Algorytmy Optymalizacji Dyskretnej 2022/2023

Laboratorium 2

Piotr Kocia

15th April 2023

Contents

1 Supplying Airports

The model is as follows

$$
\begin{aligned}
\text{minimise} \quad & \sum_{(c,a,p)\in P} x_{c,a} \cdot p \\
\text{subject to } \forall\,(a,d) \in D : \quad & \sum_{c\in\{1..p_c\}} x_{c,a} \ \geq d \\
\forall\,(c,s) \in S : \quad & \sum_{a\in\{1..p_a\}} x_{c,a} \ \leq s
\end{aligned}
$$

where P is a set of 3-tuples containing company index, airport index and the price of the fuel delivered from a company to an airport, D is a set of 2-tuples containing airport index and its fuel demand, S is a set of 2-tuples containing company index and its maximum fuel supply, p_c is the parameter for the number of companies, p_a is the parameter for the number of airports.

The objective function, which is the total cost of the minimum amount of fuel required by all airports, has a minimum value of 8525000. All companies supply fuel, however, not all supply has been exhausted. Constraints, values and bounds are presented in table 1.

Table 1: Constraints, their values and bounds.

Constraint	Value	Lower bound	Upper bound
demand_constraint[1,110000]	110000	110000	
demand_constraint[2,220000]	220000	220000	
demand_constraint[3,330000]	330000	330000	
demand_constraint[4,440000]	440000	440000	
supply_constraint[1,275000]	275000		275000
supply_constraint[2,550000]	165000		550000
supply_constraint[3,660000]	660000		660000
cost	8525000		

Table 2: Supply values.

Supply (company -> airport)	Value	Lower bound
supply[1,1]	0	0
supply[2,1]	110000	0
supply[3,1]	0	0
supply[1,2]	165000	0
supply[2,2]	55000	0
supply[3,2]	0	0
supply[1,3]	0	0
supply[2,3]	0	0
supply[3,3]	330000	0
supply[1,4]	110000	0
supply[2,4]	0	0
supply[3,4]	330000	0

2 Shortest Path

The shortest path problem is a special case of the maximum flow problem where the source has an outflow of 1 (and no inflow) and the destination (sink) has an inflow of 1 (and no outflow). In this case we also have an additional constraint of maximum time.

$$
\begin{aligned}
\text{minimise} \quad & \sum_{(i,j,c)\in C} c \cdot e_{i,j} & \\
\text{subject to} \quad & \sum_{(i,dst)\in E} e_{i,dst} - \sum_{(dst,i)\in E} e_{dst,i} = 1; & \text{destination constraint} \\
& \sum_{(src,i)\in E} e_{src,i} - \sum_{(i,src)\in E} e_{i,src} = 1; & \text{source constraint} \\
& \forall\, u \neq src, dst : \sum_{(u,v)\in E} e_{u,v} = \sum_{(v,u)\in E} e_{v,u} & \text{inflow} = \text{outflow} \\
& \sum_{(i,j,t)\in T} e_{i,j} \cdot t \;\leq T\_max & \text{time constraint}
\end{aligned}
$$

where E is the set of all edges in the graph, C is an attribute set of 3-tuples assigning traversal cost to edges, T is an attribute set of 3-tuples assigning time cost to edges, $e_{u,v}$ is an edge indicator (binary variable), src and dst are respectively source and destination indice and T_max is the maximum traversal time.

The graph used is an undirected dodecahedron (20 vertices, 60 directed edges) with random traversal costs and inversely proportional time costs. The path searched is from vertex 1 to vertex 16. The objective function has a minimum cost of 157 with the time constraint being 218. The solution is viable with and without the time constraint due to its immense value in comparison to the value of the constraint. The solver seems to be capable of finding the optimal path through the graph regardless of the problem being limited to ILP or not.

Table 3: Constraints, their values and bounds.

Constraint	Value	Upper bound
path	157	

Constraint	Value	Upper bound
time_constraint	218	1500

Table 4: Edge indicators for the shortest path from vertex 1 to vertex 16.

No.	Edge	Activity	No.	Edge	Activity	No.	Edge	Activity
1	1 -> 3	0	21	2 -> 7	0	41	10 -> 12	0
2	3 -> 1	0	22	7 -> 2	0	42	12 -> 10	0
3	1 -> 4	1	23	2 -> 8	0	43	10 -> 17	0
4	4 -> 1	0	24	8 -> 2	0	44	17 -> 10	0
5	1 -> 5	0	25	6 -> 15	0	45	11 -> 20	0
6	5 -> 1	0	26	15 -> 6	0	46	20 -> 11	0
7	3 -> 9	0	27	6 -> 16	0	47	13 -> 14	1
8	9 -> 3	0	28	16 -> 6	0	48	14 -> 13	0
9	3 -> 10	0	29	7 -> 17	0	49	13 -> 15	0
10	10 -> 3	0	30	17 -> 7	0	50	15 -> 13	0
11	4 -> 11	0	31	7 -> 19	0	51	12 -> 19	0
12	11 -> 4	0	32	19 -> 7	0	52	19 -> 12	0
13	4 -> 13	1	33	8 -> 18	0	53	14 -> 16	1
14	13 -> 4	0	34	18 -> 8	0	54	16 -> 14	0
15	5 -> 12	0	35	8 -> 20	0	55	15 -> 20	0
16	12 -> 5	0	36	20 -> 8	0	56	20 -> 15	0
17	5 -> 14	0	37	9 -> 11	0	57	16 -> 19	0
18	14 -> 5	0	38	11 -> 9	0	58	19 -> 16	0
19	2 -> 6	0	39	9 -> 18	0	59	17 -> 18	0
20	6 -> 2	0	40	18 -> 9	0	60	18 -> 17	0

3 Minimum Police Cars on Duty

A trivial model to minimise the number of cars with several constraints

$$
\begin{aligned}
\text{minimise} \quad & \sum_{(i,j)\in\{1..p_s\}\times\{1..p_d\}} c_{i,j} \\
\text{subject to} \; & \forall\,(s,d,v) \in C_{min}:\; c_{s,d} \geq v \\
& \forall\,(s,d,v) \in C_{max}:\; c_{s,d} \leq v \\
& \forall\,(s,v) \in S:\; \sum_{d\in\{1..p_d\}} c_{s,d} \geq v \\
& \forall\,(d,v) \in D:\; \sum_{s\in\{1..p_s\}} c_{s,d} \geq v
\end{aligned}
$$

where C_{min} is the set of 3-tuples minimum cars, C_{max} is the set of 3-tuples maximum cars, S minimum per shift, D minimum per district.

The minimum number of cars is 48. The number of cars per district per shift is presented in table 6.

Table 5: Constraints, their values and bounds.

Constraint	Activity	Lower bound	Upper bound
cars_on_duty	48		
overall_minimum[1,1,2]	2	2	
overall_minimum[2,1,4]	7	4	
overall_minimum[3,1,3]	7	3	
overall_minimum[1,2,3]	3	3	
overall_minimum[2,2,6]	6	6	
overall_minimum[3,2,5]	5	5	
overall_minimum[1,3,5]	5	5	
overall_minimum[2,3,7]	7	7	
overall_minimum[3,3,6]	6	6	
overall_maximum[1,1,3]	2		3
overall_maximum[2,1,7]	7		7
overall_maximum[3,1,5]	5		5

Constraint	Activity	Lower bound	Upper bound
overall_maximum[1,2,5]	3		5
overall_maximum[2,2,7]	6		7
overall_maximum[3,2,10]	7		10
overall_maximum[1,3,8]	5		8
overall_maximum[2,3,12]	7		12
overall_maximum[3,3,10]	6		10
shift_minimum[1,10]	10	10	
shift_minimum[2,20]	20	20	
shift_minimum[3,18]	18	18	
district_minimum[1,10]	16	10	
district_minimum[2,14]	14	14	
district_minimum[3,13]	18	13	

Table 6: Cars per district per shift.

Shift	District	Cars
1	1	2
1	2	3
1	3	5
2	1	7
2	2	6
2	3	7
3	1	7
3	2	5
3	3	6

4 Cameras and Containers

The model involves many binary variables to form a two dimensional grid, each one being an indicator of a camera in the corresponding tile. We require at least one camera to be within k units of every container and all tiles with containers to not have any cameras.

$$
\begin{aligned}
\text{mnimise} \quad & \sum_{(x,y)\in\{1..m\}\times\{1..n\}} \text{grid}_{x,y} \\
\text{subject to } \forall\,(i,j)\in C : \quad & \sum_{(x,y)\,\in\,(\{(i-k)..(i+k)\}\times\{(j-k)..(j+k)\})\,\cap\,(\{1..m\}\times\{1..n\})} \text{grid}_{x,y} >= 1 \\
\forall\,(x,y)\in C : \quad & \text{grid}_{x,y} = 0
\end{aligned}
$$

where C is the set of 2-tuples representing positions of containers, grid are the binary variables, m and n are the grid size parameters.

We solve an example of the problem with parameters $n = 9$, $m = 9$, $k = 2$ and containers in positions (1, 2) (1, 3) (1, 4) (7, 7) (9, 9) (7, 9) (9, 7). The minimum number of cameras for the given data is 2.

Constraints are trivial in this model, hence we omit them.

Table 7: Placement of cameras in the grid.

No.	Position	Value	No.	Position	Value	No.	Position	Value
1	x: 1, y: 1	0	28	x: 4, y: 1	0	55	x: 7, y: 1	0
2	x: 1, y: 2	0	29	x: 4, y: 2	0	56	x: 7, y: 2	0
3	x: 1, y: 3	0	30	x: 4, y: 3	0	57	x: 7, y: 3	0
4	x: 1, y: 4	0	31	x: 4, y: 4	0	58	x: 7, y: 4	0
5	x: 1, y: 5	0	32	x: 4, y: 5	0	59	x: 7, y: 5	0
6	x: 1, y: 6	0	33	x: 4, y: 6	0	60	x: 7, y: 6	0
7	x: 1, y: 7	0	34	x: 4, y: 7	0	61	x: 7, y: 7	0
8	x: 1, y: 8	0	35	x: 4, y: 8	0	62	x: 7, y: 8	1
9	x: 1, y: 9	0	36	x: 4, y: 9	0	63	x: 7, y: 9	0

No.	Position	Value	No.	Position	Value	No.	Position	Value
10	x: 2, y: 1	0	37	x: 5, y: 1	0	64	x: 8, y: 1	0
11	x: 2, y: 2	1	38	x: 5, y: 2	0	65	x: 8, y: 2	0
12	x: 2, y: 3	0	39	x: 5, y: 3	0	66	x: 8, y: 3	0
13	x: 2, y: 4	0	40	x: 5, y: 4	0	67	x: 8, y: 4	0
14	x: 2, y: 5	0	41	x: 5, y: 5	0	68	x: 8, y: 5	0
15	x: 2, y: 6	0	42	x: 5, y: 6	0	69	x: 8, y: 6	0
16	x: 2, y: 7	0	43	x: 5, y: 7	0	70	x: 8, y: 7	0
17	x: 2, y: 8	0	44	x: 5, y: 8	0	71	x: 8, y: 8	0
18	x: 2, y: 9	0	45	x: 5, y: 9	0	72	x: 8, y: 9	0
19	x: 3, y: 1	0	46	x: 6, y: 1	0	73	x: 9, y: 1	0
20	x: 3, y: 2	0	47	x: 6, y: 2	0	74	x: 9, y: 2	0
21	x: 3, y: 3	0	48	x: 6, y: 3	0	75	x: 9, y: 3	0
22	x: 3, y: 4	0	49	x: 6, y: 4	0	76	x: 9, y: 4	0
23	x: 3, y: 5	0	50	x: 6, y: 5	0	77	x: 9, y: 5	0
24	x: 3, y: 6	0	51	x: 6, y: 6	0	78	x: 9, y: 6	0
25	x: 3, y: 7	0	52	x: 6, y: 7	0	79	x: 9, y: 7	0
26	x: 3, y: 8	0	53	x: 6, y: 8	0	80	x: 9, y: 8	0
27	x: 3, y: 9	0	54	x: 6, y: 9	0	81	x: 9, y: 9	0

5 Manufacturing Optimisation

The model is described by

$$\begin{aligned}
\text{maximise} \quad & \sum_{i,p,c:(i,p)\in P,(i,c)\in C} x_i \cdot (p - c - \sum_{u,m:(j,u)\in U,(i,j,m)\in M} u \cdot m) \\
\text{subject to } \forall\, j \in \{1..p_m\}: \quad & \sum_{(i,j,m)\in M} x_i \cdot m \ \leq 60 && \text{machine use time} \\
\forall\, (i,d) \in D: \quad & x_i \ \leq d && \text{maximum production}
\end{aligned}$$

where:

- P is the set of 2-tuples with a product index and the price per kilogram,
- C is the set of 2-tuples with a product index and the cost of the material required to produce a kilogram of the product,
- D is the set of 2-tuples with a product index and the demand of the product in kilograms,
- U is the set of 2-tuples with a machine index and the cost for an hour of work,
- M is the set of 3-tuples with a product index, a machine index and the time in hours to process a kilogram of a product on a machine,
- p_m is a parameter describing the number of machines.

The maximum profit is 3632.5 currency units. Product 1 is the least profitable in our case. The weekly production of each product is summarised in table 9 .

Table 8: Constraints, their values and bounds.

Constraint	Value	Upper bound
profit		3632.5
production[1,400]	125	400
production[2,100]	100	100
production[3,150]	150	150
production[4,500]	500	500
machine use time[1]	58.75	60
machine use time[2]	60	60
machine use time[3]	35	60

Table 9: Weekly production of each product.

Variable	Value	Lower bound
x[1]	125	0
x[2]	100	0
x[3]	150	0

Variable	Value	Lower bound
x[4]	500	0