COmBinatorial optimisation and Related Algorithms

Practical Exercise 2019-2020

Integer Linear Programming Formulations for the Elementary Longest/Shortest Path Problems (ELPP/ESPP)

The classical ELPP/ESPP states as follows:[1]

ELPP/ESPP : Given a directed graph $G = (V, E, \lambda)$ where $\lambda_e \geq 0$ denotes the length of edge $e \in E$, find an elementary path with maximum/minimum total lengthfootnote[2].

Firstly, you will study the below described formulations for solving this problem. Follow carefully the list of tasks associated with any of the below models and report your observations.

1 Modeling by adding initial and final vertices s and t

Denote by $\delta^+(v) \subset E$ and $\delta^-(v) \subset E$ the set of outgoing and incoming edges of node v. Denote $G' = (V', E', \lambda)$ where

- $V' = V \cup \{s, t\}$ and s and t are such that $s \notin V$ and $t \notin V$.
- $E' = E \cup \{(s, v) | v \in V\} \cup \{(u, t) | u \in V\}$

We assume that $|\delta^-(s)| = |\delta^+(t)| = 0$.

The standard integer programming formulation to find a longest/shortest path from vertex s to t is the following:

$$\forall e \in E',\ x_e \in \{0, 1\} \tag{1}$$

$$\forall u \in V, \sum_{e \in \delta^+(u)} x_e \leq 1 \tag{2}$$

$$\forall u \in V' : \sum_{e \in \delta^+(u)} x_e - \sum_{e \in \delta^-(u)} x_e = \begin{cases} 1 & \text{if u} = s \\ \text{-1} & \text{if u} = t \\ 0 & \text{else} \end{cases} \tag{3}$$

$$\max(\min) \sum_{e \in E} x_e \lambda_e \tag{4}$$

Tasks :

1.1 Use AMPL to describe the model (1)-(4).

[1] An elementary (also called simple) is a path that traverses each vertex at most once.

[2] For the state of the art refer to ([1, 3, 5])

1.2 Run your model on the graph **G1** from the set of benchmarks and verify the correctness of the obtained results.[3]

1.3 Replace (1) with its relaxed version, i.e. with

$$\forall e \in E',\ 0 \leq x_e \leq 1 \tag{5}$$

1.4 Run again the model (2)-(5) on the graph **G1** and analyse the obtained result. What is the impact of replacing (1) by (5)? Give a theoretical explanation of your observation.

1.5 Run now the model (1)-(4) on the graph **G2** from the set of benchmarks and interpret the obtained result.

1.1 Sequential formulation (MTZ)

To derive an extended formulation à la Miller, Tucker and Zemlin ([4]) (hereafter MTZ) we introduce, for each vertex $\forall v \in V$, an auxiliary variable y_v and add the constraint (7)

$$\forall v \in V,\ y_v \geq 0 \tag{6}$$

$$\forall (u,v) \in E, (y_v - y_u) \geq x_{uv} - (1 - x_{uv})|V| \tag{7}$$

Tasks :

2.1 Use AMPL to describe the model (1)-(7). Denote it by MTZ.

2.2 Run the model MTZ on the graph **G2** and verify the correctness of the obtained result. Give a theoretical explanation of your observation.

2.3 Run now the model MTZ on the graphs **fast-10, ladder-8, wheel-15** and verify the correctness of the obtained result.

2.4 Compute theoretically the number of the variables (either binary, integer, or real), the number of equalities and the number of inequalities in the model (MTZ). Chose an instance and check if the data provided by AMPL corroborates your estimation.

1.2 Single-flow formulation (SF)

A formulation similar to the single-flow Asymmetric Traveling Salesman Problem (ATSP) formulation of Gavish and Graves ([3]) can be obtained introducing an auxiliary flow $f_e \in R^+$ to be delivered to the nodes belonging to the $s - t$ path.

$$\forall e \in E' : 0 \leq f_e \leq |V|. \tag{8}$$

The variable f_e expresses the quantity of the flow circulating along the edge $e \in E$. In the case $x_e = 0$, no flow can use this edge, as ensured by

[3] Many thanks to Sebastien François who generated some of the instances for this practical!

$$f_e \leq |V| x_e \quad \forall e \in E'. \tag{9}$$

Furthermore, to any vertex $v \in V$ a binary variable i_v is associated.

$$\forall v \in V,\ i_v \in \{0,1\} \tag{10}$$

According to (11), the flow exiting s equals the number of nodes that are reached by the $s-t$ path. Constraints (12) ensure that the balance of the auxiliary flow on each node is equivalent to i_v, which, according to constraint (13), is either 1, if node v is in the $s-t$ path, or 0 otherwise. Constraints (14) and (15) ensure that the path begins with s and ends with t.

$$\sum_{e \in \delta^+(s)} f_e = |V| \tag{11}$$

$$\forall v \in V: \sum_{e \in \delta^-(v)} f_e - \sum_{e \in \delta^+(v)} f_e = i_v \tag{12}$$

$$\forall v \in V: \sum_{e \in \delta^-(v)} x_e = i_v \tag{13}$$

$$\sum_{e \in \delta^+(s)} x_e = 1 \tag{14}$$

$$\sum_{e \in \delta^-(t)} x_e = 1 \tag{15}$$

$$\max(\min) \sum_{e \in E} x_e \lambda_e \tag{16}$$

Tasks :

3.1 Use AMPL to describe the model obtained by adding constraints (8)-(16) and denote it by (SF).

3.2 Run the model SF on the graph **G2** and verify the correctness of the obtained result.

3.3 Run now the model SF on the graphs **fast-10, ladder-8, wheel-15** and verify the correctness of the obtained result.

3.4 Compute theoretically the number of the variables (either binary, integer, or real), the number of equalities and the number of inequalities in the model (SF). Chose an instance and check if the data provided by AMPL corroborates your estimation.

3.5 Compare the performance of the model (MTZ) with the one of (SF) on larger instances from the benchmarks data sets and/or using your own graph instances generator[4]. For this purpose you should use the number of iterations, the number of branch and bounds nodes and the time provided by the solver. Check if this statistic changes when using cplex instead of gurobi.

[4]Many thanks to Kerian Thuillier who provided a graph instances generator!

2 Direct modelling (GAT model [2])

This model uses the original graph $G = (V, E)$ without adding new vertices. As in the previous models, a binary variable is associated to any edge of the graph, i.e.

$$\forall e \in E : x_e \in \{0, 1\} \tag{17}$$

Furthermore, to any vertex $v \in V$ we associate three variables, i_v, s_v, and t_v, which stand respectively for intermediate, source, and target vertex for some path, and satisfy

$$0 \leq i_v \leq 1, \quad 0 \leq s_v \leq 1, \ 0 \leq t_v \leq 1. \tag{18}$$

Each vertex can be visited at most once, i.e.

$$\forall v \in V : i_v + s_v + t_v \leq 1. \tag{19}$$

All three variables are set to zero when the associated vertex v is outside the path. Otherwise, it could be either a source/initial vertex ($s_v = 1, t_v = 0, i_v = 0$), or a target/final ($t_v = 1, s_v = 0, i_v = 0$), or an intermediate vertex ($i_v = 1, t_v = 0, s_v = 0$). These four possibles states for a vertex v are determined by the following constraints

$$s_v + i_v = \sum_{e \in \delta^+(v)} x_e \tag{20}$$

and

$$t_v + i_v = \sum_{e \in \delta^-(v)} x_e. \tag{21}$$

Finally, only one path is searched for

$$\sum_{v \in V} s_v = 1 \text{ and } \sum_{v \in V} t_v = 1. \tag{22}$$

Theorem 1. *The variables $i_v, s_v, t_v, \forall v \in V$ take binary values.*

Proof. See [2]. □

We introduce a continuous variable $f_e \in R^+$ to express the quantity of the flow circulating along the edge $e \in E$.

$$\forall e \in E : 0 \leq f_e \leq |V|. \tag{23}$$

$\forall e \in E$, the value of x_e is set to 1, if the edge e is part of the path. Otherwise, $x_e = 0$ and no flow can use this edge, as ensured by

$$f_e \leq |V| x_e \quad \forall e \in E. \tag{24}$$

The balance of the auxiliary flow on each node is provided by the following constraints

$$\forall v \in V : \sum_{e \in \delta^+(v)} f_e - \sum_{e \in \delta^-(v)} f_e \geq i_v + s_v - t_v |V| \tag{25}$$

$$s_v + i_v|V| \geq \sum_{e \in \delta^+(v)} f_e. \tag{26}$$

The purpose of the last two constraints is manifold. When a vertex v is a source ($s_v = 1$), (25) and (26) generate an initial flow of value 1. When v is an intermediate vertex ($i_v = 1$), constraint (25) forces the flow exiting v to increase in respect to the flow that inputs v. This feature forbids cycles in the context of (20) and (21). When v is a final vertex, (25) is simply a valid inequality.

We search for the longest path in the graph. The objective hence is :

$$\max \sum_{e \in E} x_e \lambda_e \tag{27}$$

Tasks :

4.1 Use AMPL to describe the model obtained by adding constraints (17)-(27) and denote it by GAT from Genscale Assembly Tool.

4.2 Run the model GAT on the graphs **G2, fast-10, ladder-8, wheel-15** and verify the correctness of the obtained result.

4.3 Compute theoretically the number of the variables (either binary, integer, or real), the number of equalities and the number of inequalities in the model GAT. Chose an instance and check if the data provided by AMPL corroborates your estimation.

4.4 Verify experimentally Theorem 1

4.5 Compare the performance of above three models on larger instances from the benchmarks data sets and/or using your own graph instances generator.

References

[1] Bui, Q.T., Deville, Y., Pham, Q.D.: Exact methods for solving the elementary shortest and longest path problems. Annals of Operations Research 244(2), 313–348 (2016), `http://dx.doi.org/10.1007/s10479-016-2116-5`

[2] François, S., Andonov, R., Lavenier, D., Djidjev, H.: Global optimization for scaffolding and completing genome assemblies. Electronic Notes in Discrete Mathematics 64, 185–194 (2018), `https://doi.org/10.1016/j.endm.2018.01.020`

[3] Gavish, B., Graves, S.C.: The travelling salesman problem and related problems. Tech. Rep. GR-078-78, Massachusetts Institute of Technology. (1978)

[4] Miller, C.E., Tucker, A.W., Zemlin, R.A.: Integer programming formulation of traveling salesman problems. J. ACM 7(4), 326–329 (Oct 1960), http://doi.acm.org/10.1145/321043.321046

[5] Taccari, L.: Integer programming formulations for the elementary shortest path problem. European Journal of Operational Research 252(1), 122–130 (2016), http://dx.doi.org/10.1016/j.ejor.2016.01.003