# Combinatorial Optimization
# CoContest Semester Project Assignment:
# **Krocan's Delivery Service™**

Industrial Informatics Department
Czech Technical University in Prague
https://industrialinformatics.fel.cvut.cz/

February 25, 2026

**Abstract**

This document introduces the assignment for the CoContest semester project.

## 1 Motivation

The popularity of Theodor Krocan's business has soared significantly in recent weeks. His new "pills" are flying off the shelves, and yet none of his clients seem keen on picking them up directly from the depot. Instead, they all insist on home delivery at specific times. Therefore, each morning, Theodor faces a daily dilemma. He must carefully plan each morning's delivery routes of his vans to minimize transportation costs while maximizing his profits. Indeed, although he controls a nearly unlimited number of super vans, he is forced to pay a significant amount of money to convince a driver to leave for delivery, and each mile driven incurs gas expenses. Moreover, a driver refuses to perform more than one tour per day.

But he has to take into account that:

- Each customer orders one parcel, and it must be delivered only within an imposed time window.
- Although Krocan has an unlimited number of vans available, each van cannot perform more than one tour per day.

This problem will be solved in two phases. In the first phase (**optimal**), we will consider that Theodor has a limited number of vans. He will seek to search for an optimal planning for them. In the second phase (**heuristic**), we will consider an unlimited number of vans.

## 2 Formal Problem Statement

Let us consider a series of vans $\mathcal{V} = \{V_1, \ldots, V_k\}$ attached to a unique depot $o$, and a list of customers $\mathcal{C} = \{C_1, \ldots, C_n\}$. Each customer $i$ is associated with a time window $(\underline{T}_i, \overline{T}_i)$. The parcel belonging to this customer (we assume one parcel per customer only) can be delivered to him only during this time window. Each parcel has a size $s_i \in \mathbb{N}^*$. All vans have the same loading capacity $Q$. We define $T$ and $C$ as the matrices of travel time and travel cost, respectively. Thus, for any two customers $(u, v)$, $T_{uv}$ is the travel time required to reach $v$ from $u$, and $C_{uv}$ is the cost to do so. We can extend these notations so that $T_{ou}$ and $C_{ou}$ denote the travel time and travel cost from the van depot to customer $u$, respectively. And vice versa for $T_{uo}$ and $C_{uo}$. Both the travel times and travel costs are symmetrical (i.e., $\forall(u, v)$, $T_{uv} = T_{vu}$ and $C_{uv} = C_{vu}$). We do not limit the amount of time a van can wait at a customer's place. The van path $P(d) =$

$\{(o,u),(u,v),\ldots,(w,o)\}$ is the vector of each tuple $(u,v)$ the van $d$ passed by, starting and ending at the depot. Figure 1 depicts a solution with 2 vans and 5 customers. We require that a van leave the depot at most once.

The objective is to minimize both the total travel cost to deliver all of the parcels and the number of vans used in the solution, i.e.,

$$\min \sum_{d \in \mathcal{V}} \sum_{(u,v) \in P(d)} C_{uv} + \Gamma \cdot \mathbb{1}_d$$

where $\mathbb{1}_d$ is the function indicator which equals 1 if the truck $d$ is used, 0 otherwise, and $\Gamma$ is the cost of using a truck (this cost is fixed and the same for all trucks).

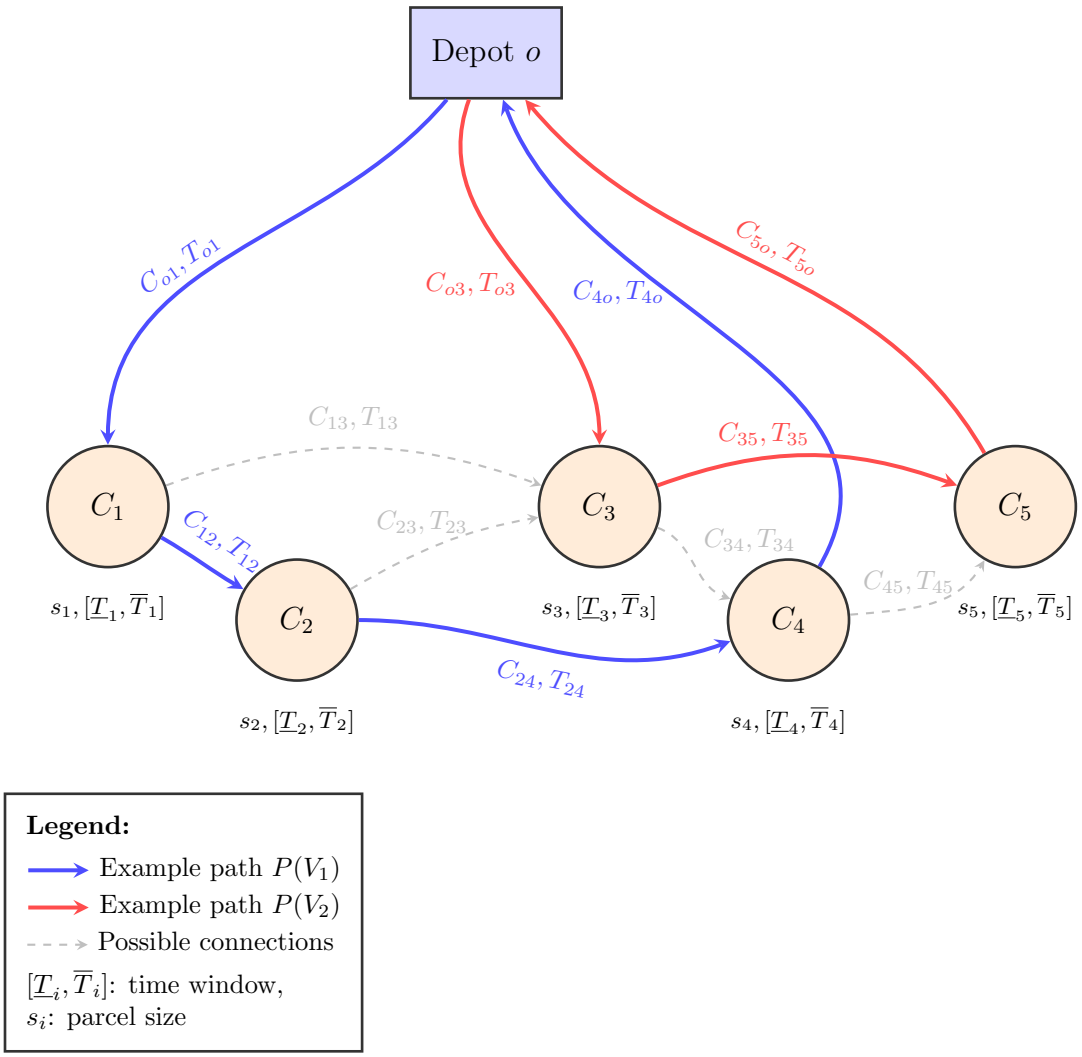


Figure 1: Schematic representation of a solution of the *Krocan's delivery service* problem with 5 customers and 2 vans.

In the optimal part, we will consider that the number of vans is bounded to a value $K \in \mathbb{N}$.

# 3 Rules

If you decide to choose CoContest as your semester project, then you are expected to implement a correct solver for Krocan's Delivery Service™ problem. The implementation will be submitted to

BRUTE https://cw.felk.cvut.cz/brute/ where it will be automatically evaluated (the number of submissions is not limited). The grading is a combination of the ability to find good solutions and the achieved rank relative to other students (w.r.t. the objective function). Therefore, you can acquire a small number of points even if your solver is not very efficient relative to other students.

In BRUTE, you will find 3 tasks related to the contest. Each task has specific instances, rules, and grading. The contest is split into different tasks to avoid re-evaluation of the instances (which is time-consuming) and so that you can implement a specific solver for each task.

1. `SP_CC_O`: you have to formulate an ILP model using Gurobi. If your solver solves all the instances in this task optimally, then you will get **3 points** for this task. If the solver returns a suboptimal solution for **ANY** instance in this phase, then the evaluation of your solver is stopped, and you will get 0 points in this task.

2. `SP_CC_T`: the goal is to find the best possible feasible solution within the specified time limit. The optimal solutions are not required, and you are encouraged to implement clever heuristics to solve these instances. For each instance in this task, you will obtain some fraction of the point if your solution's cost is not worse than our threshold (**4 points** at max).

3. `SP_CC_R`: similarly to `SP_CC_T`, in this task, we are also interested in finding the best possible feasible solution within the specified time limit. However, your solver's evaluation will depend on how good your solver is relative to other students' solvers, i.e., the number of points obtained will depend on your rank (**4 points** at max).

Some general contest rules also apply:

1. Usage of the single-purpose problem-specific solvers is prohibited (i.e., a MILP solver is allowed, but somebody else's code for solving "Krocan's Delivery Service™" like problem is not).

2. Every participant is required to write their own code. However, sharing ideas and discussing the problem is fostered.

# 4 Input and Output Format

In `SP_CC_O`, your solver will be called as

```
$ ./your-solver PATH_INPUT_FILE PATH_OUTPUT_FILE
```

whereas in `SP_CC_T` and `SP_CC_R` we include a time limit

```
$ ./your-solver PATH_INPUT_FILE PATH_OUTPUT_FILE TIME_LIMIT
```

- `PATH_INPUT_FILE` and `PATH_OUTPUT_FILE`: similarly to homework, these parameters represent the path to the input and output files, respectively (see below for a description of the file formats).

- `TIME_LIMIT`: a float representing the time limit in seconds given to your solver. Your solver will be killed after the time limit is reached, and you will be awarded 0 points. Hence, your solver's output is considered only if your program exits with a status code 0 before it times out.

## 4.1 Input

The input file has the following form (we use one space as a separator between values on one line):

$$
\begin{array}{llllll}
N & K & Q & \Gamma & & \\
s_1 & \underline{T}_1 & \overline{T}_1 & & & \\
\vdots & & & & & \\
s_N & \underline{T}_N & \overline{T}_N & & & \\
T_{oo} & T_{o1} & T_{o2} & T_{o3} & \dots & T_{oN} \\
T_{1o} & T_{11} & T_{12} & T_{13} & \dots & T_{1N} \\
\vdots & & & & & \\
T_{No} & T_{N1} & T_{N2} & T_{N3} & \dots & T_{NN} \\
C_{oo} & C_{o1} & C_{o2} & C_{o3} & \dots & C_{oN} \\
C_{1o} & C_{11} & C_{12} & C_{13} & \dots & C_{1N} \\
\vdots & & & & & \\
C_{No} & C_{N1} & C_{N2} & C_{N3} & \dots & C_{NN}
\end{array}
$$

$N$ is number of customers, $K$ is maximum number of vans. $Q$ is capacity of the van and the $\Gamma$ is cost of using a van. $s_i$, $\underline{T}_N$, $\overline{T}_N$ are parcel size and time window of customer $C_i$. $T_{ij}$ is time required to travel from customer (or depot) $i$ to $j$, $C_{ij}$ is cost of doing so. ***Do <u>not</u> forget*** that matrices $T$ and $C$ contain $N + 1$ rows and columns!

## 4.2 Output

The output file must have the following layout:

$$
\begin{array}{llllll}
Obj & \text{K} & & & & \\
\nu_1 & C_1^1 & t_1^1 & \dots & C_{\nu_1}^1 & t_{\nu_1}^1 \\
\vdots & & & & & \\
\nu_k & C_1^k & t_1^k & \dots & C_{\nu_k}^k & t_{\nu_k}^k
\end{array}
$$

where $Obj$ is the objective value of the solution, $K$ is the number of vans used in your solution. Then, each line $d$ defines the tour of the van $d$. $d$ $\nu_d$ corresponds to the number of parcels this van transports. Then, for all $\iota \in \{1, \dots, \nu_d\}$, $C_\iota^d$ and $t_\iota^d$ are the customer and the time at which the van $d$ arrives at this customer, respectively.
If the solution is infeasible, just write "$-1$" in the file.

## 4.3 Example

The following example presents an instance with 3 customers and 2 vans, and its optimal solution.

Input:

```
3 2 2 5
1 0 10
1 3 8
2 2 5
0 1 1 1
1 0 2 5
1 2 0 5
1 5 5 0
0 3 7 4
3 0 10 5
7 10 0 1
```

```
4 5 1 0
```

Output:

```
38.0 2
2 1 1 2 3
1 3 5
```