Model-driven decision-making methods: Project 12

Federica Di Pasquale: 493195

Contents

1 Description of the problem

A software house is implementing a suggestions system for tourists visiting a given city. To this aim it has mapped the set of "points of interest" (PoI) in the city; these can be represented by a directed graph with a node for each PoI (plus one for the current location of the tourist), and a directed arc for each pair of nodes (in both directions) labeled with the distance between the two (the distance can be different between the two directions due to one-way streets), expressed in walking minutes. From the preferences expressed by the tourist and their own database, the software house computes numerical value which encodes how much the tourist would like to visit each PoI, provided it stops there for a given minimum amount of time (depending on the specific PoI). The starting time for the tour is given, as well as its maximum duration; furthermore, each PoI can only be visited for a given time window in the day. The problem is to find the tour (starting from the current location of the tourist and ending back there) that maximizes the sum of the preferences of the visited PoIs while satisfying the constraint on the maximum duration.

1.1 Model

We can model our problem using a complete direct graph $G(N, A)$ with $|N| = n + 1$ nodes, where n is the number of PoIs and the $+1$ is referred to the current position of the tourists, which is represented by node number 0. The maximum duration of the tour is T and the starting time is t_{in}.

The other parameters of the problem are:

- q_i: how much the tourist would like to visit node i;

- r_i: time spent in node i;
- $[t_1^i, t_2^i]$: time windows, it is $[t_{in}, t_{in}+T]$ for $i = 0$;
- d_{ij}: distance between nodes i and j (in walking minutes).

The variables are:

- x_{ij}: binary variable for each arc $(i,j) \in A$ which is 1 if the arc (i,j) belongs to the selected tour;
- a_i: continuous variable that represents the arrival time in node i if the node is visited.

The **ILP** model is:

$$\max \sum_{(i,j)\in A} x_{ij} q_j$$

$$\sum_{(0,j)\in FS(0)} x_{0j} = 1 \tag{1}$$

$$\sum_{(i,j)\in FS(i)} x_{ij} - \sum_{(j,i)\in BS(i)} x_{ji} = 0 \qquad \forall i \in N \tag{2}$$

$$\sum_{(i,j)\in FS(i)} x_{ij} \leq 1 \qquad \forall i \in N \tag{3}$$

$$a_i \geq (t_{in} + d_{0i})x_{0i} \qquad \forall i \in N \setminus \{0\} \tag{4}$$

$$a_i \geq a_j + d_{ji} + r_j - (1 - x_{ji})M_{ij} \qquad \forall i \in N, \forall j \in N \setminus \{0\} \tag{5}$$

$$t_1^i \leq a_i \leq (t_2^i - r_i) \qquad \forall i \in N \tag{6}$$

$$x_{ij} \in \{0,1\} \qquad \forall (i,j) \in A \tag{7}$$

The constraints are the same as for a **TSP** problem with time windows, more in detail:

- Constraints (1), (2), and (3) guarantee that there is an activated outgoing arc from the initial position of the tourists, that each incoming arc corresponds to an outgoing arc and that each node is traversed at most once.
- Constraint (4) guarantees that the arrival time of the first visited node is at least equal to the initial time plus the time d_{0i} needed to get there; then we want that each arrival time is equal to the arrival time of the previous visited node plus the time spent in that node and the time needed to reach the current node, so:

  $$a_i = \sum_{j\in N} (a_j + d_{ji} + r_j)x_{ij} \qquad \forall i \in N \setminus \{0\} \tag{8}$$

  this is *non linear* but can be linearized obtaining constraint (5) which is equivalent, in fact if $x_{ij} = 1$ then:

  $$a_i \geq a_j + d_{ji} + r_j \tag{9}$$

  but, since the objective function maximizes the number of visited nodes, the solver will choose the minimum feasible value for a_i.

  On the other hand, if $x_{ij} = 0$ then:

  $$a_i \geq a_j + d_{ji} + r_j - M_{ij} \implies M_{ij} \geq a_j + d_{ji} + r_j - a_i$$

we don't know the values of a_i and a_j but for sure $t_2^j - r_j$ is an upper-bound for a_j and t_1^i is a lower bound for a_i, therefore it's enough that M_{ij} is:

$$M_{ij} \geq t_2^j + d_{ji} - t_1^i \tag{10}$$

so we can choose the smallest possible value $M_{ij} = t_2^j + d_{ji} - t_1^i$ and (5) is always satisfied.

- Finally (6) are the *time windows* constraints that guarantee that the arrival times are within the time window and, therefore, also the maximum duration constraint is satisfied.

1.2 Data

In order to generate some reasonable data for this problem, we supposed that the tour is only possible within 12 hours from 9.00 a.m. to 9.00 p.m. and we measured all in minutes. 9.00 a.m. corresponds to minute 0 and 9.00 p.m. to minute 720.

Instances of this problem were generated and stored in txt files with the following format:

$$\begin{array}{l}
N \quad T \quad T_{in} \\
q_0, \dots, q_n \\
r_0, \dots, r_n \\
t_1^0, t_2^0 \\
\vdots \\
t_1^n, t_2^n \\
d_{00}, \dots, d_{0n} \\
\qquad \vdots \\
d_{n0}, \dots, d_{nn}
\end{array}$$

The first line contains:

- N, T_{in} and T that are respectively the number of PoIs, the starting time and the maximum duration of the tour;

In the code it is also possible to modify the following parameters:

- R: the maximum time spent in each node;
- W: the maximum width of a time window;
- D: the maximum distance between two nodes.

According to these parameters, data are generated randomly, in particular:

- q_i (preferences) are supposed to be measured as a percentage. q_0 is 0 because it is the starting node, while the others are random numbers between 0 and 100;
- r_i is the time spent in each node, so $r_0 = 0$ and the others are random numbers between 0 and R (measured in minutes);
- t_1^i is a random number between 1 and $(720 - W)$ and t_2^i is equal to $t_1^i + r_i$ plus a random number between 0 and $W - r_i$. For the starting node the time window is $(T_{in}, T_{in} + T)$.
- in order to produce d_{ij}, we generated random points in the unitary square and then we considered three different possibilities:

- starting point in the center of the square;
- starting point in the origin;
- random starting point.

then we computed the euclidean distance between each pair of points plus a small random correction and we multiplied by D.

We generated instances for each combination of number of nodes N $(10, 15, 20)$, maximum duration of the tour T $(300, 480)$ and position of the starting node (STARTING_POINT $= 0$ origin / 1 center / 2 random). The other parameters are set to $T_{in} = 120$, $R = 30$, $W = 480$ and $D = 120$.

2 Implementations and results

The model was implemented in the mathematical programming language CMPL that provides an interface between different solvers, including CBC and CPLEX, which are the ones we considered.

The first goal was to identify the smallest size of the instances not solvable in the time limit and try to understand from the log which are the parameters of the solver that could be tuned to improve the gap. In all experiments a time limit of 5 minutes has been set and nodes have been preprocessed as explained below:

2.1 Preprocessing

In the preprocessing phase all the variables not compatible with the costraints of the problem have been deleted. In particular:

Nodes

- All the nodes i for which:
$$t_2^i - r_i - d_{0i} < t_{in} \tag{11}$$
cannot be reached even from the initial node and therefore not from any other node.
- Similarly, all the nodes i for which:
$$t_1^i + r_i + d_{i0} > T \tag{12}$$
cannot satisfy the total time limit constraint.

For all of these nodes the corresponding variables x_{ij}, x_{ji} and a_i have been eliminated.

Arcs

- All the arcs x_{ij} for which:
$$t_1^i + r_i + d_{ij} > t_2^i - r_j \tag{13}$$
have been eliminated because they are not compatible with the time windows constraints.

The preprocessing of the nodes was done immediately after the generation of the data (with the gen.c program), while the preprocessing of the arcs was done later. As expected this phase had more effects on instances where the total duration of the tour is $T = 300$, for which the percentage of deleted node is 28%, while for $T = 480$ is only 12%. There is no big difference between instances with the starting point in the origin and random (about 25% of deleted nodes), while for the starting node in the middle the number of deleted nodes is less (only 11%).

2.2 Cmpl

After writing the model in the CMPL language it was necessary to convert data into *.cdat* format, so the c program txt2cdat.c takes as input the txt files generated before, performs the preprocessing of the arcs and writes the corresponding .cdat file.

In the following are shown separately the results of CBC and CPLEX on the same instances.

2.2.1 Cbc

Instances were generated with 10, 15 and 20 PoIs but "Nodes" and "Arcs" in the table are referred to the number of nodes and arcs after the preprocessing phase. For simplicity instances are numbered (Id).

Id	Nodes	Arcs	Best Integer	Best Bound	Time [s]	Gap
3	8	53	416	416	0.59	/
6	8	55	172	172	13.16	/
1	9	68	339	339	0.04	/
2	10	84	346	346	16.18	/
7	10	90	217	217	213.56	/
4	11	108	433	433	180.66	/
5	11	89	441	441	7.73	/
13	11	110	404	445	/	0.09
9	12	118	518	518	125.34	/
15	12	117	422	422	89.75	/
11	13	149	586	730	/	0.20
10	14	170	467	674	/	0.31
12	14	172	405	633	/	0.36
8	16	223	490	656	/	0.25
14	16	215	697	919	/	0.24
18	18	247	449	876	/	0.49
16	19	330	252	890	/	0.72
17	19	325	592	1051	/	0.44

Table 1: CBC: default parameters

Even if the parameters of the data change (total time and starting point), results are more or less similar; on the other hand there are big differences between instances with the same number of nodes but obtained from different number of generated PoIs; for example instances 2 and 7 have both 10 nodes and a similar number of arcs, but the first one was obtained from the generation of 10 PoIs and the second one from 15. Another example is instance 13 obtained from 20 PoIs, that CBC was not able to solve even if it has only 11 nodes.

In all cases it's clear that CBC is not able to solve instances with more than about 12 nodes. Let's consider for example instance 11, from the log:

After 0 nodes, 1 on tree, -374 best solution, best possible -730 (0.25 seconds)

⋮

After 94000 nodes, 2335 on tree, -586 best solution, best possible -730 (156.01 seconds)

⋮

After 237000 nodes, 4305 on tree, -586 best solution, best possible -730 (298.60 seconds)
Exiting on maximum time

At the beginning the continuous relaxation provides an upper-bound of 730, and, looking at the data, it is the trivial bound obtained summing up all the preferences; but the real problem is that it remained the same until the end. The same happened also for all the other unsolved instances; this probably means that CBC in most cases found the optimal solution but was not able to prove it. For this reason, in order to get better bounds, we tried to change the parameters related to the cuts but there were no improvements.

2.2.2 Cplex

The results on the same instances obtained by Cplex are:

Id	Nodes	Arcs	Best Integer	Best Bound	Time [s]	Gap
3	8	53	416	416	0.35	/
6	8	55	172	172	0.44	/
1	9	68	339	339	0.20	/
2	10	84	346	346	0.99	/
7	10	90	217	217	13.83	/
4	11	108	433	433	7.95	/
5	11	89	441	441	0.65	/
13	11	110	404	404	11.76	/
9	12	118	518	518	4.00	/
15	12	117	422	422	1.15	/
11	13	149	652	652	59.97	/
10	14	170	509	612	/	0.17
12	14	172	487	633	/	0.23
8	16	223	584	656	/	0.11
14	16	215	725	919	/	0.21
18	18	247	566	876	/	0.35
16	19	330	602	890	/	0.32
17	19	325	675	1051	/	0.36

Table 2: Cplex: default parameters

Even if Cplex showed better results with respect to CBC, also in this case the maximum size still solvable is no more than 13 nodes and, apart from instance 10 for which the bound is a little bit better, in all the other cases the bound remains the trivial one and the parameters of the solvers were not enough to improve it.

3 Reformulation of the problem

Since the previous formulation showed bad results in terms of continuous relaxation bounds, in this section it is proposed another model which uses flow variables and avoids the use of the MTZ constraints.

3.1 Model

The parameters and the x_{ij} variables are the same as for the previous model; in addition there are the following flow variables:

- y_{ij}: represents the amount of time still available to complete the tour after traversing arc $(i,j) \in A$. If the node (i,j) in not traversed then $y_{ij} = 0$.

The model can be written as:

$$\max \sum_{(i,j)\in A} x_{ij} q_j \tag{14}$$

$$\begin{aligned}
&\sum_{(0,j)\in FS(0)} x_{0j} = 1 && && (15)\\
&\sum_{(i,j)\in FS(i)} x_{ij} - \sum_{(j,i)\in BS(i)} x_{ji} = 0 && \forall i \in N && (16)\\
&\sum_{(i,j)\in FS(i)} x_{ij} \le 1 && \forall i \in N && (17)\\
&\sum_{(0,j)\in FS(0)} y_{0j} \le T - \sum_{(0,j)\in FS(0)} x_{0j} d_{0j} && && (18)\\
&\sum_{(i,j)\in FS(i)} y_{ij} - \sum_{(j,i)\in BS(i)} y_{ji} \le - \sum_{(i,j)\in FS(i)} x_{ij}(r_i + d_{ij}) && \forall i \in N \setminus \{0\} && (19)\\
&x_{ij}(t_1^i + r_i) \le x_{ij}(t_{in} + T - d_{ij}) - y_{ij} \le x_{ij} t_2^i && \forall (i,j) \in A && (20)\\
&y_{ij} \le (T - d_{ij}) x_{ij} && \forall (i,j) \in A && (21)\\
&x_{ij} \in \{0,1\} && \forall (i,j) \in A && (22)
\end{aligned}$$

where the first three sets of constraints are the same as before.

Constraints (18), (19) and (21) guarantee no subtours, in fact node 0 is the only "source" of time, therefore for all the nodes i that don't belong to a tour coming from 0, the only way to satisfy (19) is with all $y_{ij} = x_{ij} = 0$.

(20) are the time windows constraints, in fact $T - y_{ij}$ is the elapsed time from the beginning of the tour immediately after traversing (i,j), so:

$$t_{in} + T - y_{ij} - d_{ij} \tag{23}$$

is the absolute time when tourists start to move from node i, therefore this must be within the time windows of i if i is visited, otherwise y_{ij} is 0 and (20) is satisfied because $x_{ij} = 0$.

Finally (21) are the linking constraints between y_{ij} and x_{ij} variables and $T - d_{ij}$ is put there because is the smallest upper-bound for y_{ij}.

3.2 Results

Also this model was written in the cmpl language and the same phase of preprocessing was done; the results of Cbc and Cplex are shown in Table 3.

	CPLEX			CBC		
Id	**Best Integer**	**Best Bound**	**Time [s] / Gap**	**Best Integer**	**Best Bound**	**Time [s] / Gap**
13	404	404	0.44	404	404	2.12
9	518	518	0.94	518	518	6.32
15	422	422	0.84	422	422	5.56
11	652	652	0.60	652	652	2.71
10	509	509	8.52	509	509	77.83
12	487	487	1.14	487	487	6.67
8	584	584	4.52	584	584	15.05
14	725	725	4.00	725	725	81.81
18	566	566	5.12	566	566	34.06
16	623	623	1.11	623	623	6.11
17	750	750	7.40	750	750	40.12

Table 3: Model 2 with CPLEX and CBC: default parameters

Looking at the previous results, we can see that from instance 10 to instance 18, optimal solutions had been obtained even with the previous model (with Cplex), but as said before without a good bound to prove it, while, with this new formulation, both Cplex and Cbc are able to terminate in very short time for all the generated instances.

But the advantages of the second model are not only related to the better bounds but also because better integer solutions are found faster by heuristics. Considering for example instance 12, from the logs of CBC for both models we have:

```
Continuous objective value is 633 - 0.00 seconds
Cgl0004I processed model has 184 rows, 185 columns (172 integer (172 of which binary)) and 1988 elements
Cbc0038I Solution found of -92
Cbc0038I Relaxing continuous gives -92
Cbc0038I Solution found of -174
Cbc0038I Relaxing continuous gives -174
Cbc0012I Integer solution of -212 found by RINS after 3229 iterations and 41 nodes (0.83 seconds)
Cbc0012I Integer solution of -241 found by RINS after 4865 iterations and 95 nodes (1.12 seconds)
Cbc0012I Integer solution of -274 found by RINS after 16052 iterations and 400 nodes (1.94 seconds)
Cbc0016I Integer solution of -307 found by strong branching after 1444282 iterations and 48255 node
s (181.12 seconds)
Cbc0012I Integer solution of -337 found by RINS after 1763190 iterations and 57600 nodes (193.62 seconds)
Cbc0012I Integer solution of -368 found by rounding after 4469261 iterations and 135856 nodes (275.14 seconds)
Cbc0010I After 136000 nodes, 4651 on tree, -368 best solution, best possible -633 (275.23 seconds)
Cbc0012I Integer solution of -405 found by rounding after 4472585 iterations and 136012 nodes (275.25 seconds)
```

Figure 1: CBC log: model 1 - instance 12

```
Continuous objective value is 591.839 - 0.01 seconds
Cgl0004I processed model has 386 rows, 344 columns (172 integer (172 of which binary)) and 1707 elements
Cbc0038I Solution found of -241
Cbc0038I Relaxing continuous gives -241
Cbc0038I Solution found of -359
Cbc0038I Relaxing continuous gives -359
Cbc0038I Solution found of -487
Cbc0038I Relaxing continuous gives -487
Cbc0012I Integer solution of -487 found by feasibility pump after 0 iterations and 0 nodes (0.77 seconds)
```

Figure 2: CBC log: model 2 - instance 12

Despite the fact that the second model has a larger number or rows and columns, the nonzero elements after the preprocessing done by the solver are even less; with the second model the optimal solution is found even before starting the branching (in general in almost all cases good solutions are found quickly); on the other hand the heuristics applied to the first model provide solutions with values of the objective function that grow slowly.

Regarding CPLEX, the main difference between the first and the second model is the fact that for the second one the number of applied cuts is always less, but they effectively change the bound.

In order to test this formulation, bigger instances with 30 and 40 PoIs were generated:

Id	19	20	21	22	23	24	25	26	27	28	29	30
Nodes	20	29	23	28	29	29	28	36	25	35	40	35
Arcs	339	746	480	689	613	719	701	1161	576	1122	1362	1110

Table 4: Larger instances

and the results are the following:

	CPLEX			**CBC**		
Id	**Best Integer**	**Best Bound**	**Time [s] / Gap**	**Best Integer**	**Best Bound**	**Time [s] / Gap**
19	561	561	7.86	561	561	68.70
21	635	635	19.30	635	635	138.45
27	712	712	87.12	675	799	0.16
23	804	804	146.86	759	964	0.21
25	726	763	0.05	681	966	0.30
22	678	678	172.06	649	899	0.28
20	634	671	0.05	631	857	0.26
24	769	769	59.02	769	890	0.14
28	737	737	58.56	737	948	0.22
30	972	1070	0.09	/	1221	/
26	1019	1145	0.11	889	1273	0.30
29	850	1050	0.19	765	1275	0.40

Table 5: Model 2 with CPLEX and CBC: default parameters

In almost all cases Cplex found optimal solutions, or at least solutions with small gap, in the time limit, while Cbc was not able for more than 25 nodes, and for instance 30 was not found even a feasible solution.

In conclusion we can say that the second model, even if it is larger in terms of number of variables and constraints, is probably a better formulation of the same problem and therefore it shows better results which could possibly be improved with the right choice of the algorithmic parameters of the solver.