Vienna University of Technology

Mathematical Programming

Programming Exercise (Part 1)

Alexander Ponticello BSc. (01226441)
Jonas Ferdigg BSc. (01226597)

April 30, 2018

For our compact models we chose single-commodity flows and multi-commodity flows. We wrote a testbench to run the different graph instances automatically. If you want to invoke it, you can do so by calling *kmst* with the following parameters:

Calling the first 10 test instances for SCF (2 per .dat file)

```
./kmst -m 0 -t test/test.in -n 10
```

Calling the first 10 test instances for MCF (2 per .dat file)

```
./kmst -m 1 -t test/test.in -n 10
```

4 Single-Commodity Flows

4.1 Formulation

We added an artificial root node v_0 to our graph and edges $(v0, i) : i \in V$ from the root node to every other node with weight 0:

$$V' = V \cup \{v_0\}$$

$$E' = E \cup \{(v_0, i) : i \in V\}$$

$$\forall i \in V : w_{v_0 i} = 0$$

4.2 Variables

- x_e : Edge decision variable
- y_{ij} : Arc decision variable
- f_{ij} : Flow variable

4.3 Constraints

$$\forall e = \{i, j\} \in V' : x_e = y_{ij} + y_{ji} \tag{1}$$

$$\sum_{j, j \neq v_0} f_{v_0 j} = k \tag{2}$$

$$\sum_{j, j \neq v_0} x_{(v_0, j)} = 1$$

$$\forall j \in V : 0 \leq \sum_{i \in V, i \neq j} f_{ij} - \sum_{i \in V, i \neq j} f_{ji} \leq 1 \tag{3}$$

$$\sum_{j \in V} (\sum_{i \in V, i \neq j} f_{ij} - \sum_{i \in V, i \neq j} f_{ji}) = k \tag{4}$$

$$\forall (i, j) \in E' : 0 \leq f_{ij} \leq y_{ij} * k \tag{5}$$

$$\forall (i, j) \in E' : y_{ij} \in \{0, 1\} \tag{6}$$

- (1) : Establishes the connection between arc decision variable y_{ij} and edge decision variable x_e
- (2) : Guarantees that exactly k tokens are being produced by the artificial root node v_0
- (3) : Guarantees that only one edge between artificial root node v_0 and any other node is selected
- (4) : Different from the ordinary MST, in the kMST not every node has to consume a token. With 4 we define that every node may consume up to one token or none.
- (5) : This constraint guarantees that tokens can only flow on selected arcs
- (6) : This is the domain constraint for the binary arc decision variable y_ij

4.4 Results

- *instance* : Graph instance file
- *k : Number* of nodes to be selected in the kMST
- *exp.optimum* : Expected optimum taken from the slides
- *obj.value* : Objective value of the
- *w8.sum* : Sum of weights of all the selected edges (obtained by iterating over all the edges in the solution after solving the problem and adding their weights)
- *cpu.time* : Running time
- *bnb.nodes* : Number of Branch-and-Bound nodes generated while solving the problem

instance	k	exp.optimum	obj.value	w8.sum	cpu.time	bnb.nodes
g01.dat	2	46	46	46	0	0
g01.dat	5	477	477	477	0.03	23
g02.dat	4	373	373	373	0.23	31
g02.dat	10	1390	1390	1390	1.81	1621
g03.dat	10	725	725	725	3.71	970
g03.dat	25	3074	3074	3074	13.3	749
g04.dat	14	909	909	909	20.81	1375
g04.dat	35	3292	3291	3292	55.32	5620
g05.dat	20	1235	1235	1235	76.38	3114
g05.dat	50	4898	4897	4898	179.57	15430

Note: For some reason the *obj.value* sometimes differs from the *w8.sum* by being 1 off. This also depends on the system the program is running on. We assume this is due to numeric errors. CPLEX does not terminate for instances *g06*, *g07* and *g08*.

5 Multi-Commodity Flows

5.1 Formulation

We added an artificial root node v_0 to our graph and edges $(v0, i) : i \in V$ from the root node to every other node with weight 0:

$$V' = V \cup \{v_0\}$$
$$E' = E \cup \{(v_0, i) : i \in V\}$$
$$\forall i \in V : w_{v_0 i} = 0$$

5.2 Variables

- x_e : Edge decision variable
- y_{ij} : Arc decision variable
- f_{ij} : Flow variable

5.3 Constraints

$$\forall e = \{i, j\} \in V' : x_e = y_{ij} + y_{ji} \tag{7}$$

$$\forall m \in V : \sum_{j \in V, j \neq v_0} f^m_{v_0 j} = 1 \tag{8}$$

$$\sum_{m \in V} \sum_{i \in V, i \neq m} f^m_{im} = k$$

$$\forall j, m \in V, j \neq m : \sum_{i \in V, i \neq j} f^m_{ij} - \sum_{i \in V, i \neq j} f^m_{ji} \tag{9}$$

$$\forall (i, j) \in E' : 0 \leq f^m_{ij} \leq y_{ij} \tag{10}$$

$$\forall (i, j) \in E' : y_{ij} \in \{0, 1\} \tag{11}$$

5.4 Results

For an explanation of the different columns, see section 4.4.

instance	k	exp.optimum	obj.value	w8.sum	cpu.time	bnb.nodes
g01.dat	2	46	46	46	0.04	0
g01.dat	5	477	477	477	0.14	0
g02.dat	4	373	373	373	0.57	389
g02.dat	10	1390	1390	1390	1.08	0
g03.dat	10	725	725	725	16.33	1039
g03.dat	25	3074	3074	3074	31.22	0
g04.dat	14	909	909	909	251.16	779
g04.dat	35	3292	3292	3292	407.32	153
g05.dat	20	1235	1234	1235	847.35	618
g05.dat	50	4898	4898	4898	1263.32	0

Note: Same as with the SCF *obj.value* sometimes differs from the *w8.sum* by being 1 off. For some instances, $getNnodes()$ seems to return a (most likely) wrong 0 as number of Branch-and-Bound nodes. CPLEX does not terminate for instances *g06*, *g07* and *g08*.

6 Interpretation

We invested quite some time in this programming exercise and so we were very disappointed that instances $g06-g08$ did not terminate for both SCF and MCF. The overall performance of the solutions does not seem good either. Since we didn't have any experience with CPLEX, we could not tell if this behaviour was normal or not.