignment in a congested network. *Transportation Research Part B: Methodological*, *94*, pp.217-239.
5. Zhou, X. and Taylor, J., 2014. DTALite: A queue-based mesoscopic traffic simulator for fast model evaluation and calibration. *Cogent Engineering*, *1*(1), p.961345.

Dantzig-Wolfe Decomposition algorithm

6. https://en.wikipedia.org/wiki/Dantzig%E2%80%93Wolfe_decomposition

Alternating Direction Method of Multipliers (ADMM)

7. https://web.stanford.edu/~boyd/papers/pdf/admm_slides.pdf

THANK YOU

IRA A. FULTON SCHOOLS OF engineering ARIZONA STATE UNIVERSITY