TU Wien
Institute for Logic and Computation
Algorithms and Complexity Group

Heuristic Optimization Techniques, WS 2025

Programming Exercise: General Information

v1, 2025-07-22

The programming exercise consists of two assignments. This document contains information valid for these assignments, please read it carefully. If you have questions, do not hesitate to contact us either after the lectures or via `heuopt@ac.tuwien.ac.at`.

1 The Selective Capacitated Fair Pickup and Delivery Problem

We study the *Selective Capacitated Fair Pickup and Delivery Problem (SCF-PDP)*, where the goal is to design fair and feasible routes for a subset of n customer requests. Each customer needs transportation of a certain amount of goods from a specific pickup location to a corresponding drop-off location.

The problem is modeled on a complete directed graph $G = (V, A)$, where:

- The node set V includes the vehicle depot, as well as all pickup and drop-off locations associated with customer requests.
- The arc set $A = \{(u, v) : u, v \in V, u \neq v\}$ represents the fastest travel routes between each pair of locations.
- Each arc $(u, v) \in A$ is associated with a travel distance $a_{u,v}$. The distance is calculated as the euclidean distance between the locations rounded up to the next integer. With the x and y coordinates of a location u being denoted as x_u and y_u. The distance is thus calculated as:

$$a_{u,v} = \lceil \sqrt{(x_u - x_v)^2 + (y_u - y_v)^2} \rceil$$

There are n customer requests denoted by $CR = \{1, \dots, n\}$. Each request $i \in CR$ is defined by a pickup location $v_i^{\uparrow} \in V$ and a drop-off location $v_i^{\downarrow} \in V$. We assume that each request concerns the transportation of a certain amount of goods $c_i \in \mathbb{N}$.

Rather than serving all requests, the problem focuses on satisfying only a subset of them. The required number of requests to be fulfilled is given by a fixed parameter γ, where $\gamma \leq n$ and a solution is required to fulfill at least γ requests.

Requests are served by a fleet K of n_K identical vehicles starting from the *depot*, each with a maximum capacity of C.

A feasible solution consists of a route R_k for each vehicle $k \in K$, defined as an ordered sequence of pickup and drop-off locations with the i-th stop of a route being $R_{k,i}$. Each route must fulfill the following properties:

- The vehicle capacity must never be exceeded at any point along the route.
- Each served request must be handled in its entirety by a single vehicle.
- At least γ requests must be served across all vehicles.

The objective is to minimize a combination of the total travel time and a fairness measure among the vehicle routes. To simplify notation we write $(u, v) \in R_k$ to describe a pair of consecutive locations of stops in R_k. The total travel duration of vehicle k is described as follows where $a_{depot,R_{k,1}}$ is the inital travel time from the depot to the first stop and $a_{R_{k,|R_k|},depot}$ is the final travel time from the last stop back to the depot:

$$d(R_k) = a_{depot,R_{k,1}} + a_{R_{k,|R_k|},depot} + \sum_{(u,v)\in R_k} a_{u,v}$$

We define the fairness among all routes using the Jain fairness measure, which is defined as

$$J(R) = \frac{\left(\sum_{k\in K} d(R_k)\right)^2}{n_K \cdot \sum_{k\in K} d(R_k)^2}.$$

The overall objective function to be minimized is

$$\sum_{k\in K} d(R_k) + \rho \cdot (1 - J(R)).$$

where ρ is a weighting parameter that controls the trade-off between total travel time and fairness across vehicle routes.

2 Instances & Solution Format

A problem instance is given as plain ASCII file and contains in the first line, separated by a space, the number of requests n, the number of vehilces n_K, the capacity C, the minimum number of requests to be served γ, and the fairness weight ρ.

The second part of the file separated by the string `#demands` contains the demands of the requests.

The third part of the file separated by the string `#request locations` contains for all locations their x and y coordinates, starting with the depot and followed by each request's pickup and drop-off location.

$n \ n_K \ C \ \gamma \ \rho$
`# demands`
$c_1 \ c_2$ ... c_n
`# request locations`
$x_{depot} \ y_{depot}$
$x_{v_1^\uparrow} \ y_{v_1^\uparrow}$
$x_{v_2^\uparrow} \ y_{v_2^\uparrow}$
...
$x_{v_n^\uparrow} \ y_{v_n^\uparrow}$
$x_{v_1^\downarrow} \ y_{v_1^\downarrow}$
$x_{v_2^\downarrow} \ y_{v_2^\downarrow}$
...
$x_{v_n^\downarrow} \ y_{v_n^\downarrow}$

To calculate the value $d(R_k)$ of a simple example route $R_k = [1, 11]$ in an instance with $n = 10$ requests the expression would be:

$$d(R_k) = a_{depot,1} + a_{1,11} + a_{11,depot}$$