Leonora Frangu, Fabio James Greenwood, Giada Palma, Benedetta Pasqualetto

**Discrete Optimisation Project Documentation**

**Planning and multimodal transportation**

**Index**

[Requirements 2](#_Toc118109909)

[Info 5](#_Toc118109910)

[Missing Constraints 5](#_Toc118109911)

[Known Bugs 5](#_Toc118109912)

[Assumptions 5](#_Toc118109913)

[Clarifications 5](#_Toc118109914)

[Indexes and Sets 6](#_Toc118109915)

[Special Subsets 6](#_Toc118109916)

[Constants 6](#_Toc118109917)

[Independent Variables 7](#_Toc118109918)

[Semi-Dependant Variables 8](#_Toc118109919)

[Dependent Variables 8](#_Toc118109920)

[Objective functions 8](#_Toc118109921)

[Constraints 8](#_Toc118109922)

[Basic Conservation of Flow 8](#_Toc118109923)

[Task Timing 9](#_Toc118109924)

[Bus Travel Constraints 10](#_Toc118109925)

[Bike Constraints 11](#_Toc118109926)

[Personal Spend Constraints 11](#_Toc118109927)

[Maximum people in a place 12](#_Toc118109928)

[Transportation mode change 12](#_Toc118109929)

[Optional Test Constraints 12](#_Toc118109930)

[Metaheuristic adaptation: SCATTER SEARCH 13](#_Toc118109931)

# Requirements

|  |  |  |
| --- | --- | --- |
| #1  OBJECTIVE FUNCTIONS | Determine the tours that all people have to do in order to perform their tasks, by considering the different modes of transportation available and all the constraints, in order to minimize the total travelled time by also taking into account the overall fitness score. | Not implemented |
| #2  PEOPLE | It is a new day in Verona, and a set N := {1, . . . ,N} of people is ready to perform all their errands as efficient as they can be.  Each person n ∈ N, who starts from a known point Hn (e.g., their home) with a given amount of money Wn, has a certain set Tn of tasks to do. | All Done |
| #3  TASKS | Each task t may cost ct €, has to start during the related time window [at , bt ], and generally lasts τt minutes.  Some special tasks t∗ ∈ T∗ n ⊆ Tn, either t∗ has to start in the related time window, or it can be started anytime. However, in this latter case, the duration of t∗ will be increased by a certain amount of minutes.  Each task t occurs in a different place p. | All done |
| #4  PLACES | Let P := {1, . . . , P} be the set of all places of the tasks of all people in N (e.g., shops, squares, bars, facilities, etc.).  Each place p ∈ P is described by a latitude value latp and a longitude value lonp. | Not implemented |
| #4.1  MAX NUMBER OF PEOPLE | Also, a place p cannot be visited by more than Np people at the same time. | Not implemented |
| #5  MODES OF TRANSPORTATION | To reach the places associated to their tasks Tn, each person n is allowed to either walk, or to ride a bike offered by the bike-sharing company in Verona, or also to take a bus, according to the distribution of the bike-sharing stations and the bus lines in the city. | NA |
| #6  TRANSPORTATION MODES CHANGES | Let M:= {Walking , Cycling, Bus} be the set of modes of transportation.  Each person n ∈ N can change their mode of transportation at most Kn times, and, in the end, n must return to their starting point Hn. | Red not implemented |
| #6.1  PENALTY | A penalty Qt must be paid for a task t ∈ Tn not performed. | None Implemented |
| #7  BIKES AND BIKE STATIONS | Let B := {1, . . . ,B} be the set of bike stations.  For each bike station b ∈ B, the number αb of bikes available at the beginning of the day is known, as well as the number βb of free spots where to leave a bike (of course, if all spots are available, the maximum capacity of a bike station b is Nb := αb + βb).  Similarly, to any place each bike station b is associated with a latitude value latb and a longitude value lonb. | Capacity not checked |
| #7.1  BIKE COSTS | Riding a bike costs Cbike €/min. | Not implemented |
| #8  BUSES AND LINES | Let L := {1, . . . , L} of lines is available in the city.  Each line ℓ is composed by a set of stops Sℓ := {Dℓ, . . . , Sℓ}, and starts from a certain depot Dℓ at time δ. | All Implemented |
| #8.1  RIDES FREQUENCY | Then, there is a ride every δ minutes. | All Implemented |
| #8.2  MAX NUMBER OF PEOPLE | The maximum number of people who can be on a bus at the same time is Nℓ. | Not Implemented |
| #9  STOPS AND  LINES CONSTRAINTS | Each stop s ∈ Sℓ is associated with a latitude value lats and a longitude value lons .  No two lines Sℓ1 and Sℓ2 have two identical consecutive stops.  Anyway, some lines Sℓ1 and Sℓ2 may intersect at some stops (i.e., Sℓ1 ∩ Sℓ2 ̸= ∅). | NA |
| #10  BUS SELECTION AND CHANGE | Every time that a person n chooses the bus to change their mode of transportation, they pay Cbus €.  If n just chooses to change the bus line in a stop, without changing the mode of transportation, they do not have to pay another ride. | Half Implemented |
| #11  NETWORK | Let H be the set of all starting points Hn, for every n ∈ N.  Let S be the set of all bus stops, for every line ℓ ∈ L.  Let G := (V,A) the directed weighted multi-graph such that V := H ∪ P ∪ B ∪ S is the set of all points in the city.  Let A := {(i , j ,m) | i , j ∈ V,m ∈ M} be the set of all arcs connecting point i to point j by using mode of transportation m. | Unsure about how to impliement |
| #12  TRAVELLING TIME | Each arc is associated with a travelling time tm,i,j | Half Implemented |
| #13  FITNESS COEFFICIENT | Fitness coefficient fm,i,j ∈ [−1, 1], which represents a sort of “health gain” or “health loss”, according to the mode of transportation chosen. | Not Implemented |

# Info

For complex constraints a line explaining what the constraint is controlling should be placed above. *[An additional detailed note about how exactly if works can be placed above the constraint in square brackets and italics if needed, keep this to a minimum]*

Please just steal maths from the file “DODM\_07\_ILP-VehicleRoutingProblems.pdf” and leave comments about what you’re stuck on [Review Tab > New Comment]

# Missing Constraints

* The capacity of nodes and modes of transports is not yet considered by the model

# Known Bugs

* Arrival times at nodes seem to not be working as expected

# Assumptions

* A person will not return to a node they have personally travelled to (this needs to be confirmed)
* A person will only complete a single task at each node
* A bus spends no time at a bus stop
* A bus can be caught anytime within a x second window before it arrives at a point. This time is called the bus relaxation (I assume 60 seconds is fine, however we can tune this)
* The travel times of two people riding the same bus are modelled separately and may lag between each other according to the bus relaxation parameter
* The indexing of modes of transport is currently assumed to be {1 = walking, 2 = Cycling, 3 = Bus}. I will be worth asking the prof if we are ok to always assume this is the case. Otherwise, there will have to be some hardcoding in the model to catch instances in the inputs where this is not the case.
* The bike station capacity is checked at x min intervals (default 30 mins), there will have to be a consultation about how this linear relaxation could be best handled
* People must leave their home node
* People don’t do tasks on their home node
* Task penalty is hardcoded to 1000
* Each person walks out their house

# Clarifications

* The objective function minimising travel time and doesn’t factor task time or idle time (inc waiting for bus)

# Indexes and Sets

# Special Subsets

# Constants

# Independent Variables

# Semi-Dependant Variables

***These are the variables that are technically dependant variables but are modelled as constrained independent variables***

# Dependent Variables

***This section is saved for any variable which are fully dependant on other variables, this is generally used if various variables need to be consolidated into a single figure i.e., the sum of costs. Please note that when using these variables, the developer needs to be careful not to make any illegal calculations this new variable***

This section is currently empty

# Objective functions

Determine the tours that all people have to do in order to perform their tasks, by considering the different modes of transportation available and all the constraints, in order to minimize the total travelled time by also taking into account the overall fitness score.

# Constraints

## Basic Conservation of Flow

Flow is directional, multi-medium, multi-flow (person)

*[the entries/exits from a node j, needs to be larger than one if there is a task at the node or if it is the home node of the person (h) (Boolean values)]*

Currently each node can only be visited once to not interfere with constraints around timing

The person must exit their home on foot

The person must exit a node they visit

## Task Timing

This controls the time (w) which the person arrives at node starts

*This doesn’t apply where a person is returning to their home node*

*This doesn’t apply where a person is returning to their home node*

This controls that tasks happen within their designated time windows (only applies if the tasks happens and the task extension penalty for falling out of the allotted time isn’t applied)

Task extensions/delays only apply to tasks that are undertaken

Task extensions/delays only apply to tasks that are special

A person will only complete a single task a node they visit

A person leaves their home sometime after the start of the day

All time variables must be within the bounds of the day

## Bus Travel Constraints

A person must be ready at a bus stop in time for the arrival of the bus

*The bus variable signifies if a person gets a certain bus line, at a certain node and departure time. DTime is the arrival time of the bus and BusRelaxation creates a small window for the person to be available at the bus stop*

A person can only catch a bus once from a given node

The person n can only get bus (i, d, r) the relevant Bus\_idrn must be positive

Every time a person gets on a bus from another mode of transport, they must purchase a bus fare

A maximum number of people who can be on a bus at the same time is Nℓ

## Bike Constraints

A person must start and end their bike travel at a bike stop

*If a person arrives at a non-bike stop via bike, they must leave by bike*

The bike station capacity is checked at x min intervals (default 30 mins), there will have to be a consultation about how this linear relaxation could be best handled

At the end of the period, there must be a non-negative quantity of bikes and free spaces at each bike station

## Personal Spend Constraints

The money spent by a person should be below their personal budget

*A person is only charged for bike hire while they are in motion*

## Maximum people in a place

A place p cannot be visited by more than Np people at the same time (work in progress)

*Basic concept:*

* *Occupancy tracking is refreshed every x minutes (default 60)*
* *If a person is in that area for a given period, he is modelled to be occupying for the whole period*
* *Note that the second condition requires a multiplication of a binary and continuous variable, so that must be converted accordingly*
* *Occupancy variable must follow the following conditions. It must be forced to be positive if:*
  + *w\_i for the node is larger then the start time of the period*

*or*

* + *the w\_j for the node the person travels to next is smaller then the end of the period*

*however:*

* *If w\_i for the arrive is between the start and end of the period the occupancy variable must be positive*

*The number of people at a places is checked every x minutes*

*They must of an O token if they enter the node (potentially not enought)*

*The right o token*

## Transportation mode change

Each person n ∈ N can change their mode of transportation at most Kn times

## Optional Test Constraints

This is an optional constraint to test if the bus functionality works

# Metaheuristic adaptation: SCATTER SEARCH

The picture shows the basics steps of the scatter search algorithm.

Immagine che contiene testo

Descrizione generata automaticamente

The next points show the pseudocode for a simple implementation of the scatter search algorithm

1. DIVERSIFICATION GENERATION METHOD

We start with , where P must be a large set of solutions, i.e.   
Then, we want to use the *diversification generation method* to populate as a collection of **diverse trial solutions**, using a **seed solution** as input.   
  
First, we want to create a **seed solution**.  
This can be done by calculating the Euclidean distance between places, where the tasks are to be performed.  
  
Immagine che contiene testo

Descrizione generata automaticamente

Second, we want to generate **diverse trial solutions**.  
This can be done by generating them as a list of permutations of the **seed** matrix and their random paths between places.

Immagine che contiene testo

Descrizione generata automaticamente

1. IMPROVEMENT METHOD  
   We want to improve the initial set of solution by applying an *improvement method.*So we can apply any local search algorithm, for example a **2-opt heuristic**

Immagine che contiene testo

Descrizione generata automaticamente

Let be the resulting solution.

If then add to (i.e., ), otherwise discard .  
Repeat until .

1. REFERENCE SET UPDATE METHOD

We use this method to build with the best solutions in .

We then want to sort the solutions in according to the objective function value,

s.t. is the best solution and the worst.

Immagine che contiene testo

Descrizione generata automaticamente

1. SEARCH PHASE, SUBSET GENERATION METHOD and SOLUTION COMBINATION METHOD

The search phase is initiated by setting the Boolean variable , and we start the iteration process.

The first operation in the loop is to generate a subset of solutions, (i.e., ) from the , as the basis for creating combined solutions. The *subset generation method* is focused on generating subsets of size 2 resulting in new subsets. The pairs are selected one at a time in lexicographical order.

Moreover, we set the Boolean flag to false, (i.e., )

Then, another while loop is initiated under the condition that the .

We select a subset from and we apply the *solution combination method* to obtain new trial solutions.

Then, we apply the *improvement method* on the new trial solutions, and the *reference set update method* again to a new containing the best solutions coming from both the and the new trial solutions.

If the has changed after the update method the flag is reset to true, (i.e., ) and the subset subjected to the combination is deleted from the .

Immagine che contiene testo

Descrizione generata automaticamente

TERMINATION CRITERION

The algorithm terminates after all the generated subsets in have been passed to the *solution combination method* and none of the enhanced trial solutions are admitted to .