Heuristics for CARP

Dr. Changwu HUANG (黄长武)
CSE, SUSTech

Definition & Notations

The CARP is usually defined on an **Undirected Graph** G=(V, E)

- V: the set of vertices (nodes) , n nodes includes one depot (node 1) with identical vehicles of **capacity** Q.
- E: the set of edges (arcs) , m edges contains a subset E_R of t **required edges (tasks)** to be serviced by a vehicle.
- Each $e \in E$ has a **traversal cost** $c(e)$ and a nonnegative **demand** $q(e)$.

Goal: find a set of routes for vehicles to serve all required tasks with **minimal cost**.

Route: a closed trip containing the depot and a set of traversed edges. Some of these edges are serviced by the route, while the others are traversed but not serviced.

A route can be expressed by $R = (\tau_1, \tau_2, \cdots, \tau_k)$, where $\tau_i \in E_R$ with chosen direction.

The complete route can be obtained by connecting shortest path between: depot to τ_1, τ_1 to τ_2, ..., τ_k to depot , using Dijkstra's algorithm.

Heuristics for CARP

Simple Constructive Methods	Two-phase Constructive Methods	Adaptations of Meta-heuristics
Routes are built one by one to construct a solution of CARP. **Path-Scanning** **Augment-Merge**	Route first-cluster second; Cluster first-route second. **Ulusoy’s Split Procedure**	Simulated Annealing, Tabu Search, Variable Neighborhood Search, Evolutionary Algorithms, Memetic Algorithms, Ant Colony Optimization.

Path-scanning

Start at the depot, the emerging rout ending at node i is progressively extended by adding at the end one required edge or task $\tau = (j, k)$ which is the **closest to the end of current path** (node i), not yet serviced and compatible with vehicle capacity. If multiple tasks are the closest to the end of current path, five rules are used to determine the next task:

1. Minimize the ratio $c(\tau)\ /q(\tau)$.
2. Maximize the ratio $c(\tau)\ /q(\tau)$.
3. Maximize the return cost from k to the depot.
4. Minimize the return cost from k to the depot.
5. If the vehicle is less than half-full, then apply rule 3, otherwise apply rule 4.

The construction of current route stops when the capacity of vehicle is violated or all tasks incident to the end of path are already serviced. In this case, the vehicle returns to the depot via a shortest path.

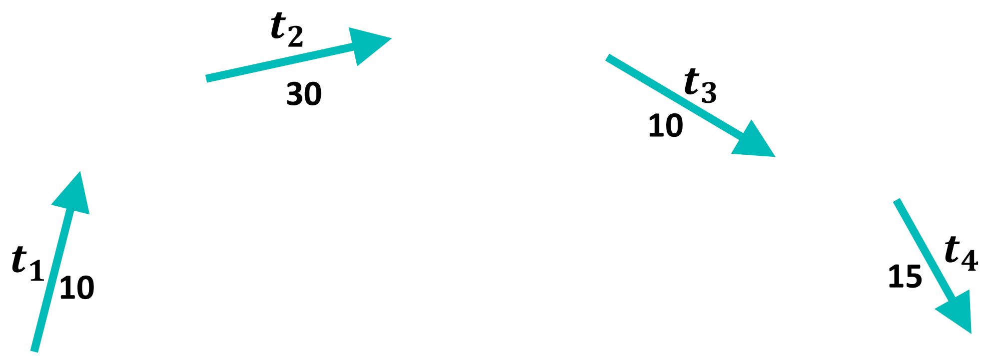
The heuristic is executed successively with the five rules, and the best of the resulting solution is returned.

Ulusoy Split

The giant tour T , which is a list of required edges with chosen directions and connected implicitly by shortest paths, is firstly created by other heuristics, such as Path Scanning. Ulusoy Split partition the tour into feasible vehicle routes.

Giant tour: $[\boldsymbol{t_1}, \boldsymbol{t_2}, \boldsymbol{t_3}, \boldsymbol{t_4}]$

Capacity = 50

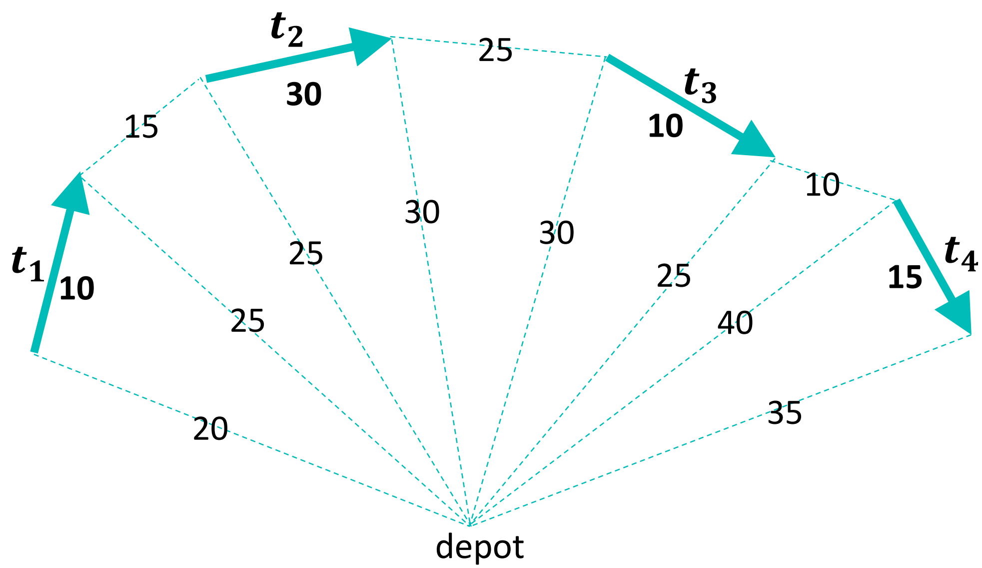


Ulusoy Split

The giant tour T , which is a list of required edges with chosen directions and connected implicitly by shortest paths, is firstly created by other heuristics, such as Path Scanning. Ulusoy Split partition the tour into feasible vehicle routes.

Giant tour: $[\boldsymbol{t_1}, \boldsymbol{t_2}, \boldsymbol{t_3}, \boldsymbol{t_4}]$

Capacity = 50

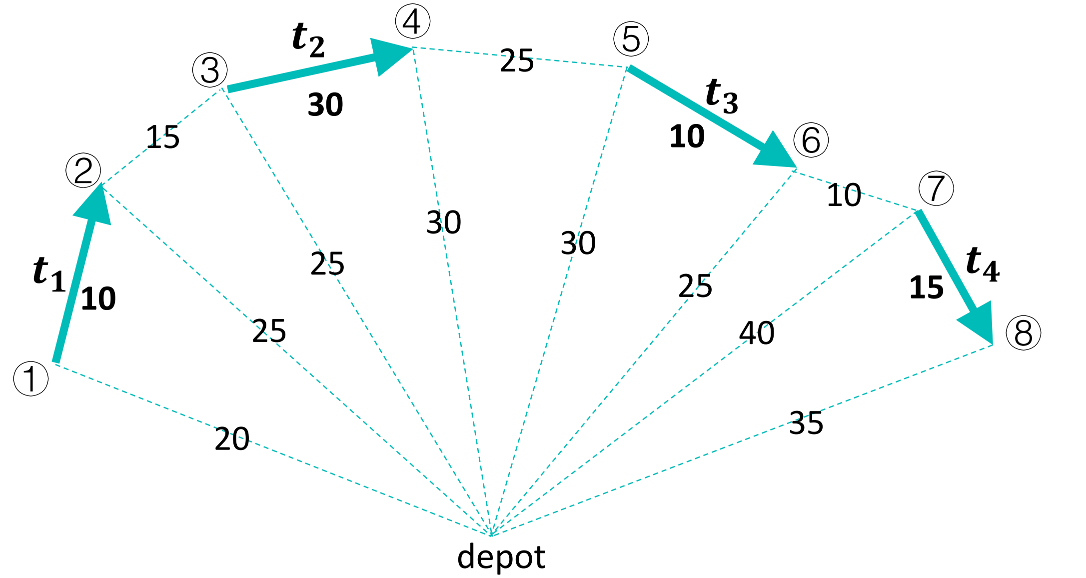


Ulusoy Split

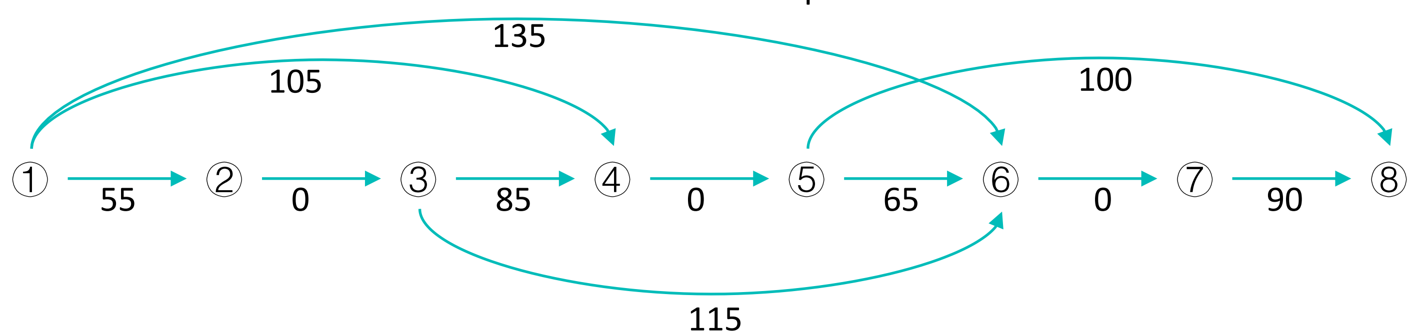
The giant tour T , which is a list of required edges with chosen directions and connected implicitly by shortest paths, is firstly created by other heuristics, such as Path Scanning. Ulusoy Split partition the tour into feasible vehicle routes.

Giant tour: $[\boldsymbol{t_1}, \boldsymbol{t_2}, \boldsymbol{t_3}, \boldsymbol{t_4}]$

Capacity = 50



New Directed Graph

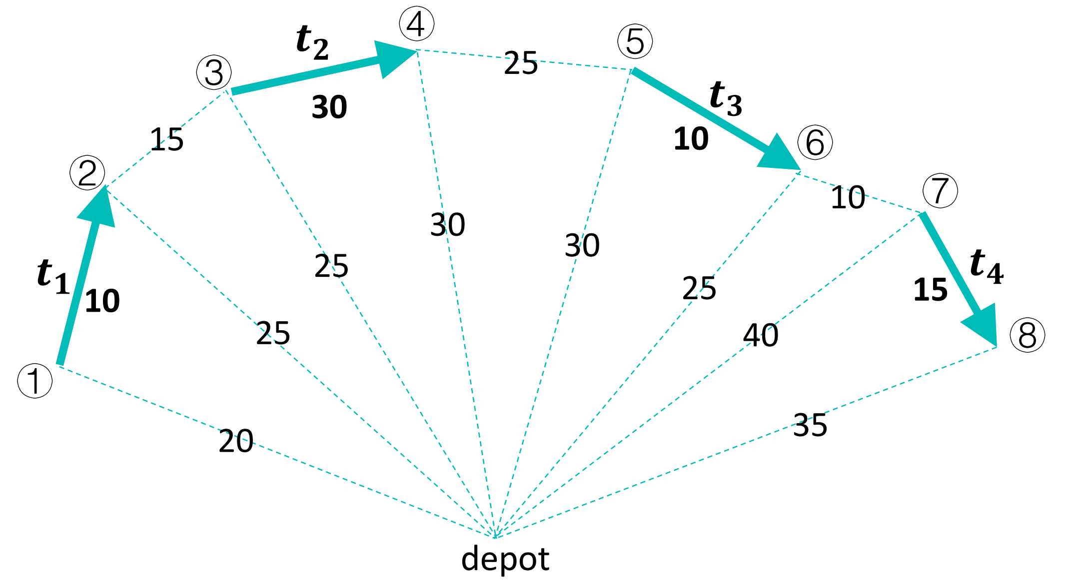


Ulusoy Split

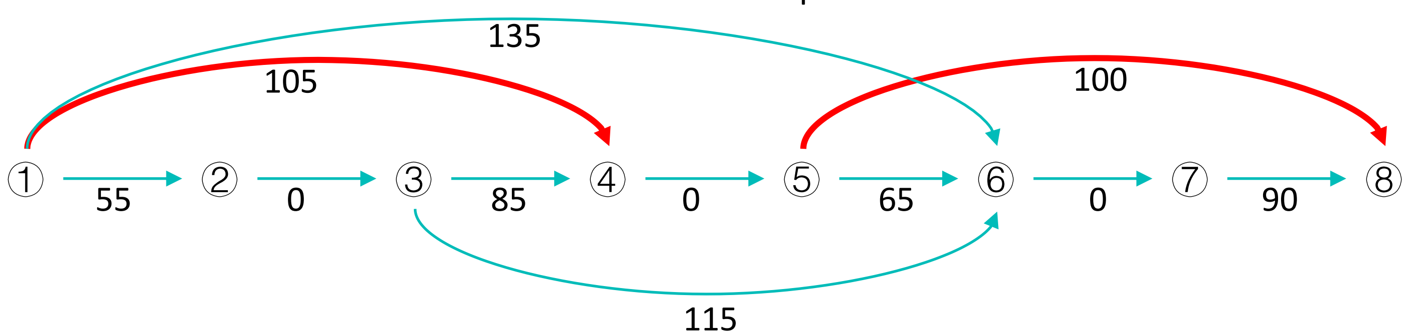
The giant tour T , which is a list of required edges with chosen directions and connected implicitly by shortest paths, is firstly created by other heuristics, such as Path Scanning. Ulusoy Split partition the tour into feasible vehicle routes.

Giant tour: $[\boldsymbol{t_1}, \boldsymbol{t_2}, \boldsymbol{t_3}, \boldsymbol{t_4}]$

Capacity = 50



New Directed Graph



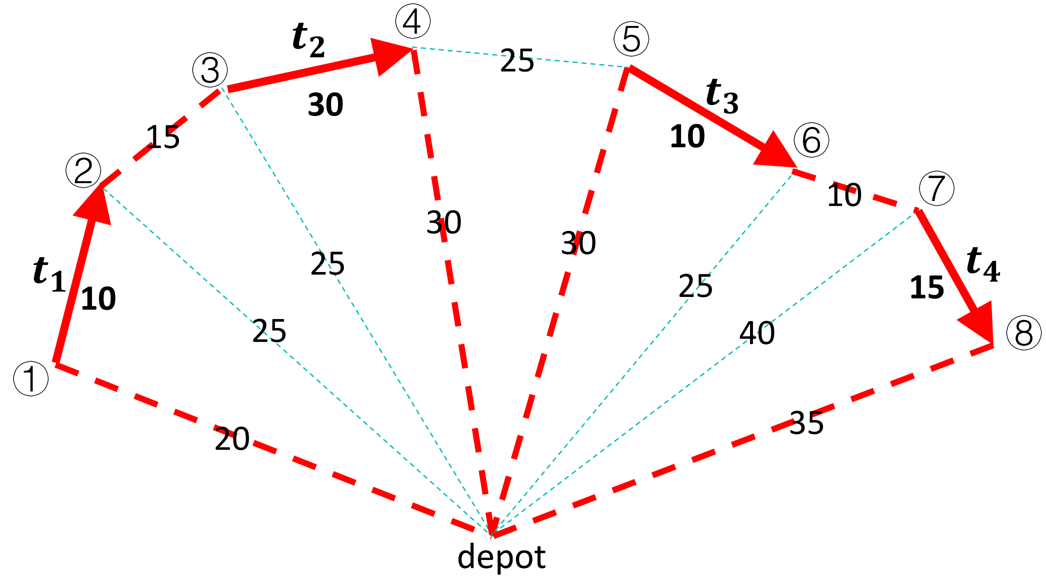
Shortest path on directed graph: ① → ④ → ⑤ → ⑧

Ulusoy Split

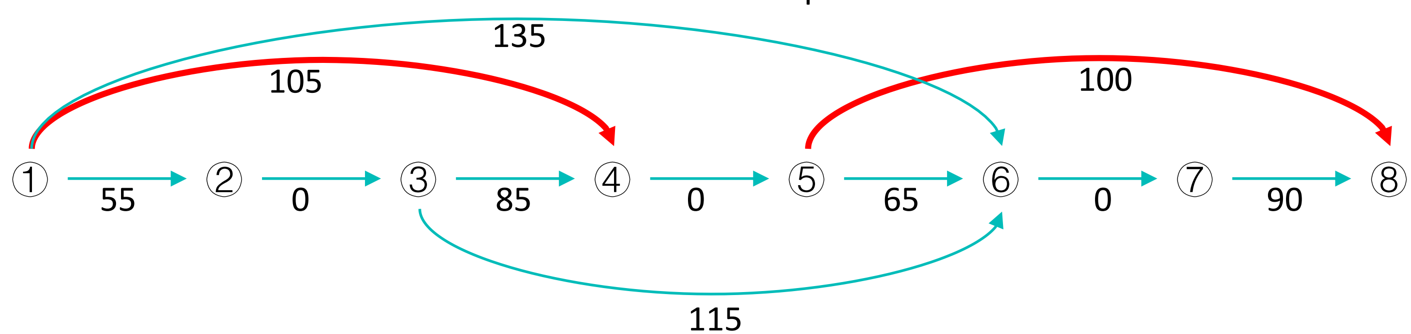
The **giant tour** T , which is a list of required edges with chosen directions and connected implicitly by shortest paths, is firstly created by other heuristics, such as Path Scanning. **Ulusoy Split partition the tour into feasible vehicle routes.**

Giant tour: $[\boldsymbol{t_1}, \boldsymbol{t_2}, \boldsymbol{t_3}, \boldsymbol{t_4}]$

Capacity = 50



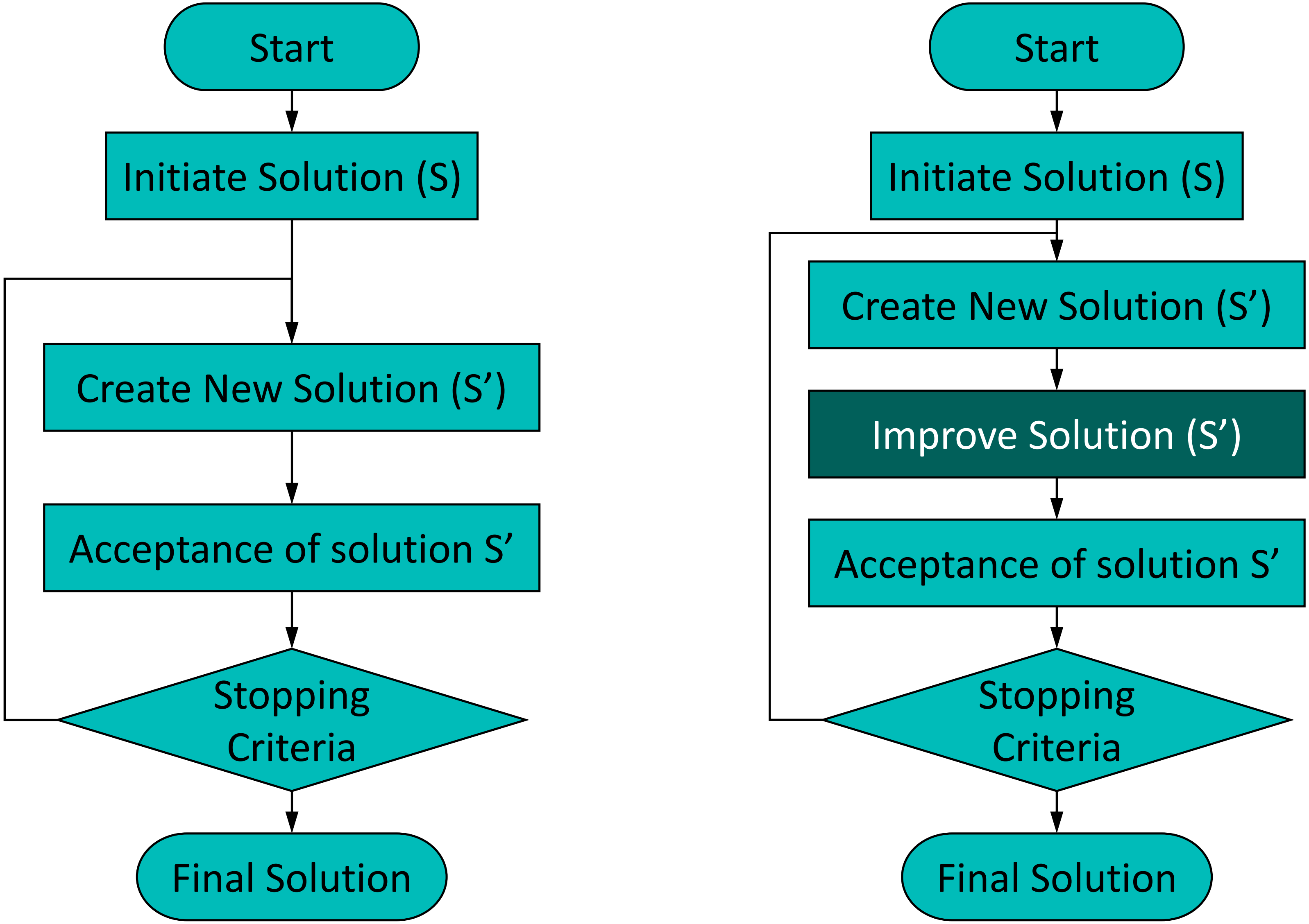
New Directed Graph



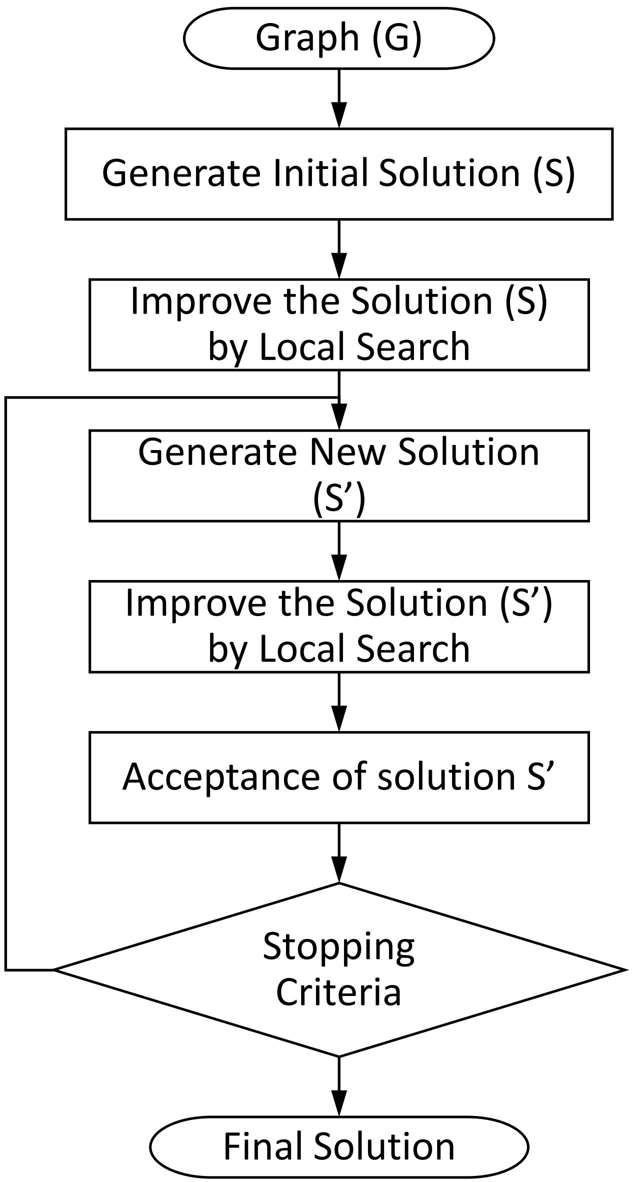
Shortest path on directed graph:

① → ④ → ⑤ → ⑧

A General Framework for Design an Iterative Algorithm for CARP



SAHiD Algorithm



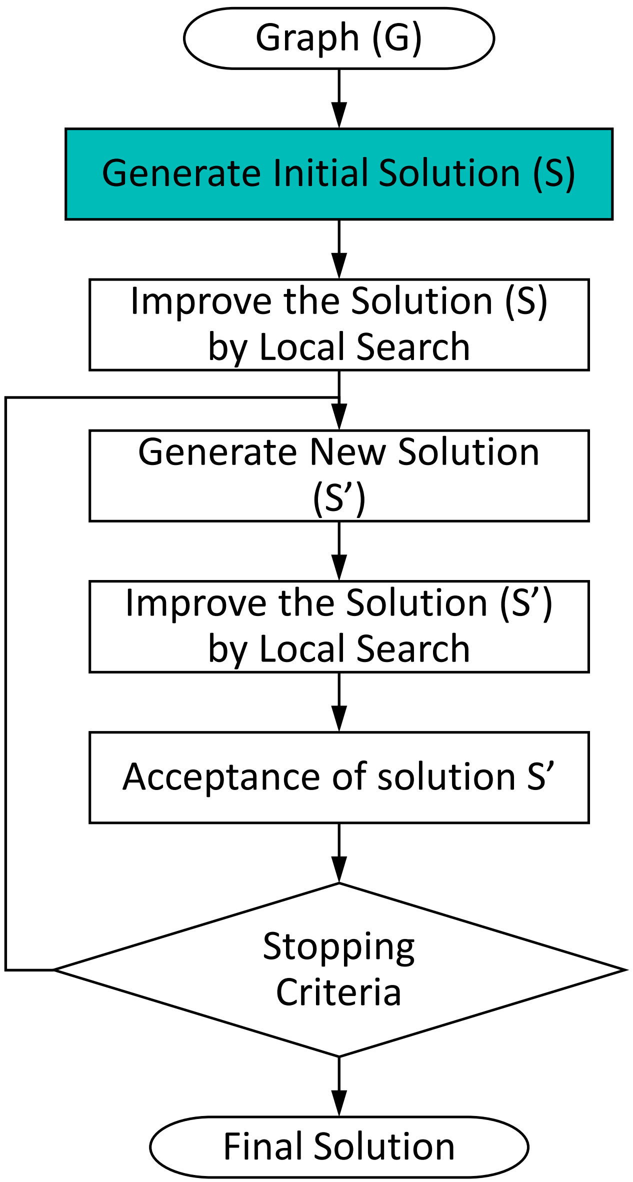
**Algorithm : ** Pseudo Code of SAHiD(T)

Procedure: **SAHiD**(T)
Input: task set T
Output: a feasible solution s^*

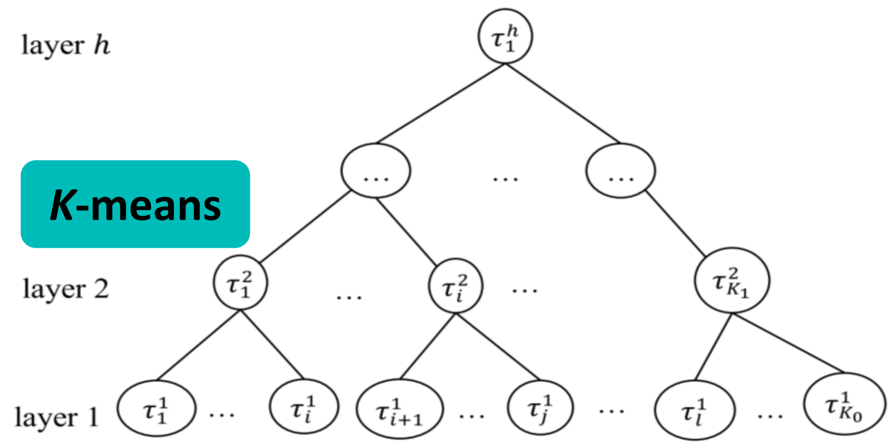
```
generate an initial solution s using HDU(T);
apply LS(s) to improve s;
s* ← s
while stopping criteria are not met do
    generate a virtual task set VT by splitting the routes of s;
    generate a solution s' using HDU(VT);
    apply LS(s') to improve s';
    if s' is acceptable then
        s ← s'
        if s' is better than s* then
            s* ← s'
        end if
    end if
end while
return s*;
```

- **Reference**: Tang, K., Wang, J., Li, X., & Yao, X. (2017). A scalable approach to capacitated arc routing problems based on hierarchical decomposition. *IEEE transactions on cybernetics*, *47*(11), 3928-3940.

SAHiD Algorithm



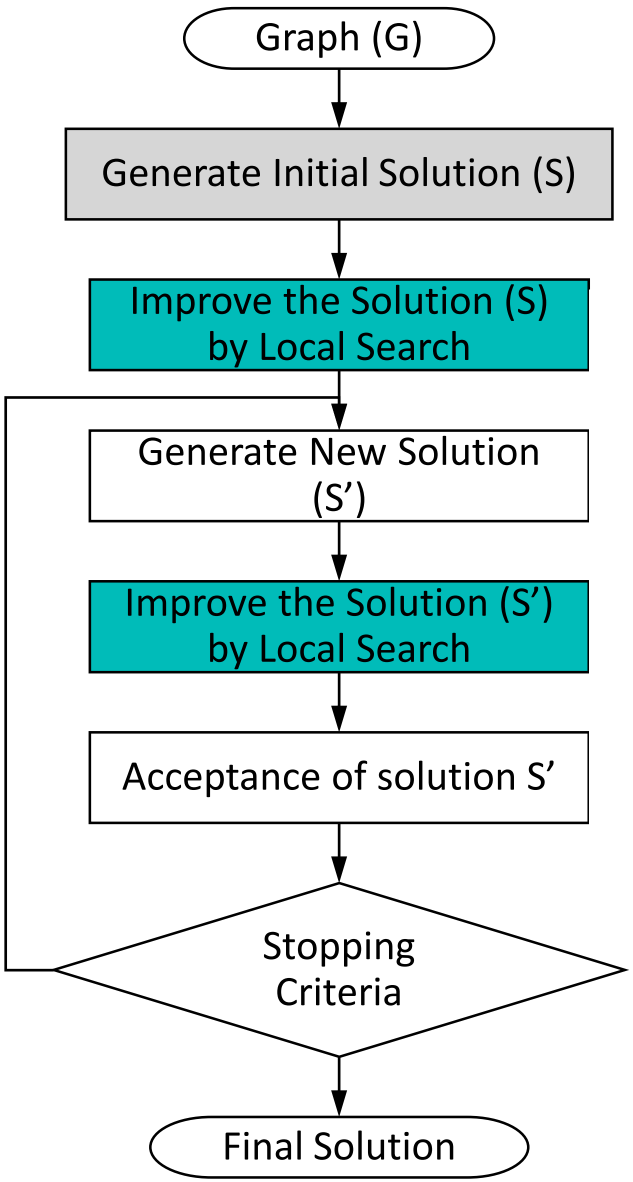
Hierarchical Decomposition and Ulusoy’s Split (HDU)



For example, suppose there are four tasks $\{\tau_1^1, \tau_2^1, \tau_3^1, \tau_4^1\}$ at layer 1, each two of them are connected to a node at layer 2, the virtual task corresponding to this node, denoted by τ_i^2 is a permutation of the two tasks, e.g., $\tau_1^2 = (\tau_2^1, \tau_1^1)$ and $\tau_2^2 = (\tau_3^1, \tau_4^1)$. τ_1^2 and τ_2^2 are then grouped and ordered, forming a virtual task at layer 3, e.g., $\tau_1^3 = (\tau_2^2, \tau_1^2) = (\tau_3^1, \tau_4^1, \tau_2^1, \tau_1^1)$. By this means, a permutation of the four tasks is obtained.

- After the permutation of tasks has been obtained, **Ulusoy’s Split** procedure is applied on the permutation to split the permutation into a number of routes that satisfy capacity constraints.

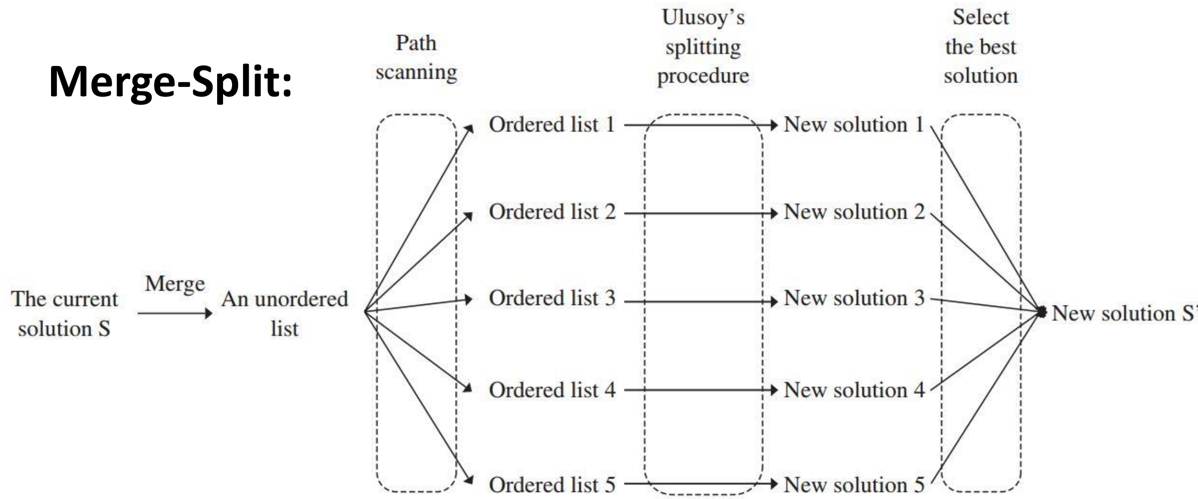
SAHiD Algorithm

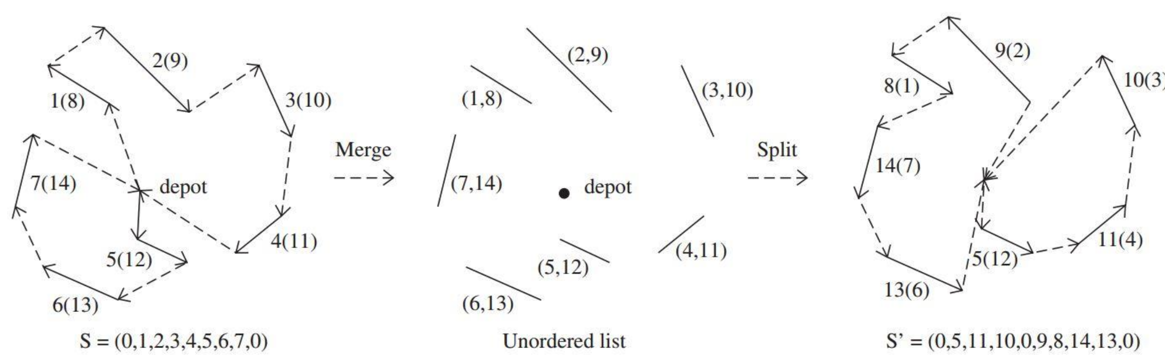


Local Search: Reverse Operator and Merge-Split Operator

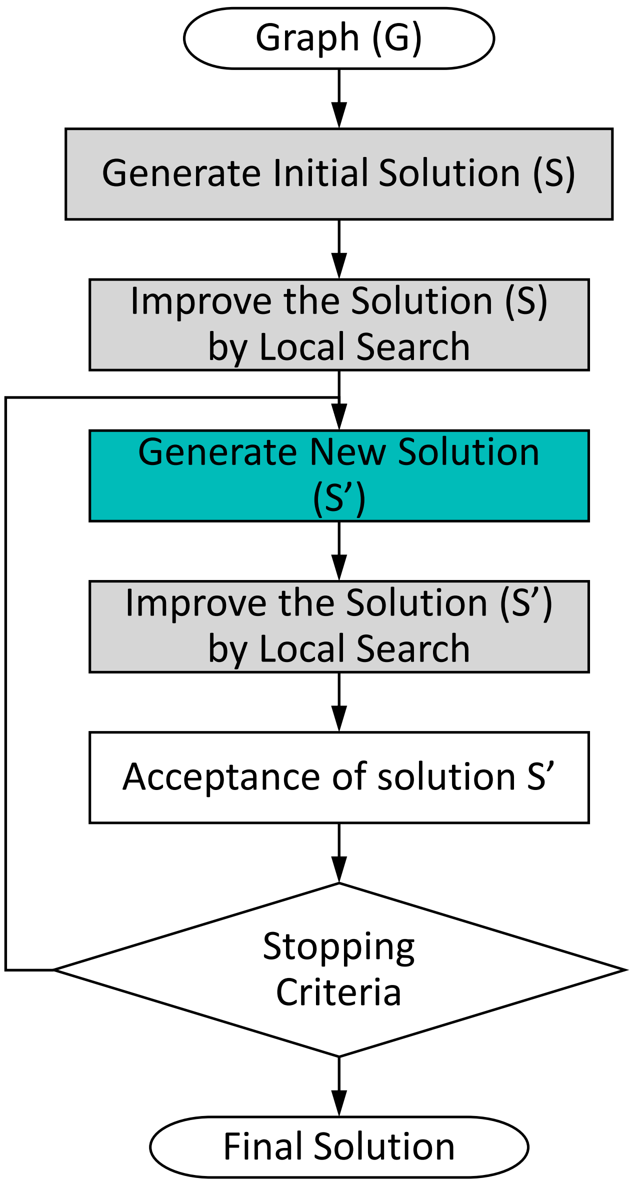
Reverse Operator: all possible sub-routes of each route in solution are enumerated and reversed.

Merge-Split:





SAHiD Algorithm

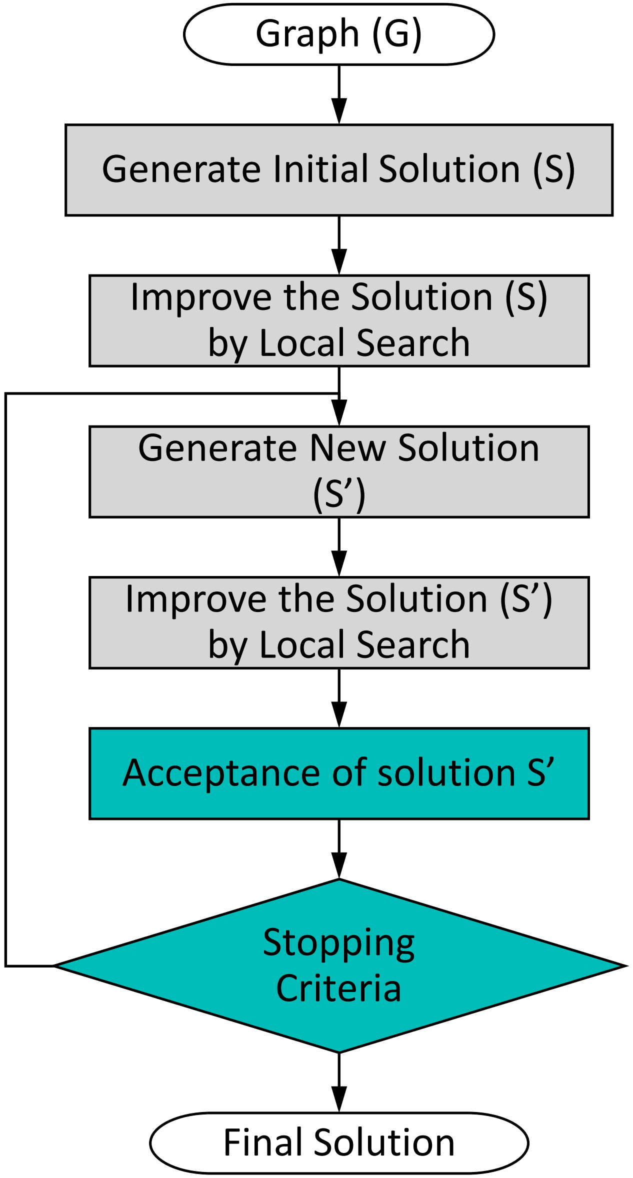


Reconstruction

In the reconstruction phase, HDU is applied to generate new candidate solutions. But different from in the initialization phase, HDU is not applied to achieve a solution from scratch, i.e., based on the un-ordered set of tasks. Instead, the new solution is generated based the solution obtained in the last iteration, say **S**.

- Each route of **S** is split into two virtual tasks with a predefined probability α, resulting in a set of virtual tasks.
- Then, HDU is applied to this set to obtain a new solution **S’**.

SAHiD Algorithm



Acceptance Condition and Stop Criteria

- It might be inappropriate to keep the best solution in the search process of SAHiD if it cannot be improved for a long time. Otherwise, the search will be stuck at this local best solution. Hence, the threshold accepting idea is adopted in SAHiD. Given a solution S, if no better solution is found after σ consecutive iterations, a new solution worse than S will still be accepted (i.e., replace s) as long as its quality is not worse than θ % of that of the best-found solution.
- Finally, the SAHiD can be terminated either when a predefined time budget is used up or no better solution is found for a predefined number of iterations.

Adaptations of Meta-heuristics

Adaptation of Evolutionary Algorithms (EAs):

- **Initialization**: create some initial solutions
- **Reproduction**: Crossover (Recombination) & Mutation
  - Crossover: more than one parent is selected and one or more off-springs are produced using the genetic material of the parents.
    - Order-based Crossover
    - Sequence based Crossover
  - Mutation: make some changes on solutions. [Reverse Operator]
- Evaluation & **Selection**

Thank You !