Avoid the Overflow of Disaster Victims around the Railway Station

- via Adaptive Management of Passengers' Routes and Train Schedule -

Chiba University
Hino Yoko, Arai Sachiyo

2015.3.25 RailTokyo 2015

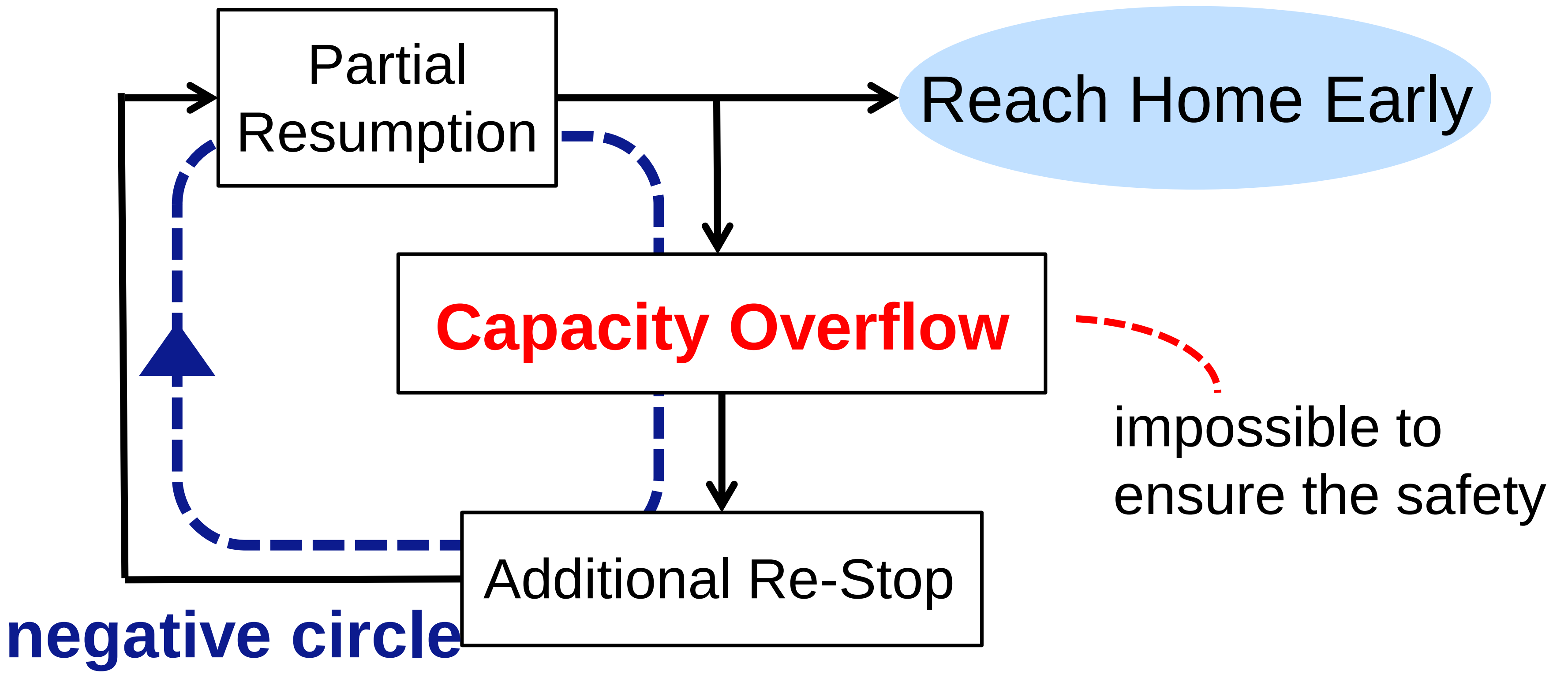
Outline

Background

2011.3.11 the Great East Japan Earthquake (Japan)

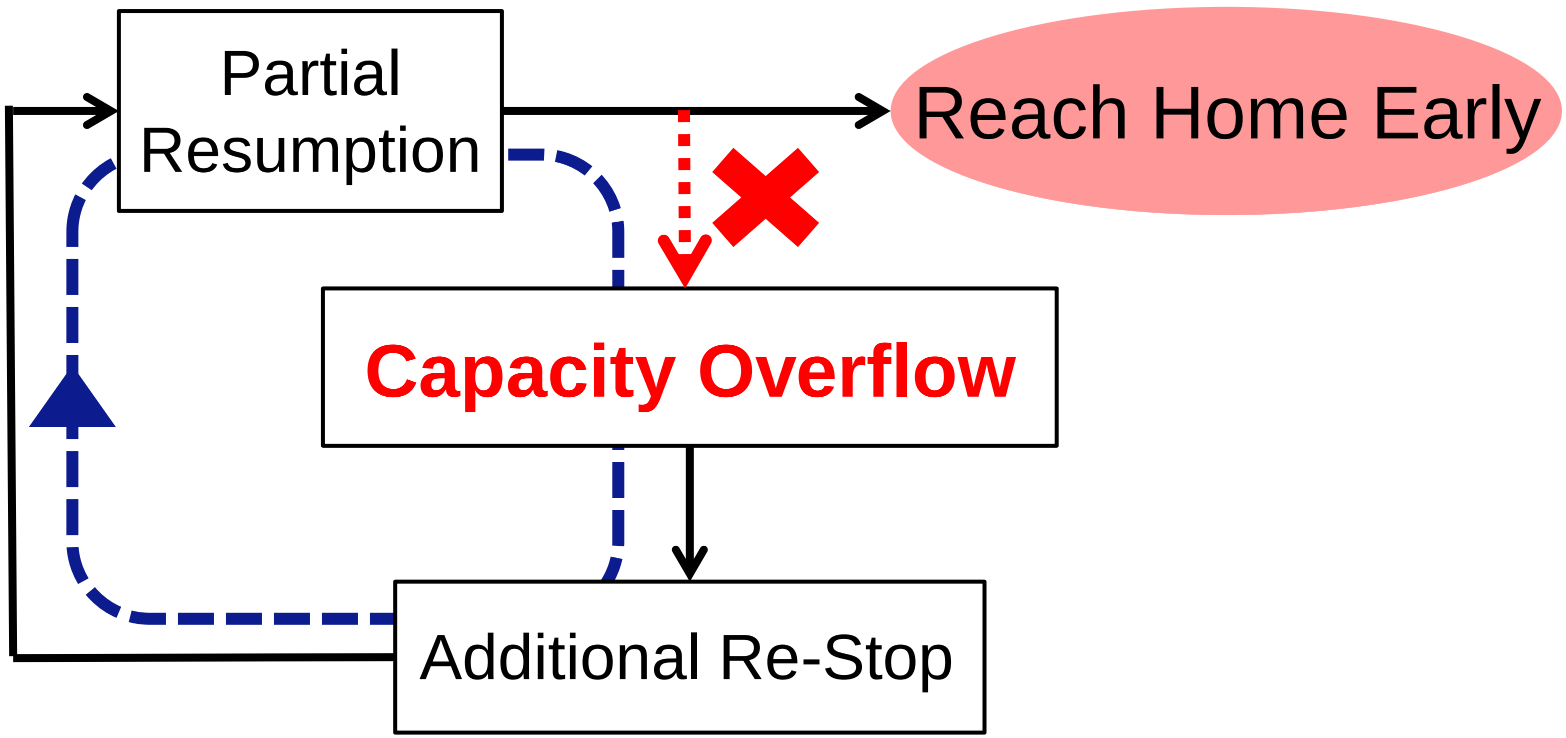
Caused by Nature	Caused by Human
▸ unavailable Major Train Line ▸ cannot reach home ▸ emerge Stranded Commuter **cannot avoid**	▸ resumed partial line ▸ rush to the station ▸ overflow the station's capacity ▸ cause additional re-stop ▸ worsen Stranded Commuter

Objective



impede the recovery

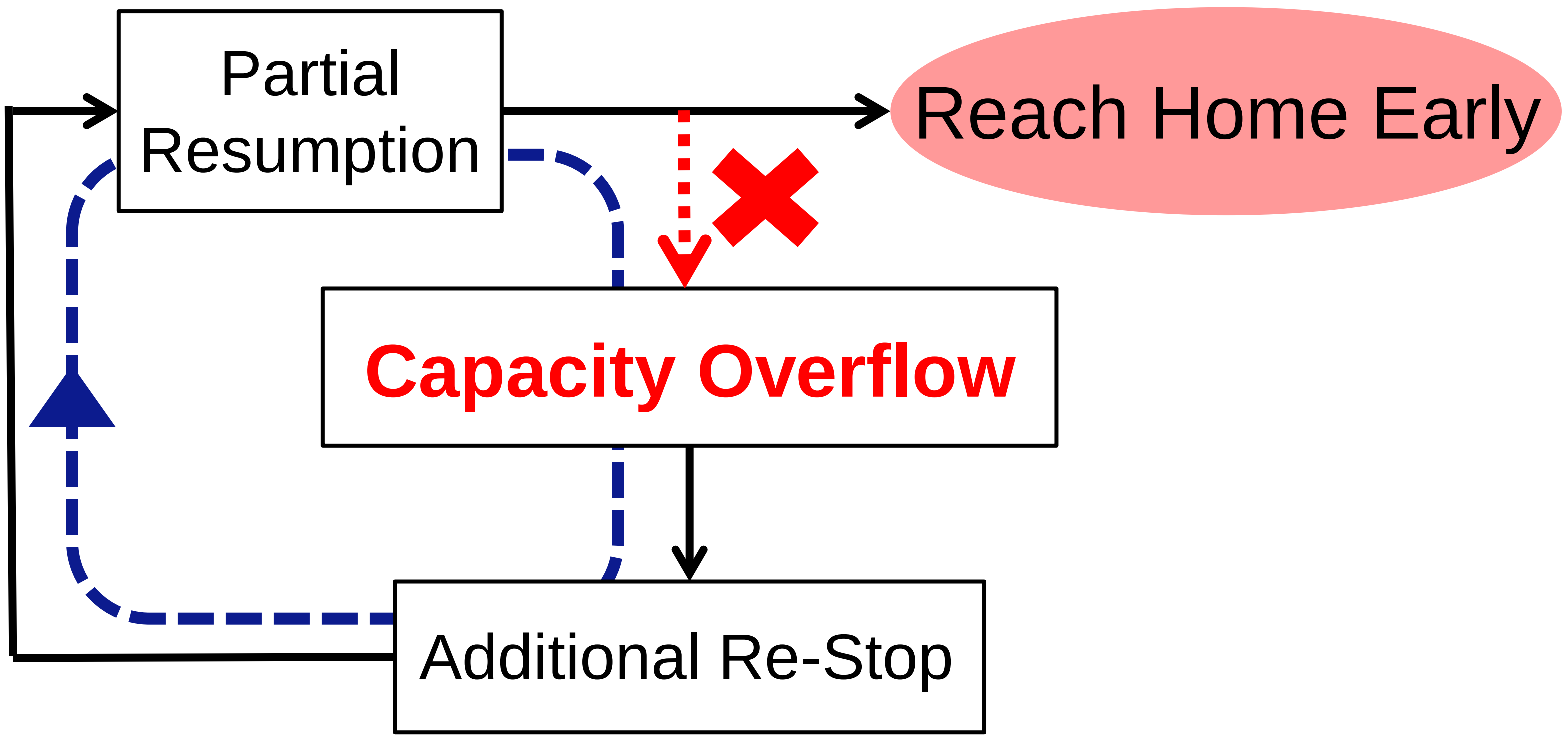
Objective



- **Hardware** measures

expand the station ← carry a cost

Objective



- **Software** measures

Change { **1. Passengers' Route** / **2. Train Schedule** } situationally

Definition

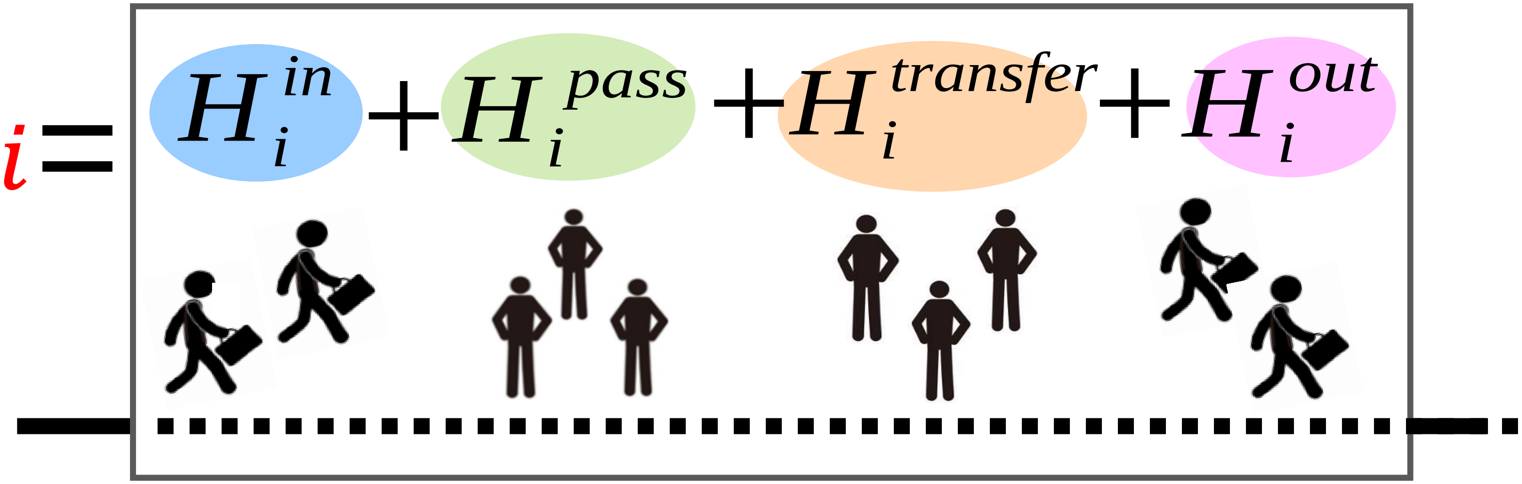
- calculate the number of passengers in the station

Station s_i

come in as Origin

getting out at Destination

$$n_i = H_i^{in} + H_i^{pass} + H_i^{transfer} + H_i^{out}$$



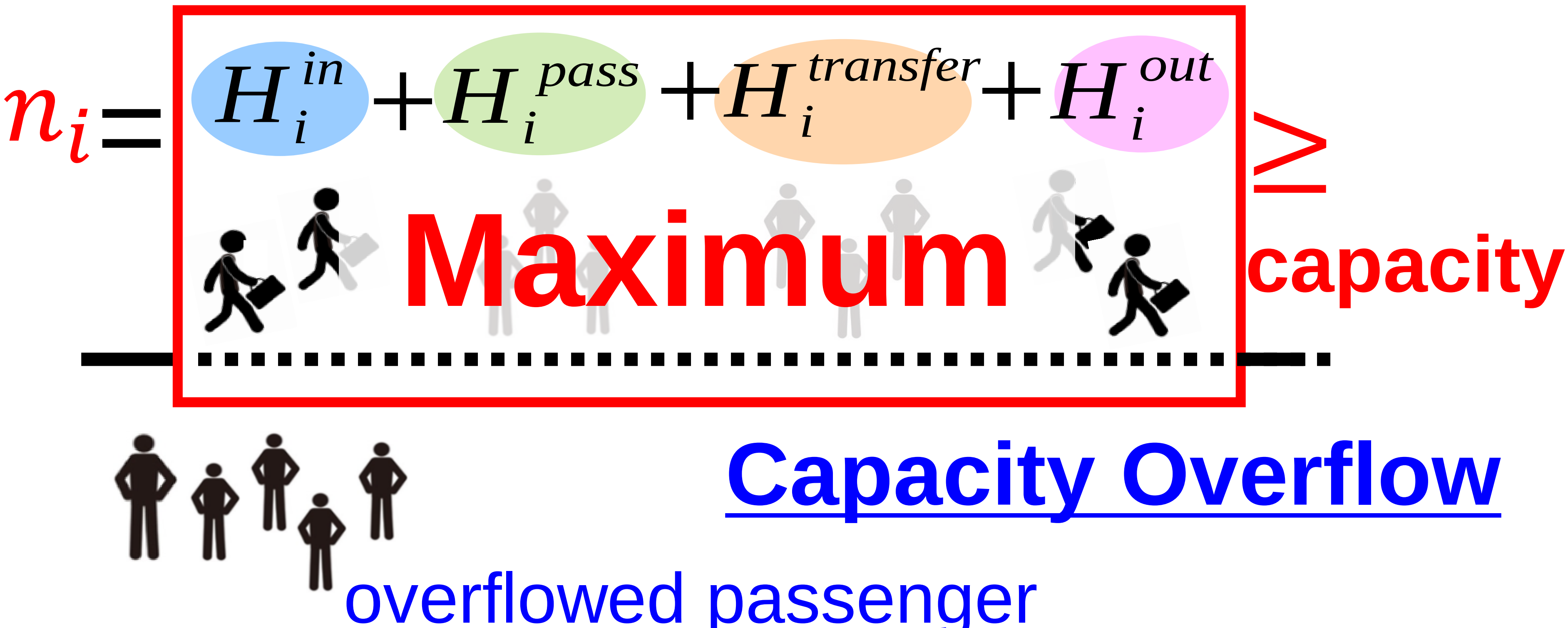
Passed the prior train because of full

Waiting to transfer

Definition

- calculate the number of passengers in the station

Station s_i



Preliminary Experiment

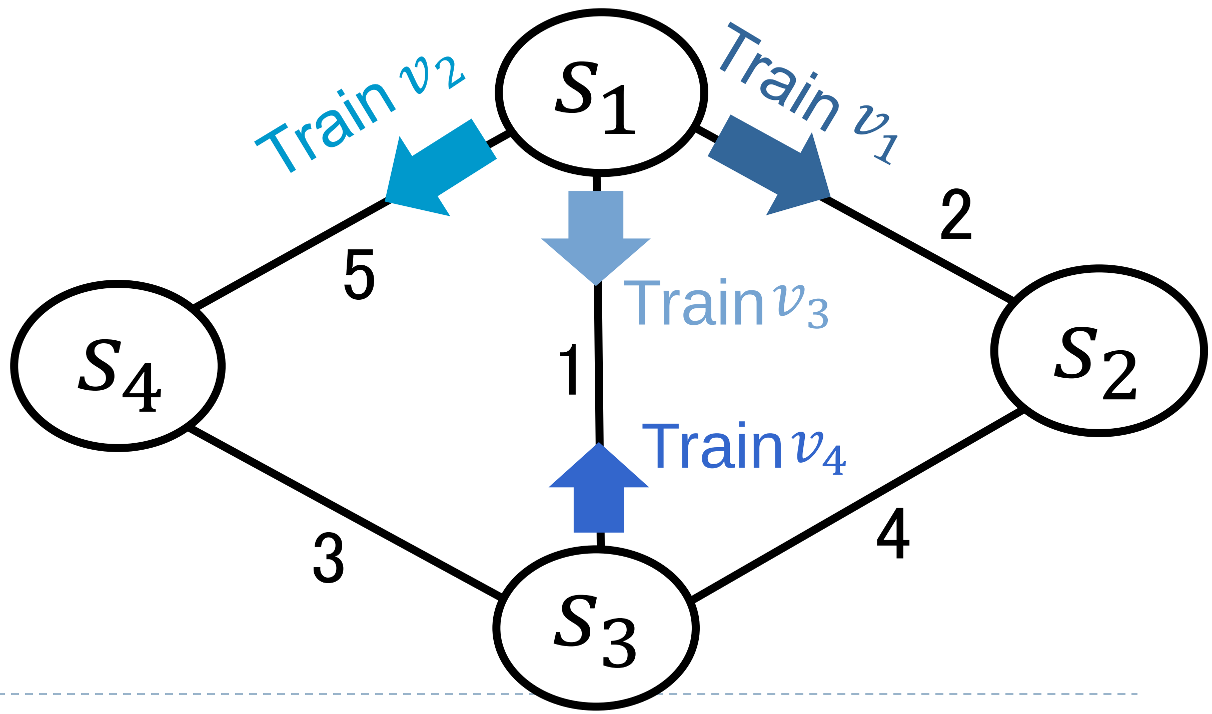
- simulate passengers’ translation in normal train service
- estimate the number of passengers in each time
- make the maximum value as capacity

- <u>Railway Network</u>

○ Station

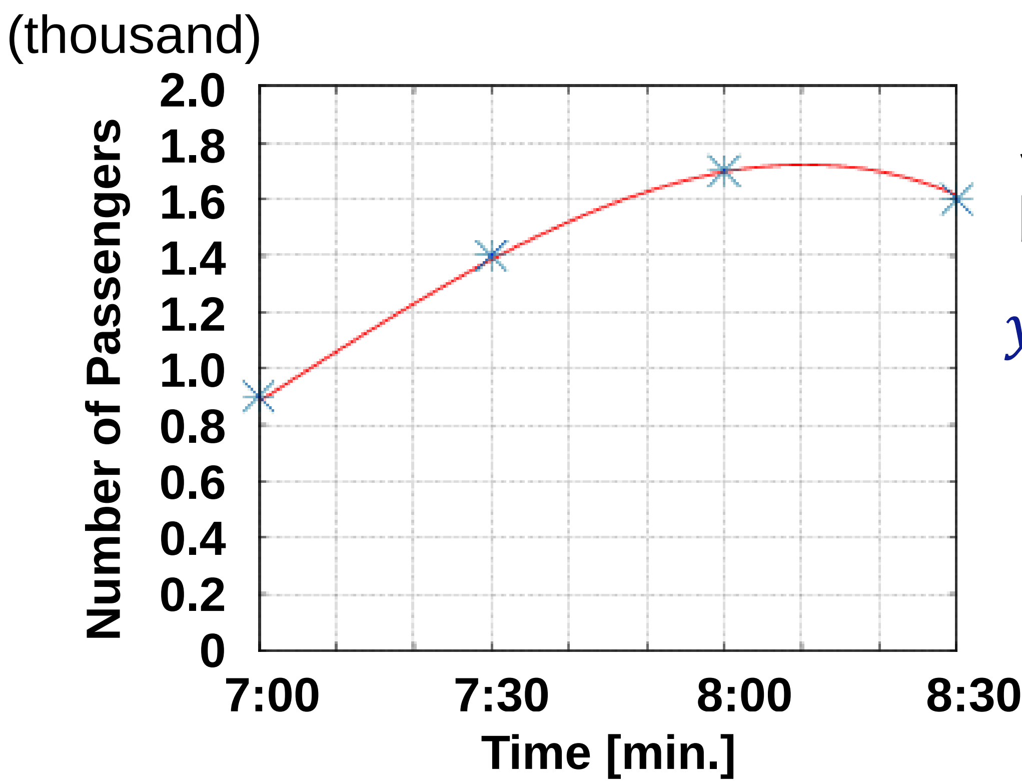
— Train line

Cost of the link
: Required time
to go through



Preliminary Experiment

- Passengers' Settings
  - move through the shortest route
  - follow actual data of Shibuya Station's inflow



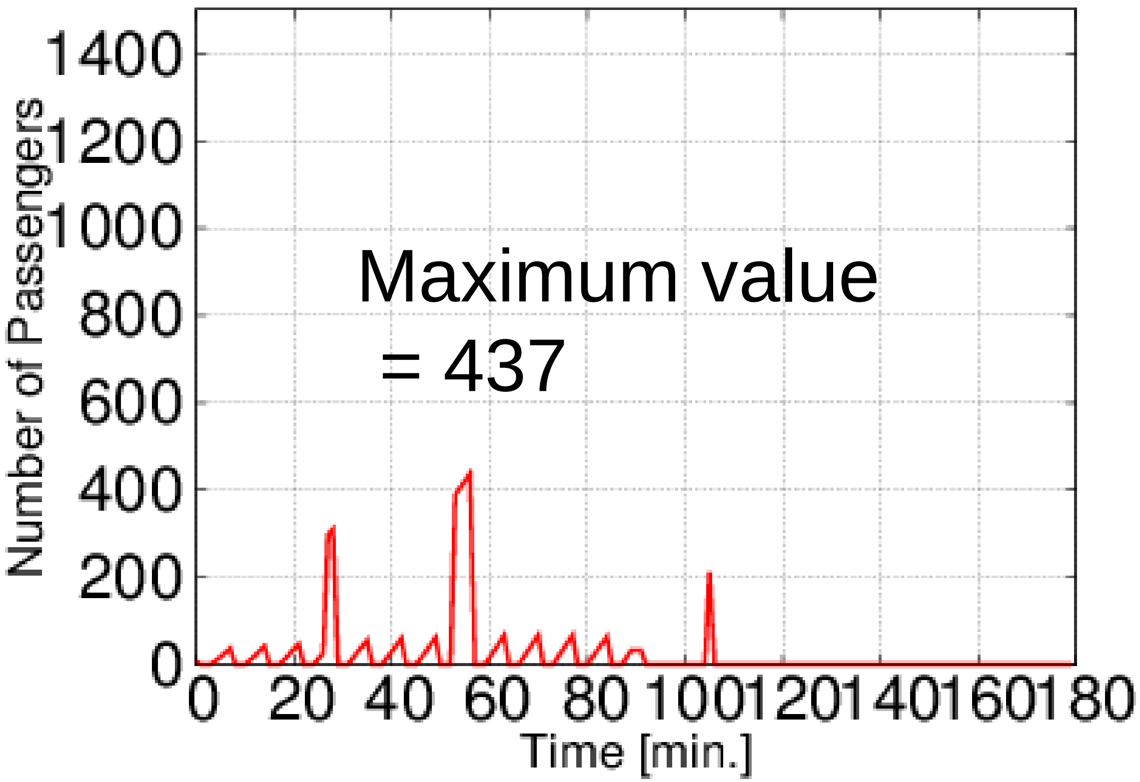
Set up approximate curve by least square method

$$y = -0.0123x^3 + 0.037x^2 + 177.74x + 8822.2$$

http://www.mlit.go.jp/common/001001548.pdf (accessed April 20)

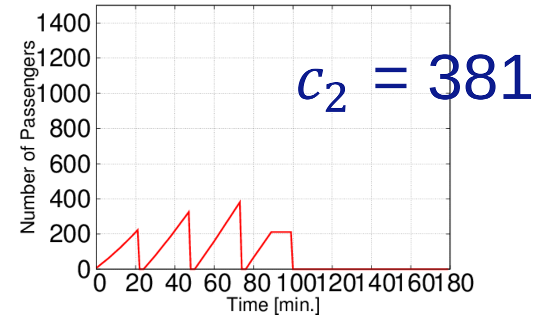
Result : Preliminary Experiment

< Station s_1 >

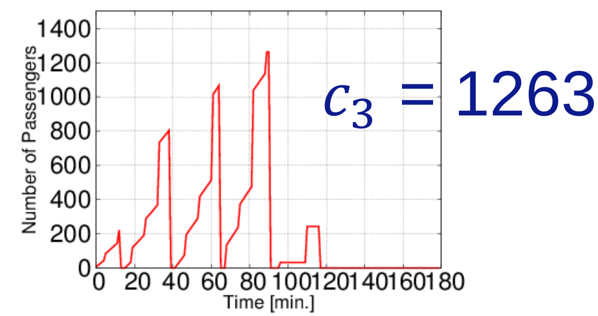


decide Maximum value as the capacity

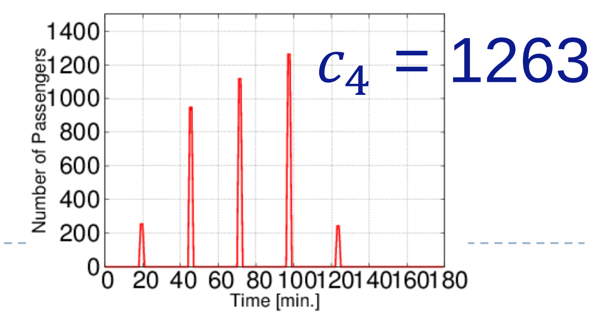
< Station s_2 >



< Station s_3 >

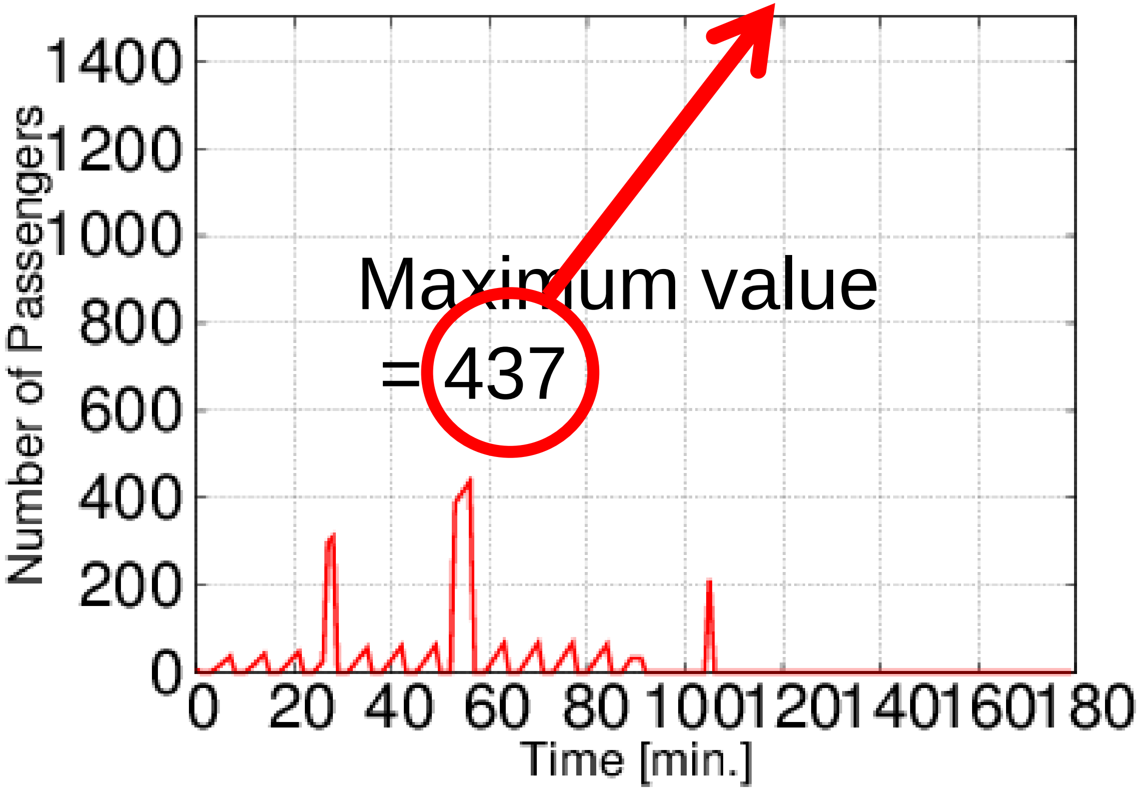


< Station s_4 >



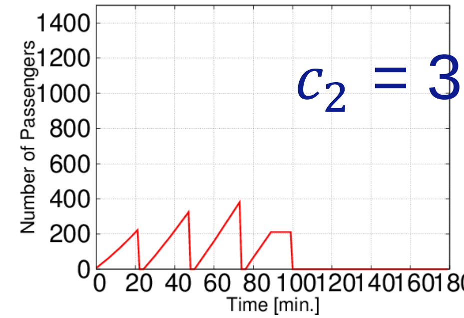
Result : Preliminary Experiment

< Station s_1 > c_1 = 437



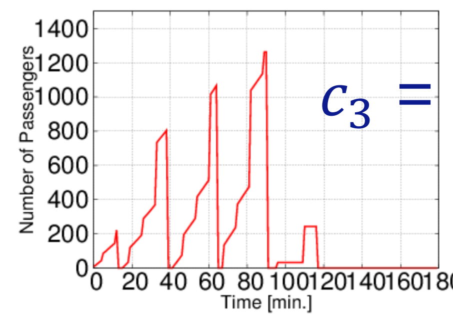
decide Maximum value as the capacity

< Station s_2 >



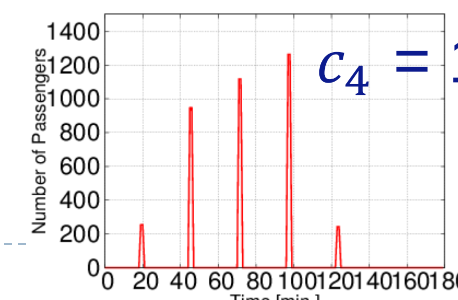
c_2 = 381

< Station s_3 >



c_3 = 1263

< Station s_4 >



c_4 = 1263

1. Passengers' Route Assignment

- Objective :

  change the **Passengers' route** not to overflow the capacity
- formulate as Constraint Satisfaction Problem
  - Constraint

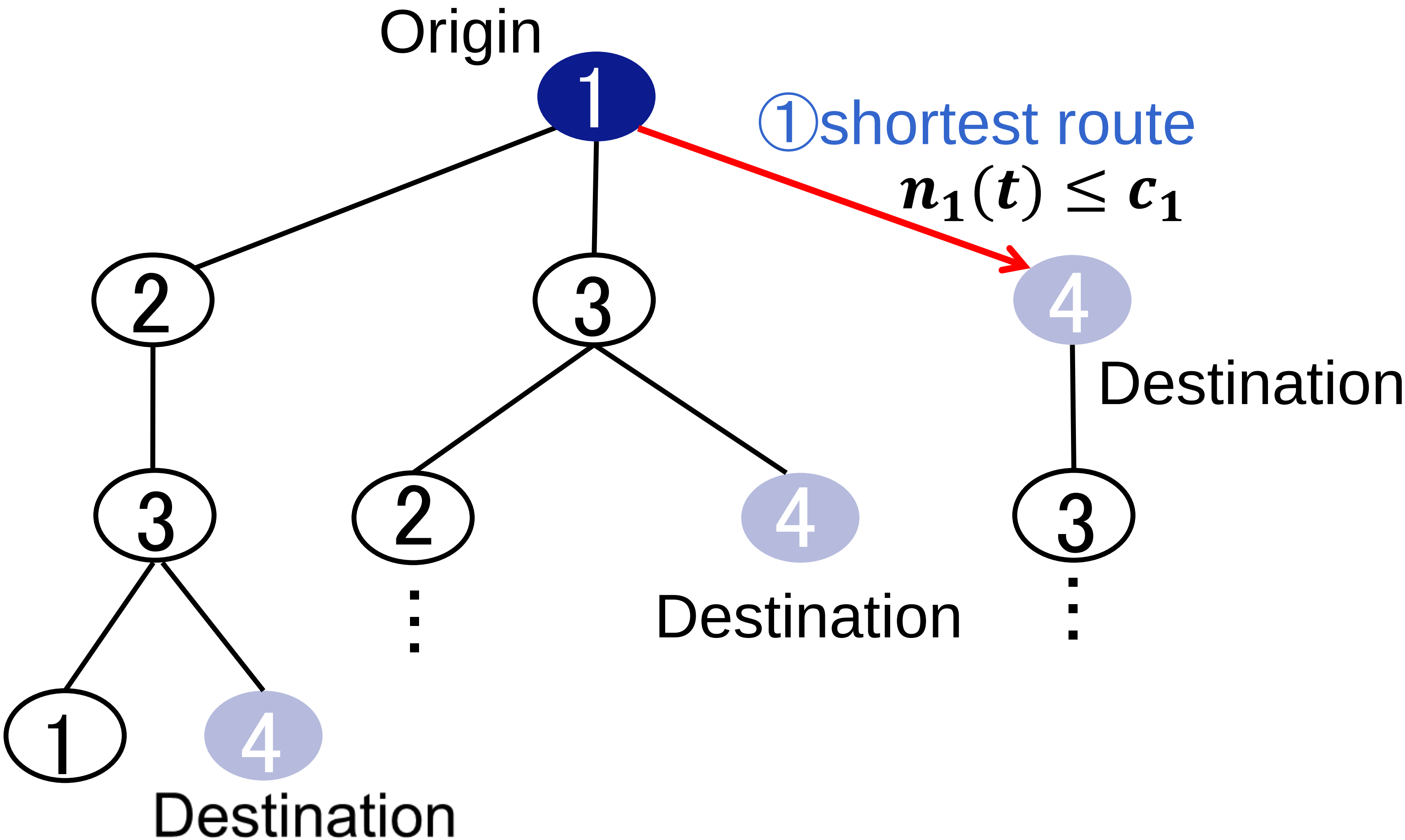
$$n_i(t) \leq c_i$$

(t : time step [min.])

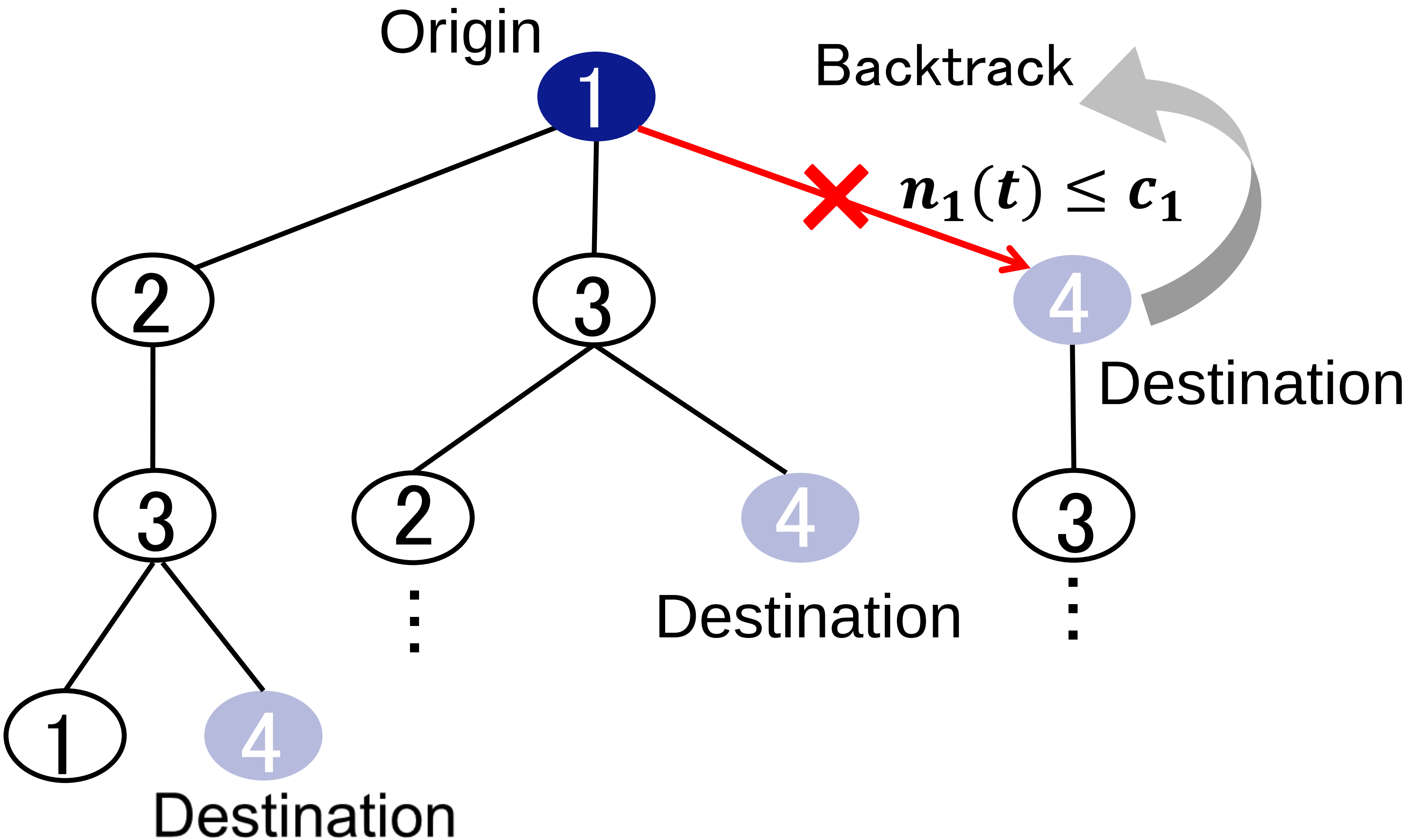
satisfy the constraints while waiting

  - use Backtracking Algorithm to search the route
  - adopt Dijkstra method to decide searching order by the shortest route

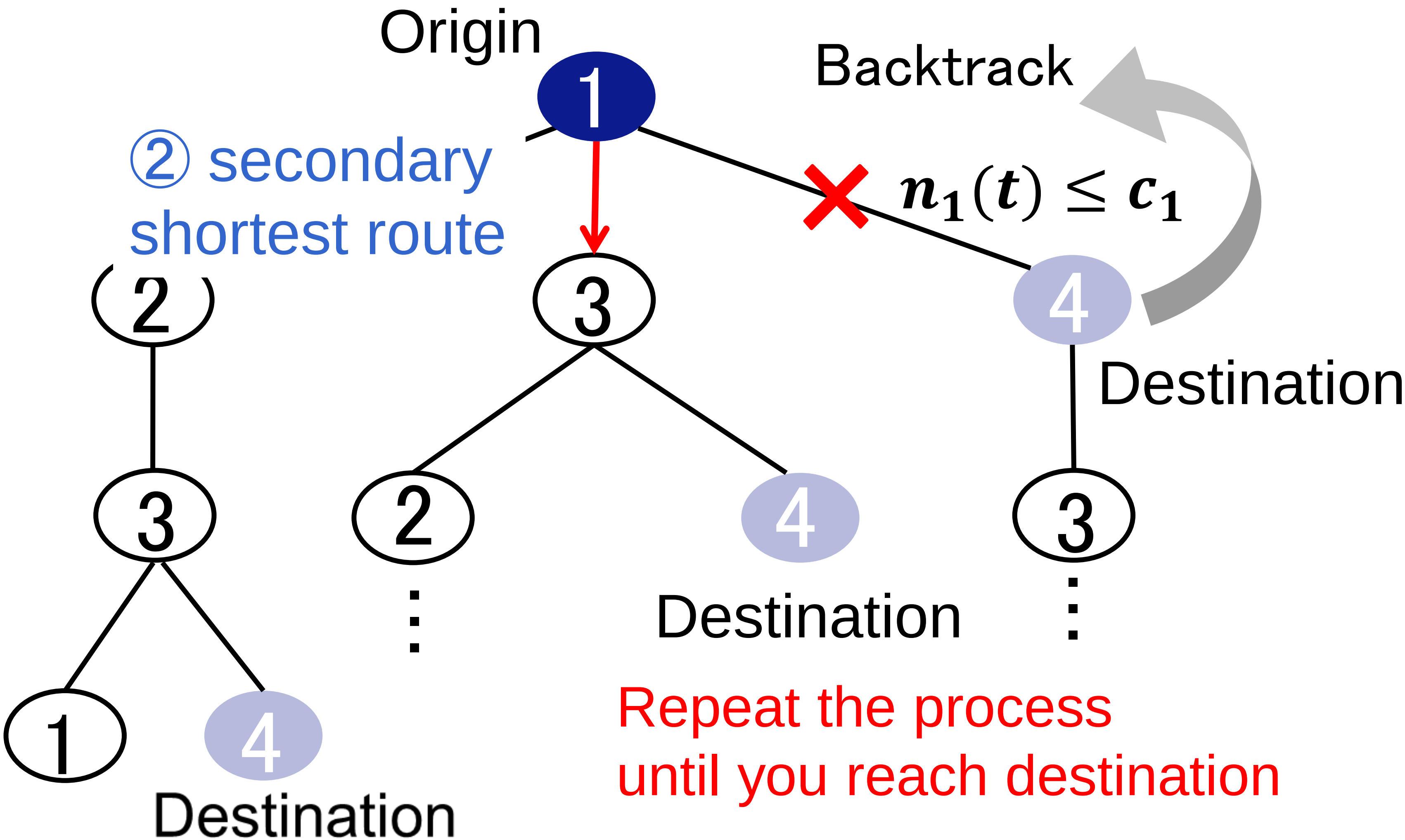
1. Passengers' Route Assignment



1. Passengers' Route Assignment



1. Passengers' Route Assignment



make possible to disperse passengers' route

Experiment : Passengers' Route Assignment

- Settings
  - equal to Preliminary Experiment
  - adopt capacity c_i obtained from Preliminary Experiment
  - stop all trains in the first 90[min.]

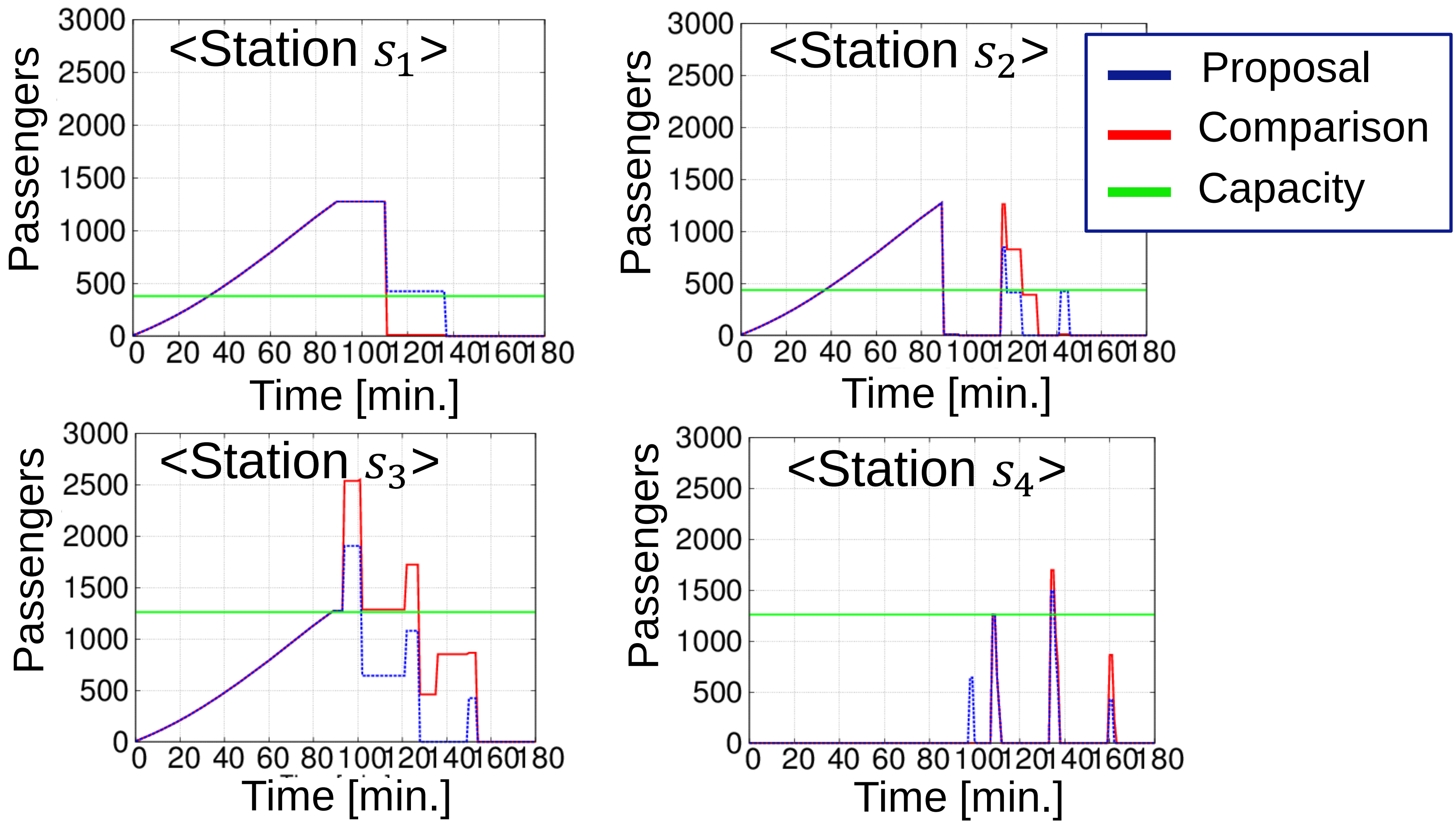
- Proposal :

change the route by backtracking & avoid capacity overflow

- Comparison (passenger's natural behavior) :

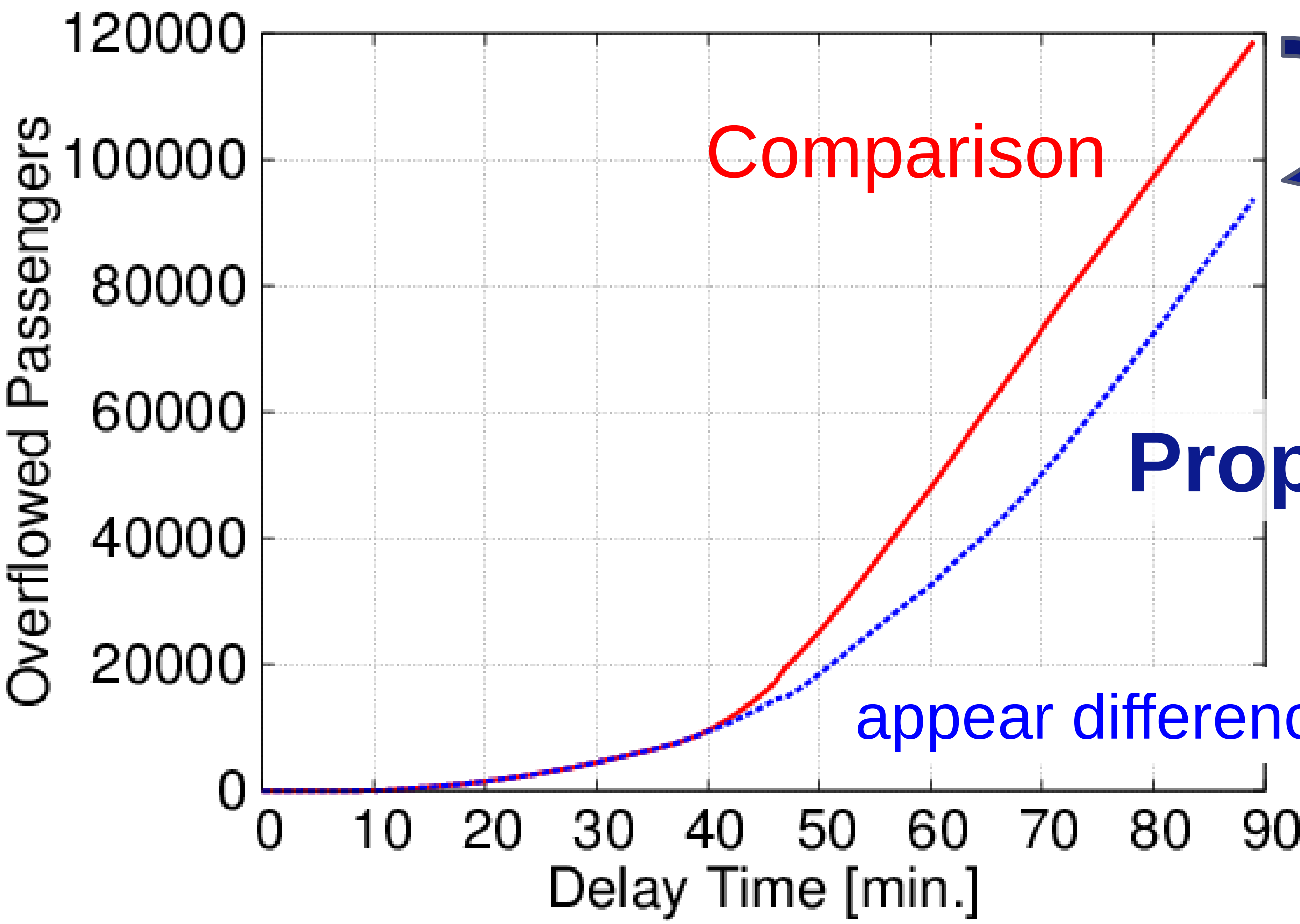
go on only the shortest route & avoid capacity overflow
wait until they are allowed to enter the station

Result : number of passengers in the station



- control overflow than passengers' natural behavior

Result : overflowed passengers upon delay time



lower overflowed passengers by **21.04%**

- decrease overflow than passengers' natural behavior in any delay time

2. Modify Train Schedule

- Objective :

change the **Train schedule** to reduce the overflowed passengers

- formulate as Combinatorial Optimization Problem
- solve by means of Genetic Algorithm

  – Objective function

$$min.\ \text{fitness} = \sum_{i=1}^{4}(n_i(t) - c_i)$$

unless $(n_i(t) - c_i) \leq 0$

➡ minimize the overflowed passengers

2. Modify Train Schedule

- define individual
  - one of the elements of Train Schedule

    length of stoppage time of each train at each Station

    → set as variable

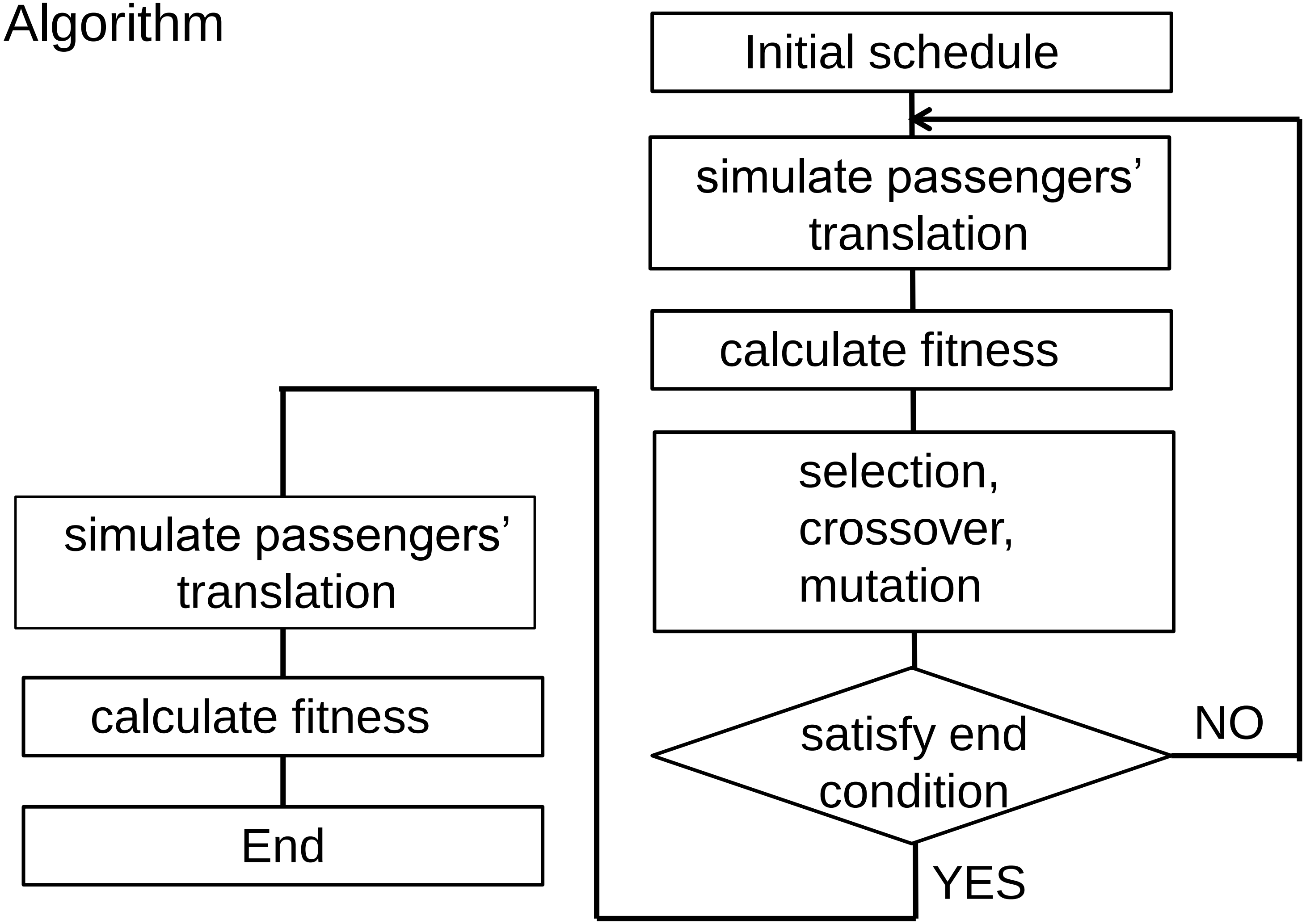
- Stoppage time

	Station s_1	Station s_2	Station s_3	Station s_4
Train v_1	x_1	x_2	x_3	x_4
Train v_2	x_5	x_6	x_7	x_8
Train v_3	x_9			
Train v_4			x_{10}	

- search the optimal combination of variables

2. Modify Train Schedule

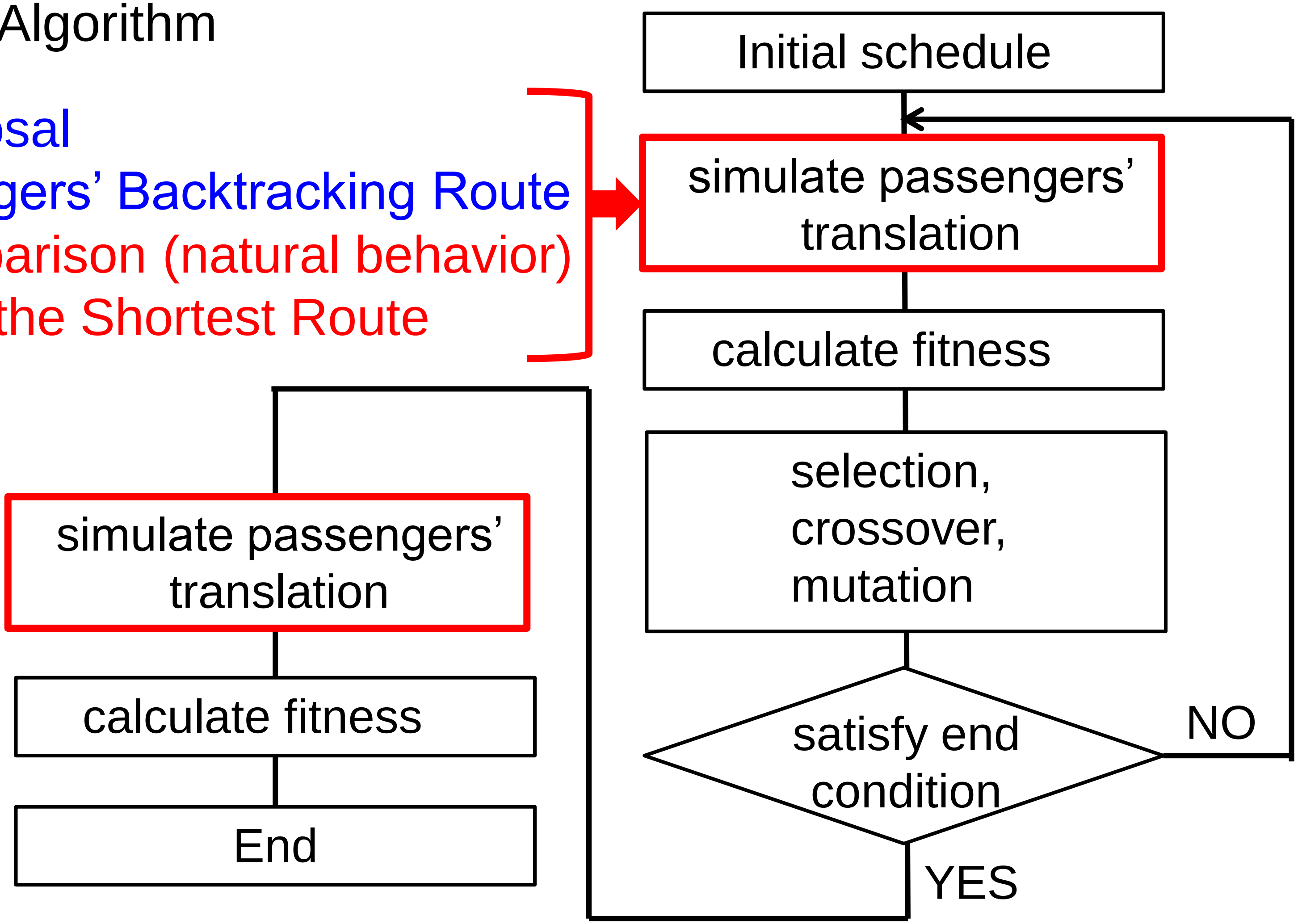
- Algorithm



2. Modify Train Schedule

- Algorithm

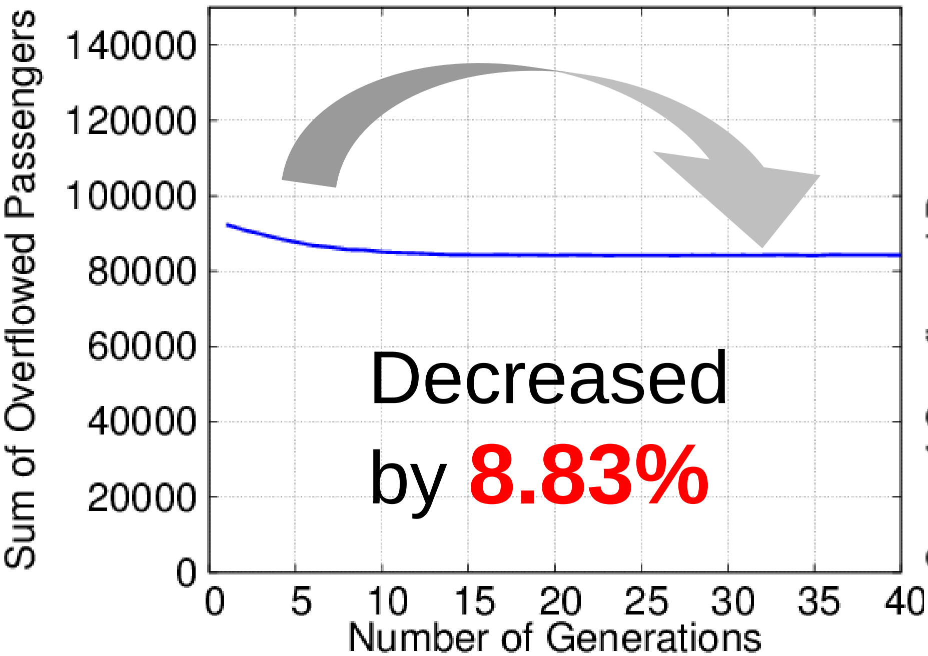
①Proposal
Passengers' Backtracking Route
②Comparison (natural behavior)
wait on the Shortest Route



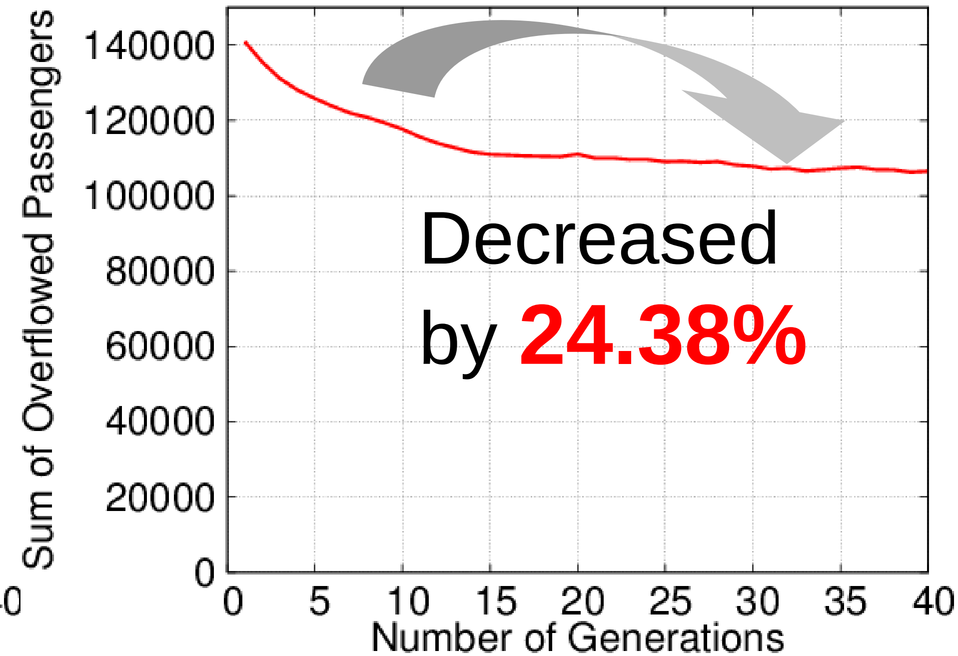
create Train Schedule with Individual
which minimize the fitness

Result : sum of overflowed passengers

Change Train Schedule
with Proposal Route



Change Train Schedule
with Comparison Route



the both overflowed passengers decreased

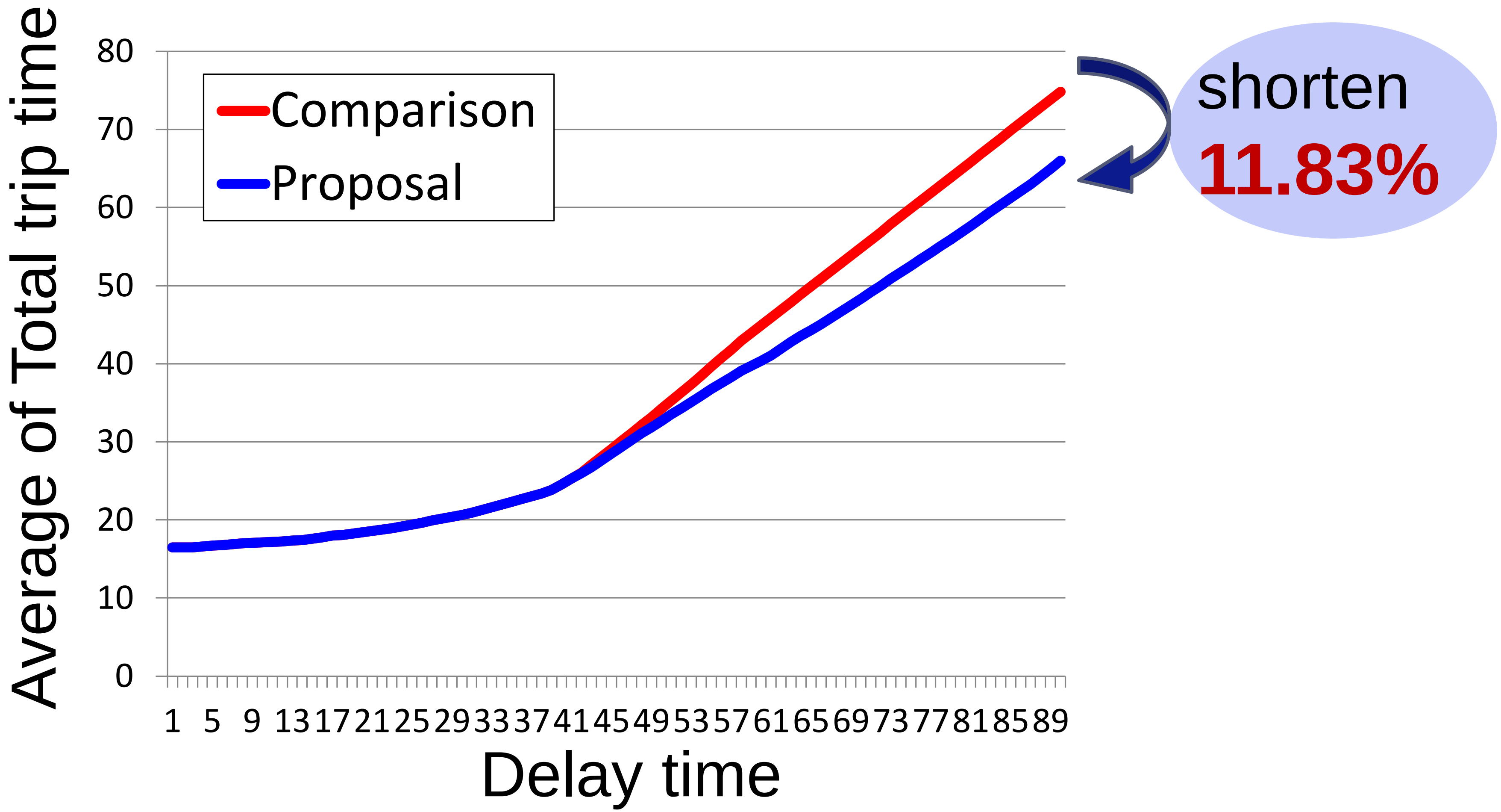
- have still lower by 20.94% than comparison after modifying the Train Schedule

Conclusion

- focused on the software measure to avoid capacity overflow
- proposed to change Passengers' Route not to overflow
  - solve as Constraint Satisfaction Problem
  - control and decrease the overflow by 21.04%
- proposed to change Train Schedule which reduce overflow
  - solve by means of Genetic Algorithm
  - decreased 8.83% of the overflow after modifying the Train Schedule
- decreased 29.04% overflowed passengers
  by changing both Passengers' route & Train schedule
- changing only software are also effective to prevent overflow

Thank you for your kind attention.

Result : total trip time upon delay time

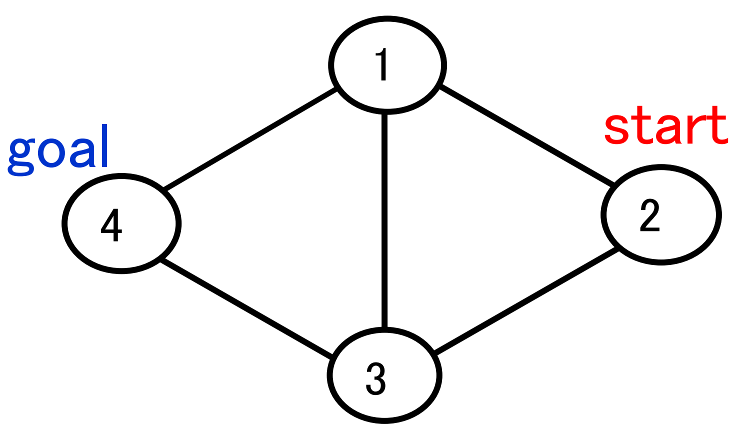


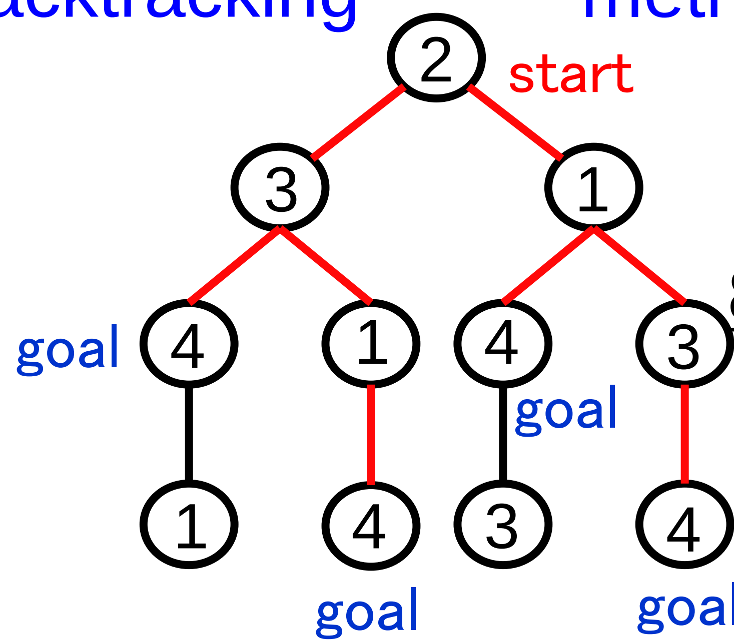
- Shorten total trip time than passengers' natural behavior in any delay time

time complexity

- Station number is n

Number of judging × $n-1$ Number of Backtracking × n^2 Dijkstra method





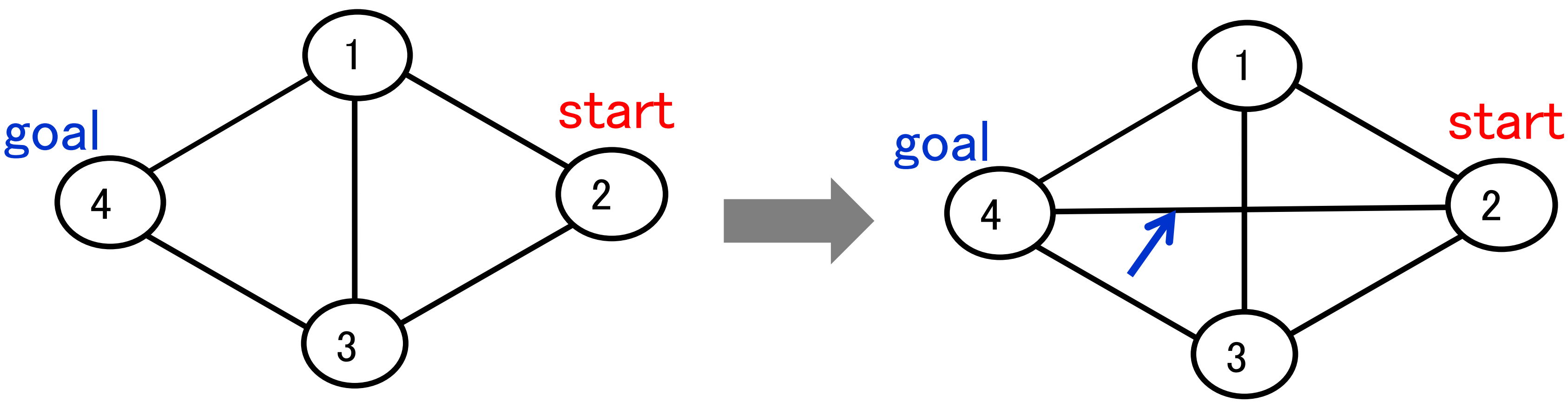
Judge 8 times

➡ Worst-case execution time $O(8(n-1)n^2) = O(n^3)$

time complexity

- Station number is n

Number of judging × $n-1$ Number of Backtracking × n^2 Dijkstra method



※if the train line(degree) increase, the route increase & directly affect the number of judging

time complexity

- Station number is n

Number of judging

Estimate worst judging time
With the complete graph

- Only 15 train lines
  in Shinjuku station

n	degree	**Number of judge**
1	0	**0**
2	1	**1**
3	2	**3**
4	3	**9**
5	4	**31**
6	5	**129**
7	6	**651**
8	7	**3913**
9	8	**27399**
10	9	**219201**
:	:	**:**
n	$n-1$	

Combinatorial Explosion