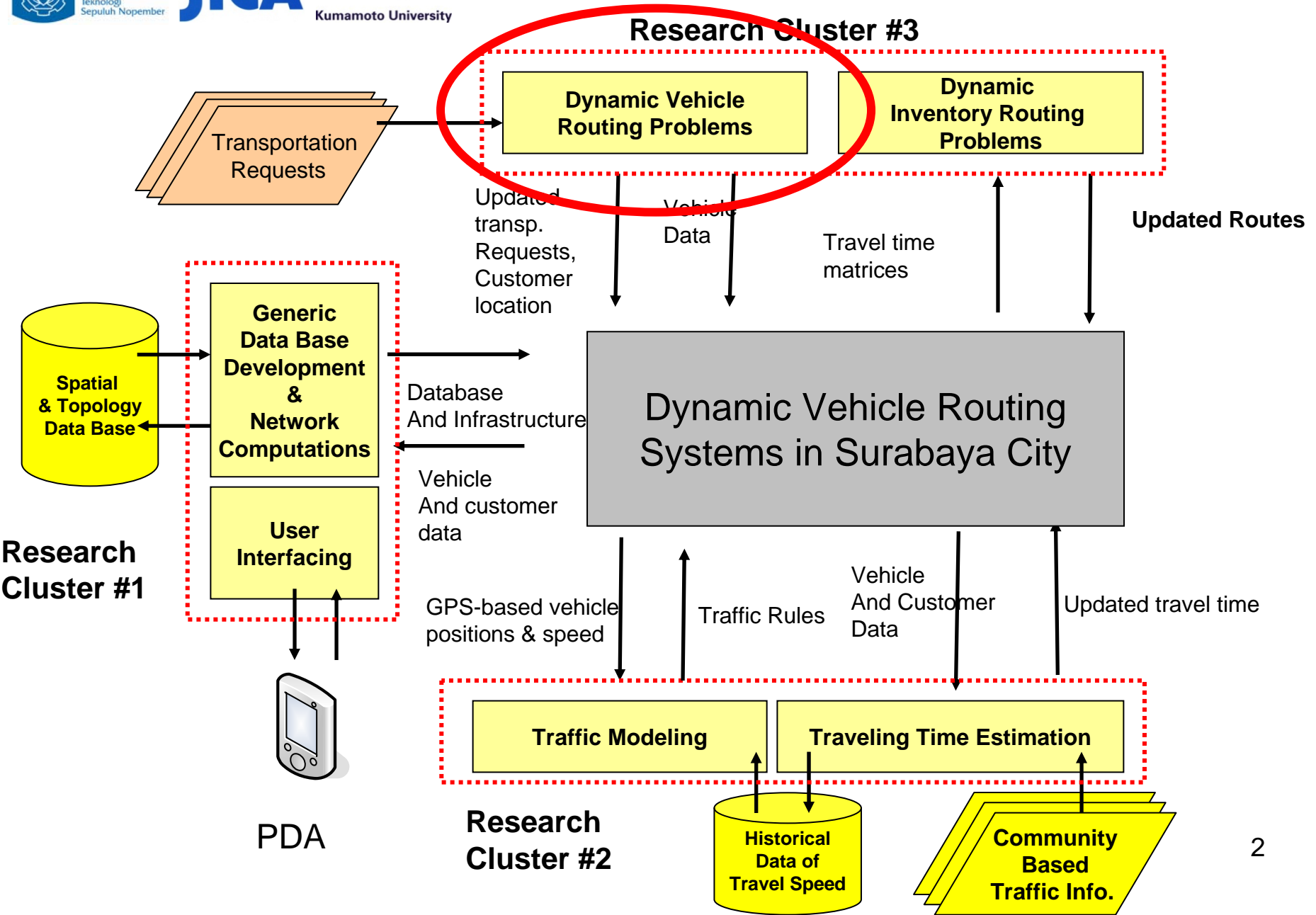


# Ant-colony Heuristics for Solving Dynamic Vehicle Routing Problem with Time Windows for Inter-city Courier Service Providers

**Nurlita Gamayanti, Ahmad Rusdiansyah, Abdullah Alkaff,  
Keiichi Uchimura**

**2nd T-LOG/EASTS Logistics IRG Seminar**  
**Tokoname, JAPAN**

# Collaboration Research



# Many-to-one (One-to-many) Problems

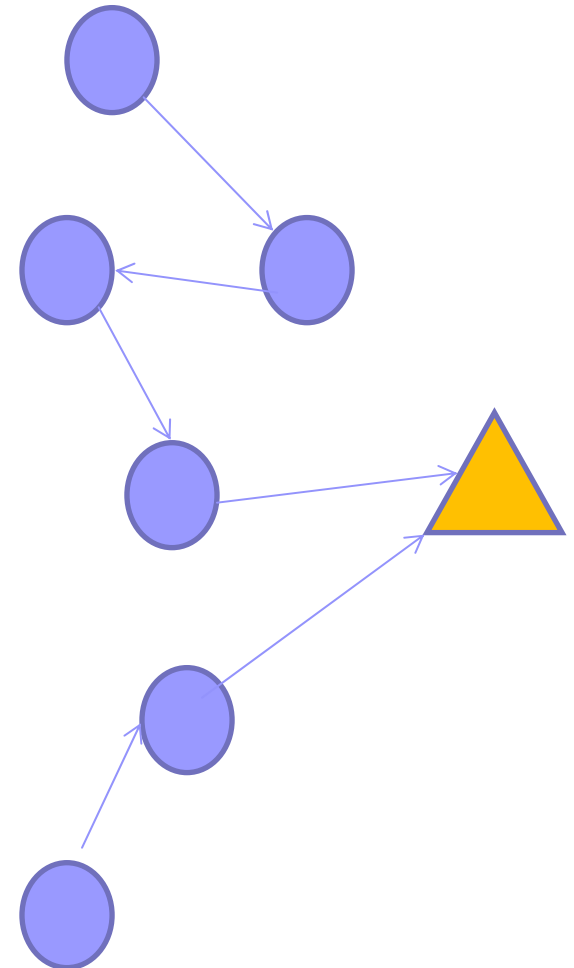
- **Inter-city Courier service problems is a many-to-one problems** where parcels and mail are collected at various locations and brought back at a central depot for further processing,

## Other Similar Systems:

**Feeder systems** where, for example, people are taken by mini-buses and transported to a train station.

## Dynamic traveling repairman problem (DTRP);

- - Utility firms (electricity, gas, water and sewer) that responds to customer requests for repair or maintenance of its facilities at the customer premise,
- Automotive Repairing Mobile Services



# The Cases of ICCS

- A fleet of vehicles with limited capacity starts from the depot, visits a number of customers within predefined time windows in order to pick up the package and finally finishes at one of the consolidation points.

-The consolidation points are located near from the highway gate  
-The consolidation point chosen is based on the schedules of the inter-city cargo vehicles that depart from that location.

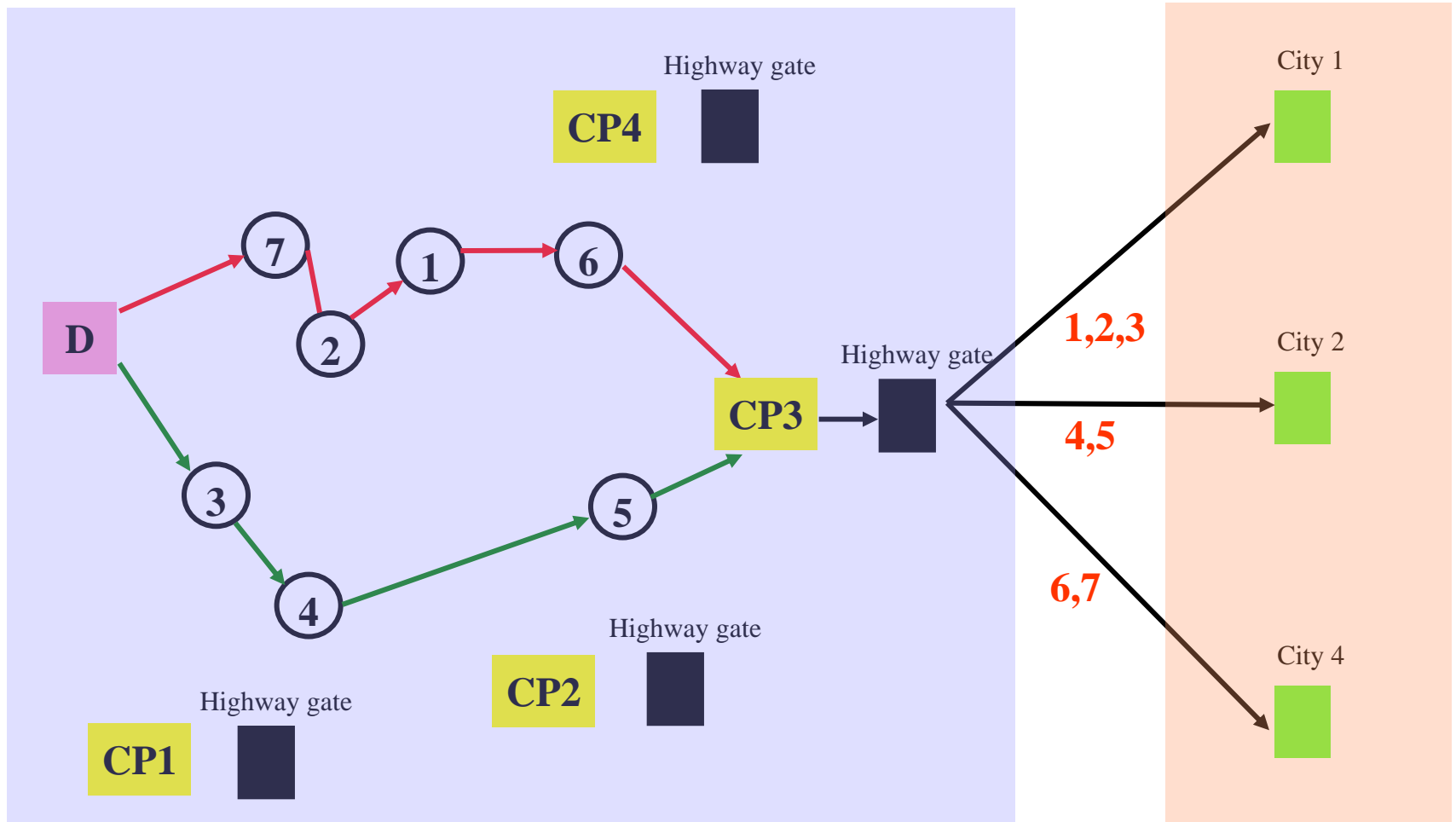
-There are 4 consolidation points

- Some of the orders are known in advance (static problem) and the others arrive after the vehicles leave the depot (dynamic problem).

- We dynamically build the vehicle routes due to real time information

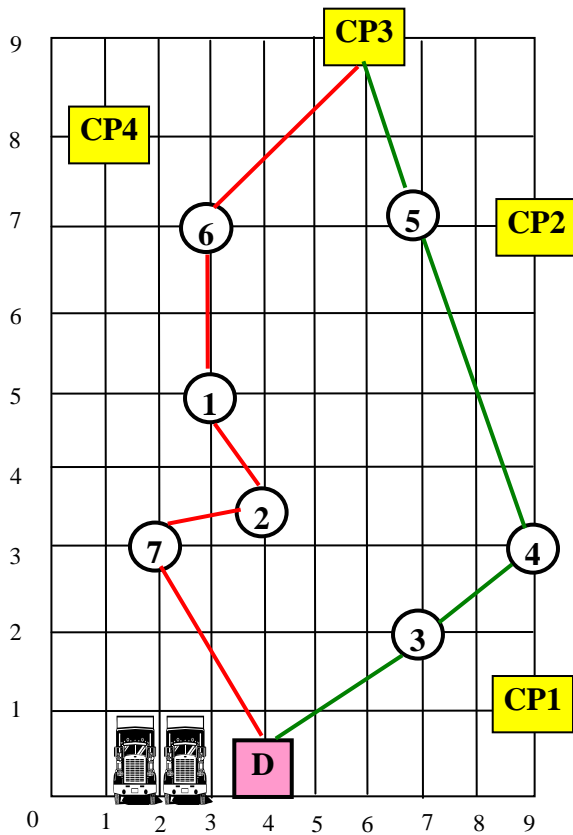


# Inter-City Courier Services

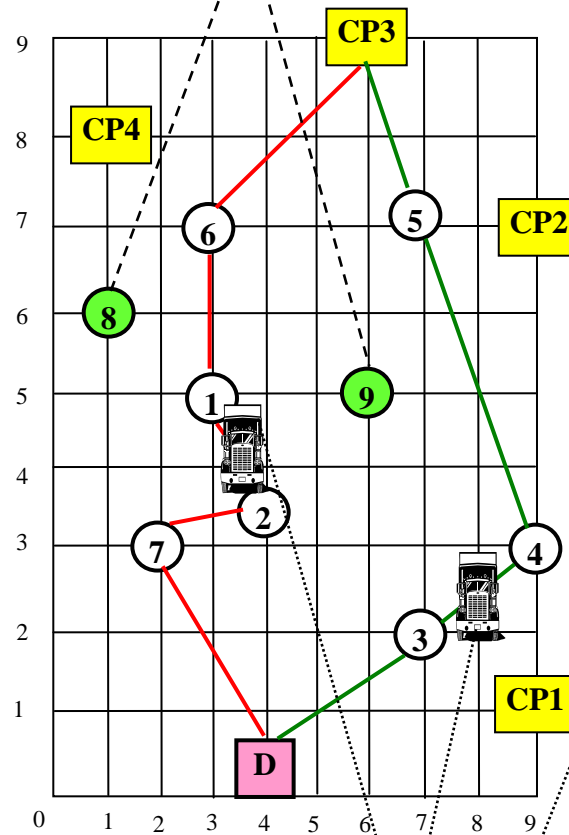


# Illustration

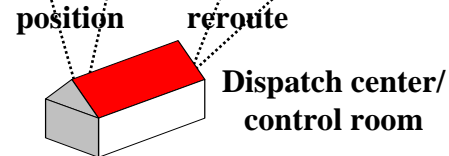
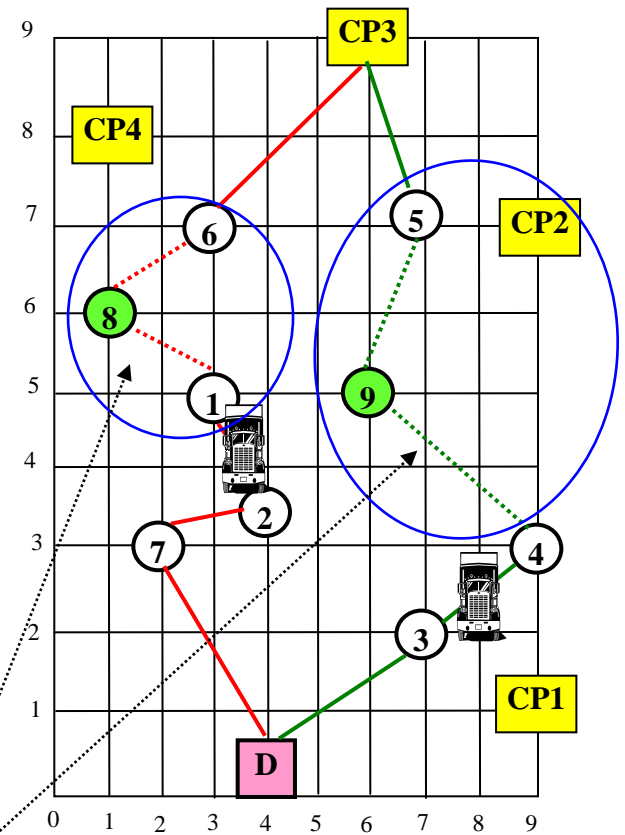
Initial Route



New Orders / Online Requests



Updating Routes



# Dynamic Routing Classifications

- Source : Larsen et al, 2007

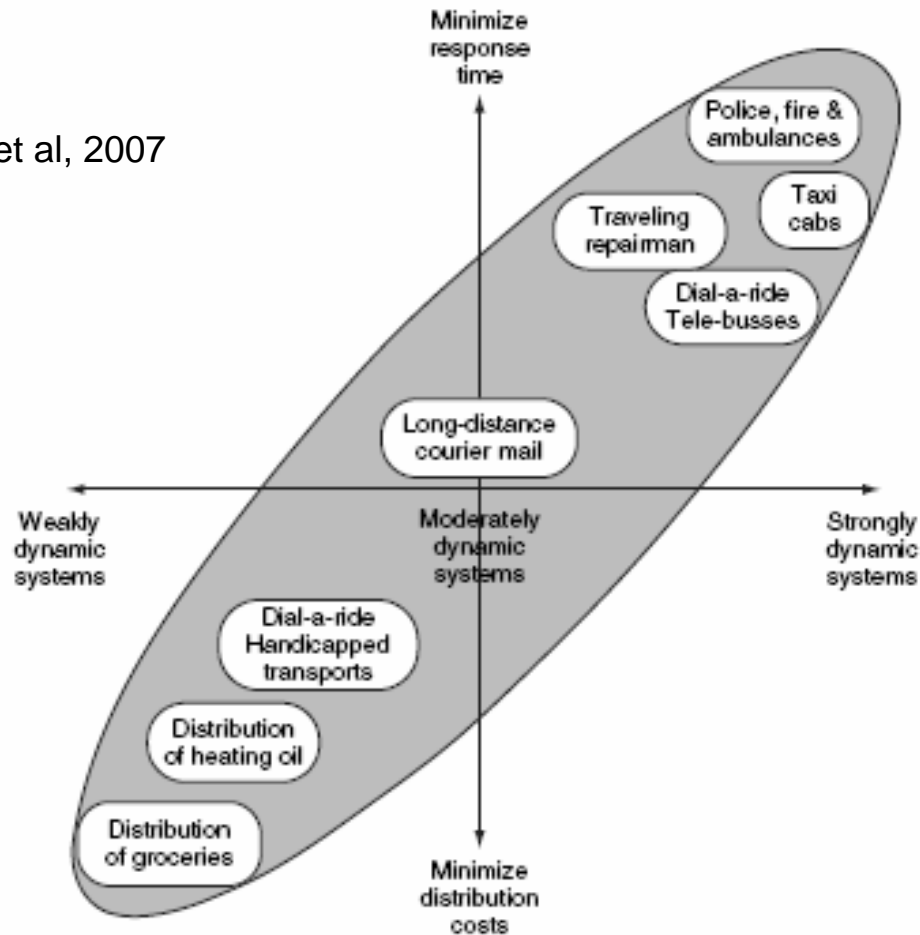


Figure 2-5. Framework for classifying dynamic routing problems by their degree of dynamism and their objective.

# Problem Formulation

How to develop a heuristics algorithm for solving Dynamic Vehicle Routing Problem with Time Windows for Inter-City Courier Services Provider

- Information of concerning travel time and other real time information got from GPS devices
- Parcels collected to be assumed transportable without differentiating its type
- Capacity of every vehicle is homogeneous.





# Problem Definition (1)

## Objective Function

Minimize the total travel time

$$\text{Min} \quad \sum_{k \in V} \sum_{i \in N} \sum_{j \in N} t_{ij} x_{ij}^k + w_j^k$$

$t_{ij}$  : travel time of edge (i,j), includes the service time at node i

$w_j^k$  : waiting time for vehicle k at node j

$x_{ij}^k$   $\left\{ \begin{array}{l} = 1 \quad : \text{if vehicle } k \text{ visits node } j \text{ immediately after node } i \\ = 0 \quad : \text{otherwise} \end{array} \right.$

# Problem Definition (2)

## Flow Constraints

- Each node is served exactly once by only one vehicle

$$\sum_{k \in V} \sum_{i \in N} x_{ij}^k = 1 \quad \forall j \in N \setminus \{0, c\}$$

$$\sum_{k \in V} \sum_{j \in N} x_{ij}^k = 1 \quad \forall i \in N \setminus \{0, c\}$$

$N$  : the set of nodes, 0 denotes the depot, c denotes the consolidation point,  $V$  : the set of vehicle

- Each vehicle must leave the node that it just entered

$$\sum_{i \in N} x_{ih}^k - \sum_{j \in N} x_{hj}^k = 0 \quad \forall h \in N \setminus \{0, c\}, \forall k \in V$$

# Problem Definition (3)

- Every vehicle must start from the depot

$$\sum_{j \in N \setminus \{0\}} x_{0j}^k = 1 \quad \forall k \in V$$

- Every vehicle must finish at the consolidation point

$$\sum_{i \in N \setminus \{0, c\}} x_{ic}^k = 1 \quad \forall k \in V$$

## Capacity Constraints

- The total demand of any vehicle route can not exceed the vehicle capacity

$$\sum_{i \in N} d_i \sum_{j \in N} x_{ij}^k \leq Q \quad \forall k \in V$$



# Problem Definition (4)

## Time Window Constraint

- The vehicle  $k$  can not arrive at  $j$  before  $b_i^k + t_{ij}$  if it travels from  $i$  to  $j$

$$x_{ij}^k (b_i^k + t_{ij}) \leq b_j^k \quad \forall i \in N, \forall j \in N, \forall k \in V$$

- The start of the service must be within the time window at each node

$$e_i \leq b_i^k \leq l_i \quad \forall i \in N, \forall k \in V$$

$e_i$  : the beginning of the time window for node  $i$

$l_i$  : the end of the time window for node  $i$

# Complete Formulation (1)

$$\text{Min } \left. \sum_{k \in V} \sum_{i \in N} \sum_{j \in N} t_{ij} x_{ij}^k + w_j^k + v^k \right\} \text{ Minimize the total travel time}$$

*s.t.*

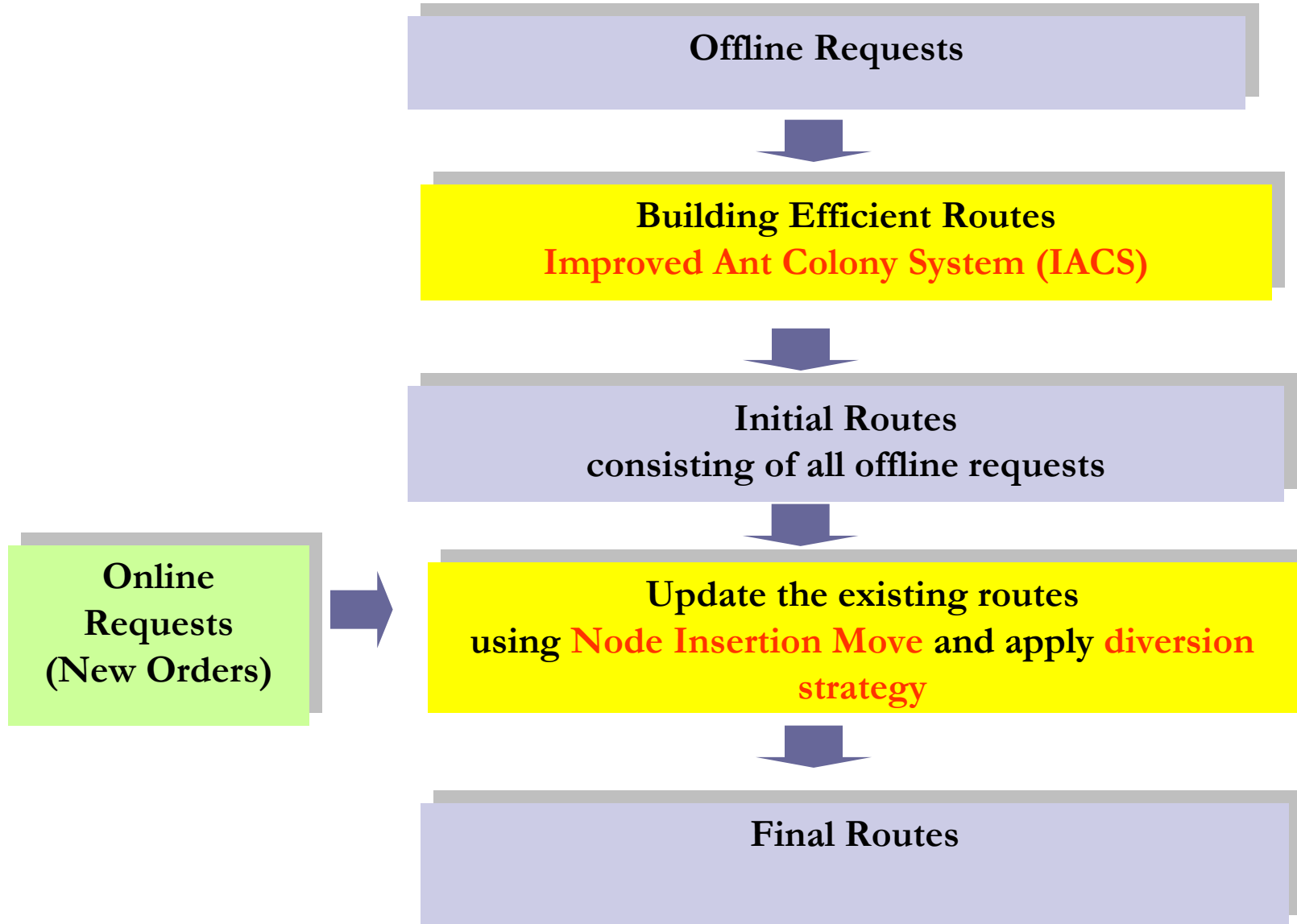
$$\left. \begin{aligned} \sum_{k \in V} \sum_{i \in N} x_{ij}^k &= 1 & \forall j \in N \setminus \{0\} \\ \sum_{k \in V} \sum_{j \in N} x_{ij}^k &= 1 & \forall i \in N \setminus \{0\} \\ \sum_{i \in N} x_{ih}^k - \sum_{j \in N} x_{hj}^k &= 0 & \forall h \in N \setminus \{0\}, \forall k \in V \\ \sum_{i \in N \setminus \{0, c\}} x_{ic}^k &= 1 & \forall k \in V \\ \sum_{j \in N \setminus \{0, c\}} x_{0j}^k &= 1 & \forall k \in V \end{aligned} \right\} \text{ Flow Constraint}$$

$$\left. \sum_{i \in N} d_i \sum_{j \in N} x_{ij}^k \leq Q \quad \forall k \in V \right\} \text{ Capacity Constraint}$$

$$\left. \begin{aligned} x_{ij}^k (b_i^k + t_{ij}) &\leq b_j^k & \forall i \in N, \forall j \in N, \forall k \in V \\ e_i &\leq b_i^k \leq l_i & \forall i \in N, \forall k \in V \\ x_{ij}^k &\in \{0, 1\} & \forall i \in N, \forall j \in N, \forall k \in V \end{aligned} \right\} \text{ Time Window Constraint}$$

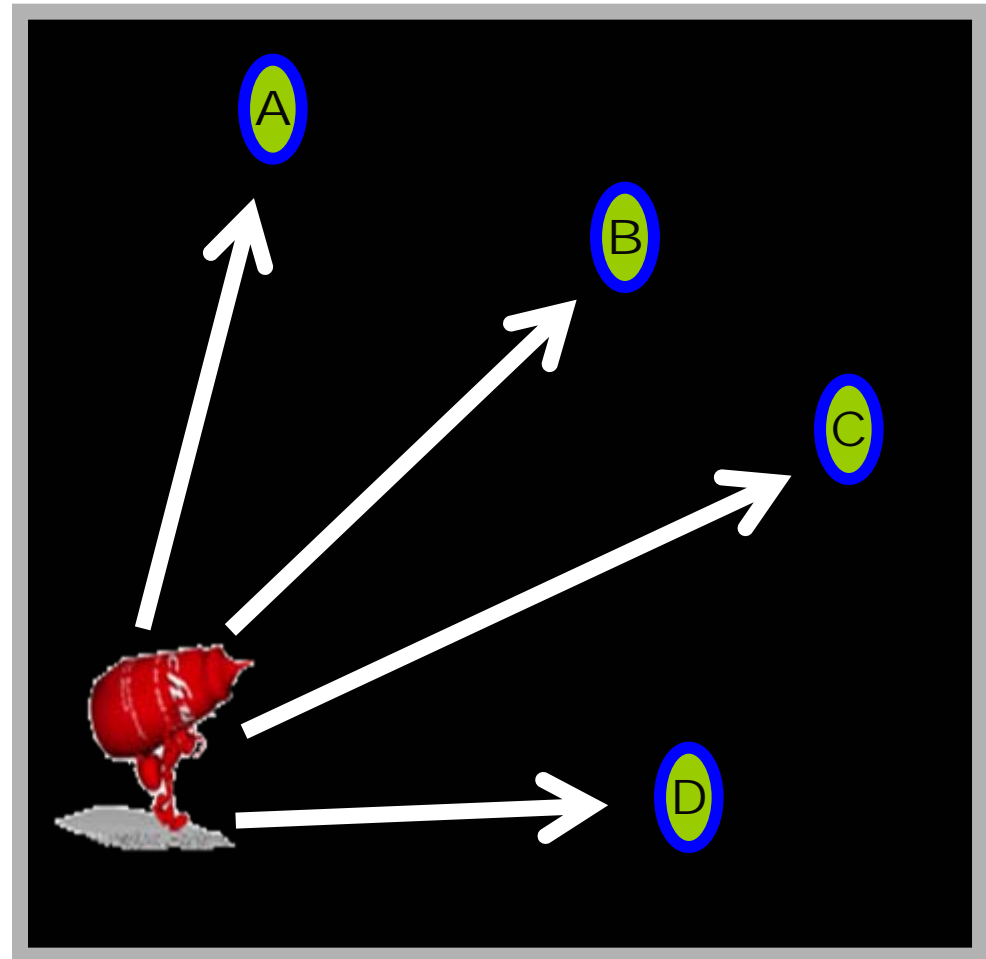


# Framework



# Ant Colony System

- Based on the ability of real ants for finding the shortest path from a food source to their nest
- While walking, ants deposit pheromone on the ground, in probability, pheromone previously deposited by other ants. The more number of pheromone denotes the shorter trajectory



# Procedure of Ant Colony System

Set parameters  
Initialize the pheromone trails

Each ant builds the solution by the State Transition Rule  
Carries out Local Pheromone Update

Apply the Local Search to improve the ants solution

Carry out global pheromone update





# ACS and IACS

<b>Standard ACS</b>	<b>Improved ACS</b>
<b>Route Construction Rule :</b> <ul style="list-style-type: none"><li>- There are b solutions in 1 iteration (b=number of ant)</li><li>- In 1 iteration, local search applied on each solution</li><li>- Parameter <math>\eta_{ij}</math> is the function of travel time of each arch only</li></ul>	<b>Route Construction Rule :</b> <ul style="list-style-type: none"><li>- There are b-1 solutions in 1 iteration</li><li>- In 1 iteration, local search applied on the best solution only</li><li>- Parameter <math>\eta_{ij}</math> is the function of travel time of each arch and waiting time on each node</li></ul>
<b>Global Pheromone Update Rule</b> <ul style="list-style-type: none"><li>- Global pheromone update is applied on arcs that form the best solution in current iteration</li></ul>	<b>Global Pheromone Update Rule</b> <ul style="list-style-type: none"><li>-Global pheromone update is applied on arcs that form the best solution in current iteration and a before-hand iteration</li></ul>

# IACS (1)

## Initialize Pheromone

The initial pheromone level of each edge :

$$\tau_0 = (N \times L_{NN})^{-1}$$

$N$  : number of nodes

$L_{NN}$  : the tour length produced by Nearest Neighbor heuristics



# IACS (2)

## Route Construction

Each ant moves from present node  $i$  to the next node  $j$  according to the state transition rule :

➤ if  $q \leq q_0$  (Exploitation)

$$j := \underset{j \in S'(i)}{\operatorname{arg\,max}} \{ (\tau_{ij})^\alpha (\eta_{ij})^\beta \}$$



➤ if  $q > q_0$  (Exploration)

- Compute ,  $P_{ij} = \frac{(\tau_{ij})^\alpha (\eta_{ij})^\beta}{\sum_{j \in S'(i)} (\tau_{cj})^\alpha (\eta_{cj})^\beta}$   $j \in S'(i)$

- Compute  $F_{ij}$

- if  $(q > F_{ia})$  and  $(q < F_{ib})$  then :  $\mathbf{j} = \mathbf{b}$   $\{a, b\} \in S'(i)$



# IACS (3)



## Local Pheromone Update

Each ant which moves from present node  $i$  to the next node  $j$  will update the pheromone level on edge  $(i,j)$

$$\tau_{ij}^{new} = \tau_{ij}^{old} + (1 - \rho)\tau_0$$

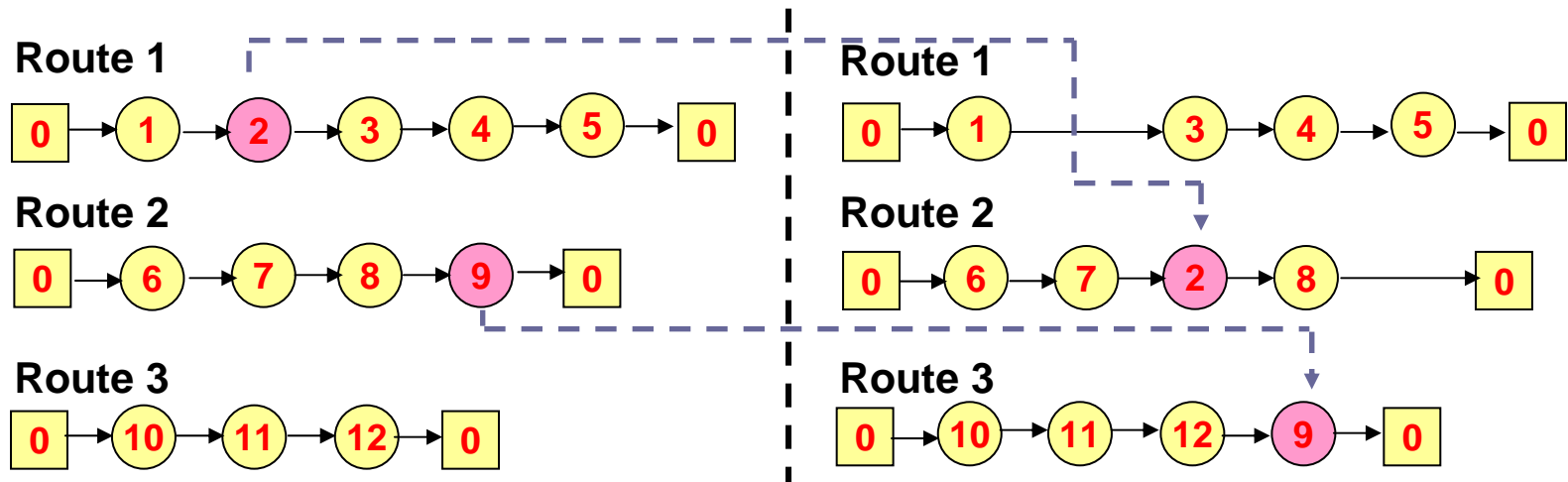
$$\tau_0 = (L_{NN} * n)^{-1}$$



# IACS(4)

## Local Search

- **Node Insertion Move** : One node is inserted to another route



# IACS (5)

## Global Pheromone Update

$$\tau_{ij}^{new} = (1 - \gamma)\tau_{ij}^{old} + \gamma \Delta\tau_{ij} \quad (i, j) \in \text{tour pada } 1^{st} \text{ Solution dan } 2^{nd} \text{ Solution}$$

$$\Delta\tau_{ij} = \frac{(Cost\ 3^{rd}\ BestIteration - Cost\ of\ The\ Best\ Solution) + (Cost\ 3^{rd}\ BestIteration - Cost\ BestIteration)}{Cost\ 3^{rd}\ BestIteration}$$



# Procedure IACS



Step 1 : Set Parameters and initialize the pheromone

Step 2 : Generate an initial solution using Nearest Neighbor heuristics

Step 3 : Apply the local search (Node Insertion Move) to the initial solution and let it to be the 1<sup>st</sup> solution of population.  $g = 1, m = 2$

Step 4 : Construct solution based on the state transition rule and progress local pheromone update.  $m = m + 1$

Step 5 : if  $m >$  number of ants,  $m = 2$  and go to step 6. Otherwise go to step 4

Step 6 : Sort the solution 2~ number of ants. Apply local search (Node Insertion Move) to the solution with optimal cost and let it to be the 2<sup>nd</sup> solution

Step 7 : Apply the global pheromone update to the 1<sup>st</sup> solution and the 2<sup>nd</sup> solution

Step 8 : Record the best solution so far and let it to be the 1<sup>st</sup> solution in the next generation.  $g = g + 1$

Step 9 : If the stopping criteria is met, stop, then the output is the best solution. Otherwise, go to step 4



# Numerical Experiment (1)

## Design Experiment

- In this experiment, we use street network data set of Surabaya city with 30 requests.
- Maximum iteration = 20. Number of ant = 3. Exploration rate = 0,5. Local Pheromone decay = 0,1. Global pheromone decay = 0,1.
- The objective of this experiment is to show the algorithm has been developed has ability to :
  1. minimize the total travel time
  2. assign vehicle has been dispatched to service new requests



# Numerical Experiment (2)

## Examples

### **Tour 1 :**

DEPOT - 075 - 054 - 180 - 260 - 333 - 125 - 221 - 157 - 110 - 140 - 155 - 166  
081 - 116 - 336 – CP1 – Highway

### **Tour 2 :**

DEPOT - 052 - 223 - 108 - 060 - 003 - 024 – CP2 - Highway

### **Tour 3 :**

DEPOT - 095 - 267 - 103 - 084 - 064 – 063 - 235 - 283 - 291 – CP3- Highway

Total Travel Time : 24838 s

# Numerical Experiment (3)

## New Requests :

**008** : D = 100 ; RT = 09.00 ; BTW = 09.00 ; ETW = 13.00

**015** : D = 50 ; RT = 09.15 ; BTW = 09.15 ; ETW = 11.10

## New Route :

### Tour 1 :

DEPO T- 075 - 054 - 180 - 260 - **008** - 333 - **015** - 125 - 221 - 157 - 110 - 140  
- 155 - 166 081 - 116 - 336 - CP1 - Highway

### Tour 2 :

DEPOT - 052 - 223 - 108 - 060 - 003 - 024 - CP2 - Highway

### Tour 3 :

DEPO T- 095 - 267 - 103 - 084 - 064 - 063 - 235 - 283 - 291 - GP3-  
Highway

Total Travel Time : 27178 s



# Compared with Nearest Neighbor + Node Insertion

Method	Number of Request	Max Iteration	Total Travel Time	Total Travel Time With New Requests
IACS	30	20	24838 s	27178 s
Nearest Neighbor	30	20	64557 s	66048 s
Nearest Neighbor + Node Insertion	30	20	34705 s	34859 s



# Numerical Experiment 2 (1)

## Design Experiment

- In this experiment, we use street network data set of Surabaya city with 30 requests.
- Maximum iteration = 20. Number of ant = 5. Exploration rate = 0.1, 0.3, 0.5, 0.7, 0.9. Local Pheromone decay = 0,1. Global pheromone decay = 0,1.
- The objective of this experiment is to show the influence of exploration rate  $q_0$

# Numerical Experiment 2 (2)

## Experiment Result

Number of Order	Total Travel Time									
	$q_0 = 0,1$		$q_0 = 0,3$		$q_0 = 0,5$		$q_0 = 0,7$		$q_0 = 0,9$	
	Initial	Update	Initial	Update	Initial	Update	Initial	Update	Initial	Update
30	23508	26760	23795	29624	23834	29943	24313	28811	24800	27388

# Conclusion

- We have developed Improved Ant Colony System algorithm that can solve The Dynamic Vehicle Routing Problem with Time Windows for Inter-City Courier Services Provider
- Some numerical experiments have shown that the proposed algorithm could reduce the total costs provided by simple algorithms. However, more numerical experiments should be conducted to verify the performance of the proposed heuristics. Especially to compare to the lower bound of the problem and using the standard data sets, if any.

