École polytechnique fédérale de Lausanne

Disopt

Master semester project

Autumn Semester 2019

A combinatorial approach to the trains routing problem

Student:
Charles Dufour

Supervisors:
Prof. Friedrich Eisenbrand
Jonas Racine

Contents

1 Introduction

2 Flatland challenge

3 Literature review

3.1 Double vertex graph

3.2 MAPF

4 Modelization of the environment

4.1 Transition network

The transition network is an oriented graph $G = (V, A)$ that represents the original 2D grid world.

Figure 1: Flatland environment example. Randomly generated 7×7 grid with 3 trains.

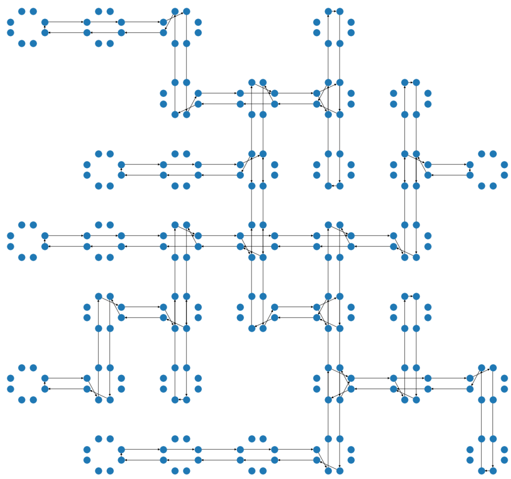

Figure 2: transition graph extracted from the flatland environment on the left.

4.2 Time expanded network

We give a definition inspired from (Skutella, 2008, p. 19):

Definition 4.1 (Time-expanded network)**.** Let $G = (V, E)$ be a network with capacities u and costs c on the arcs. For a given time horizon $T \in \mathbb{Z}_{>0}$, the corresponding *time-expanded network* $G^T = (V^T, E^T)$ with capacities and costs on the arcs is defined as follows. For each node $v \in V$ we create T copies $v_0, v_1, \ldots, v_{T-1}$, that is,

$$V^T := \{v_\theta | v \in V, \theta = 0, 1, \ldots, T-1\}.$$

For each arc $e = (v, w) \in E$, there are $T-1$ copies $e_0, e_1, \ldots, e_{T-2}$ where arc e_θ connects node v_θ to node $w_{\theta+1}$. Arc e_θ has capacity $u_{e_\theta} := u_{e_\theta}$ and cost $c_{e_\theta} := c_{e_\theta}$. Moreover, E^T contains *waiting* arcs $(v_\theta, v_{\theta+1})$ for $v \in V$ and $\theta = 0, \ldots, T-2$. The capacity of waiting arcs is 1 and they have cost 1. Summarizing, the set of arcs E^T is given by

$$\begin{aligned} E^T :=& \{e_\theta = (v_\theta, w_{\theta+1}) | e = (u, v) \in E, \theta = 0, 1, \ldots, T-2\} \\ & \cup \{(v_\theta, v_{\theta+1}) | v \in V, \theta = 0, 1, \ldots, T-2\} \end{aligned}$$

An example of a time-expanded network is given in Figure **??**. Notice that the size of the time-expanded network G^T is linear in T and therefore only pseudo-polynomial in the input size.

We then proceed to define a time step, which represent what would be added to a time expanded graph, were we to expand the horizon by one.

Definition 4.2 (Time step in time expanded network)**.** A *time step* at time θ is the set of all edges (v_θ, w) for all $v \in V$, $w \in V^T$ such that $(v_\theta, w) \in E^T$. It represents the set of all edges starting at time θ.

explain a bit more

add an example as a Figure

Remark. There are no cycles in the time expanded network since $\nexists (v_\theta, w_{\tilde{\theta}}) \in E^T$ with $\tilde{\theta} < \theta$.

4.3 Restrictions

Needs to define C_R in term of trains movements and then in term of arcs flow and explain why there only span one time step.

5 Minimum cost multicommodity flow

We now consider the time expanded network defined in section 4.2 and denote it by $G = (V, A)$. The trains will be the commodities, we suppose that we have K of them.

We now formalize the definition of restriction as described in section 4.3.

Definition 5.1. A restriction R over a time expanded network $G = (V, A)$ is a set of arcs: $R \subset A$ over which we want to impose certains specification (e.g. global capacity).

We define C_R the set of all restrictions over the time expanded network G.

We also introduce a notation we will be heavily using in this section:

Definition 5.2. Given α a set,

$$\delta_\alpha(\beta) = \begin{cases} \mathbb{1}\{\beta \cap \alpha \neq \emptyset\} \text{ if } \beta \text{ is a set} \\ \mathbb{1}\{\beta \in \alpha\} \text{ otherwise} \end{cases}$$

Informally this represents the fact that α and β are not disjoint or that β is contained in α.

5.1 Arc flow formulation

The minimum cost multicommodity flow can be formulate as an arc-flow integer program:

$$\begin{array}{lll} \text{minimize} & \displaystyle\sum_{k=1}^{K} \sum_{(i,j)\in A} x_{ij}^k & \\ \text{subject to} & \displaystyle\sum_{k=1}^{K} x_{ij}^k \leq 1, & \forall (i,j) \in A \\ & \displaystyle\sum_{k=1}^{K} \sum_{(i,j)\in R} x_{ij}^k \leq 1, & \forall R \in C_R \\ & \mathcal{N} x^k = b^k, & \forall k \in \{1, \dots, K\} \\ & x_{ij}^k \in \{0, 1\}, & \forall (i,j) \in A \end{array} \tag{5.1.1}$$

define the $\mathcal{N}^k$ matrix, b^k vector

This formulation is clear and intuitive: for each commodity we decide whether or not it will use a specific arc at a certain time by setting x_{ij}^k to 1 or to 0. But this formulation (5.1.1) is not scalable due to its high number of restrictions and its integrability (see section 6.1 for a more detailed explanation). We then proceed to find new ways to solve this problem.

5.1.1 Relaxation and approximation

We first relax (5.1.1) by allowing $x_{ij}^k \geq 0$ while letting the other restrictions untouched.

5.2 Column generation method

In this section, we give a column generation solution procedure that works with arbitrary restrictions, as long as the restrictions do not span more than one time step (see section 4.3 for an explanation). We directly consider the relaxation problem.

5.2.1 Reformulation as a path flow problem

explain what is a path flow formulation

For this section we will restate our problem using path flows instead of arc flows as before. The problem then becomes:

$$\begin{array}{lll} \text{minimize} & \displaystyle\sum_{P \in \mathcal{P}} f(P)|P| & \\ \text{subject to} & \displaystyle\sum_{P \in \mathcal{P}_R} f(P) \leq 1, & \forall R \in C_R \\ & \displaystyle\sum_{P \in P^k} f(P) = 1, & \forall k \in \{1, \dots, K\} \\ & f(P) \geq 0, & P \in \mathcal{P} \end{array} \tag{5.2.1}$$

With:

- $\mathcal{P} = \{\mathcal{P}_1, \dots, \mathcal{P}_K\}$ and $\mathcal{P}_i$ is all the possible distinct paths between s_i and t_i.
- $\mathcal{P}_R = P \in \mathcal{P} : |P \cap R| > 0$. This represents the paths that "goes trough" the restriction R.

The second restriction $\sum_{P \in P^k} f(P) = 1$ contains in our case the fact that we need to have exactly one train going from s_k to t_k for each commodity.

The dual of the primal formulation (5.2.1) is:

$$\begin{array}{lll} \text{maximize} & \displaystyle\sum_{R \in C_R} y_R + \sum_{i=1}^{K} \sigma_i & \\ \text{subject to} & \displaystyle\sum_{R \in C_R} \delta_P(R) \cdot y_R + \sigma_k \leq |P|, & \forall P \in \mathcal{P} \\ & y \leq 0, & y \in \mathbb{R}^{|C_R|} \\ & & \sigma \in \mathbb{R}^K \end{array} \tag{5.2.2}$$

With respect of the dual variables,the reduced cost $c_P^{\sigma,y}$ for each path flow variable $f(P)$ which belongs to commodity k is :

$$c_P^{\sigma,y} = |P| - \sum_{R \in C_R} \delta_P(R) \cdot y_R - \sigma_k \tag{5.2.3}$$

We can then derive complementary slackness conditions.

Theorem 5.1 (Path flow complementary slackness conditions)**.** *The commodity pah flow $f(P)$ are optimal in the path flow formulation (5.2.1) of the multicommodity flow problem if and only if for some restriction prices y_R and commodity prices σ_k, the reduced cost and arc flows satisfy the following complementary slackness conditions:*

$$y_R \left[\sum_{k \in [K]} \sum_{P \in P^k} \delta_P(R) \cdot f(P) - 1 \right] = 0 \text{ for all } R \in C_R. \tag{5.2.4}$$

$$c_P^{\sigma,y} \geq 0 \text{ for all } k \in [K] \text{ and all } P \in P^k. \tag{5.2.5}$$

$$c_P^{\sigma,y} \cdot f(P) = 0 \text{ for all } k \in [K] \text{ and all } P \in P^k. \tag{5.2.6}$$

Proof of theorem 5.1.
We show that optimality of the primal (5.2.1) implies the complementarity slackness conditions from theorem 5.1.

Denote $f^*(\mathcal{P}), y^*_S$ and σ^*_k the optimal solution from (5.2.1) and (5.2.2). Since y^* and σ^* are solution of the dual formulation we get condition (5.2.5) directly.

To show the other two conditions, we will write the primal and dual expression in matrix form.

Primal

$$\begin{aligned} &\text{minimize} && b^T f(\mathcal{P}) \\ &\text{subject to} && \begin{pmatrix} -A_{C_R} \\ A_K \\ -A_K \end{pmatrix} \cdot f(\mathcal{P}) \geq \begin{pmatrix} -1_{|C_R|} \\ 1_K \\ -1_K \end{pmatrix} \\ &&& f(\mathcal{P}) \geq 0 \end{aligned}$$

With:

- $b \in \mathbb{R}^{|\mathcal{P}|}, \quad b_P = |P|$ is the vector containing the length of the paths.
- $A_{C_R} \in \mathbb{R}^{|C_R| \times |\mathcal{P}|}$ where $(A_{C_R})_{ij}$ is 1 if the i^{th} restriction contains an edge that belongs to the j^{th} path and 0 else.
- $A_K \in \mathbb{R}^{K \times |\mathcal{P}|}$ where $(A_K)_{ij}$ is 1 if the j^{th} path belongs to $\mathcal{P}_i$ (i.e. belongs to the i^{th} commodity) and 0 else.

Dual

$$\begin{aligned} &\text{maximize} && \left(-1^T_{|C_R|}, 1^T_K, -1^T_K\right) \cdot \tilde{y} \\ &\text{subject to} && \left(-A^T_{C_R}, A^T_K, -A^T_K\right) \cdot \tilde{y} \leq b \\ &&& \tilde{y} \geq 0 \end{aligned}$$

Where $\tilde{y} = \begin{pmatrix} y \\ \tilde{\sigma_1} \\ \tilde{\sigma_2} \end{pmatrix}$ with $y \in \mathbb{R}^{|C_R|}$ and $\tilde{\sigma_1} + \tilde{\sigma_2} = \sigma \in \mathbb{R}^K$, with y, σ being as (5.2.2).

Then we rewrite the conditions (5.2.4) and (5.2.6) in matrix form:

$$\left[\left(\left(-1^T_{|C_R|}, 1^T_K, -1^T_K\right) - f(\mathcal{P})^T \left(-A^T_{C_R}, A^T_K, -A^T_K\right)\right) \cdot \tilde{y}\right]_i = 0 \qquad \text{for all } i \in \{0, 1 \ldots, |C_R|\}$$
$$f(\mathcal{P})^T \left(b - \left(-A^T_{C_R}, A^T_K, -A^T_K\right) \tilde{y}\right) = 0$$

By the weak duality theorem we have:

$$\left(-1^T_{|C_R|}, 1^T_K, -1^T_K\right) \cdot \tilde{y} \leq f(\mathcal{P})^T \left(-A^T_{C_R}, A^T_K, -A^T_K\right) \tilde{y} \leq f(\mathcal{P})^T b$$

Using the strong duality theorem, we also have

$$\left(-1^T_{|C_R|}, 1^T_K, -1^T_K\right) \cdot \tilde{y} = f(\mathcal{P})^T b$$

Combining the two results from strong an weak duality we get the following two equalities:

$$\left(-1_{|C_R|}^T, 1_K^T, -1_K^T\right) \cdot \tilde{y} = f(\mathcal{P})^T \left(-A_{C_R}^T, A_K^T, -A_K^T\right) \tilde{y}$$
$$f(\mathcal{P})^T \left(-A_{C_R}^T, A_K^T, -A_K^T\right) \tilde{y} = f(\mathcal{P})^T b$$

Which is exactly what we aimed to obtain.
The other direction is quite similar but works in the opposite way we just did.

□

5.2.2 Column generation method

With the above formulation, we reduced the number of restrictions but increased the number of variables.

add numbers

We then proceed to define a *reduced master LP* on a reduced number of paths variables that can be increased if necessary to attain an optimal solution.

Pricing problem Following (Ahuja et al., 1993, p. 669), we consider only (5.2.5).

explain a bit more in detail

Solving the pricing problem efficiently

Explain why 0 is the dual value of empty restriction (see notebook)

Because of the particular nature of the restrictions in our problem, one can see that no restriction can span more than one time step in the time expanded network, and that a path cannot have two edges in the same time step (see section 4.2).

Based on these observations, we define the cost of an arc in the graph as:

$$w_{ij} = - \sum_{R \in C_R} \delta_R((i,j)) \cdot y_R \geq 0$$

One can notice that

$$\sum_{(i,j) \in P} \left(\sum_{R \in C_R} \delta_R((i,j)) \cdot y_R \right) = \sum_{R \in C_R} \delta_P(R) \cdot y_R$$

We can then rewrite the reduced cost as:

$$c_P^{\sigma,y} = |P| + \sum_{(i,j) \in P} w_{ij} - \sigma_k$$

So equation (5.2.5) becomes:

$$
\begin{aligned}
c_P^{\sigma,y} &\geq 0 && \forall P \in P^k \\
|P| + \sum_{(i,j)\in P} w_{ij} - \sigma_k &\geq 0 && \forall P \in P^k \\
|P| + \sum_{(i,j)\in P} w_{ij} &\geq \sigma_k && \forall P \in P^k \\
\sum_{(i,j)\in P} (w_{ij} + 1) &\geq \sigma_k && \forall P \in P^k \\
\min_{P\in P^k} \sum_{(i,j)\in P} (w_{ij} + 1) &\geq \sigma_k
\end{aligned}
$$

This shows that the pricing problem can efficiently be solved by searching for a weighted shortest paths p_k^* between s_k and t_k for all commodities $k \in [K]$. The weight for each arc $(i,j) \in A$ is given by $\tilde{w}_{ij} = w_{ij} + 1$. We can use Dijkstra's algorithm to efficiently solve this problem since the weights of the edges are greater than 0.

Initial Solution We produce an initial feasible solution using a greedy algorithm.

Algorithm 1: Finding and initial feasible solution for the column generation method

```
   Result: {p_1^0, ..., p_K^0}, where p_k^0 is a path from s_k to t_k
1  pathsFeasible = [ ];
2  Set weight at 1 for all edges;
3  for k ∈ {1, ..., K} do
4      find shortest path candidate p̃ from s_k to t_k using Dijkstra's algorithm ;
5      if p̃ does not have any conflicts with pathsFeasible then
6          add p̃ to pathsFeasible;
7      end
8      else
9          Increase weight of all edges of p̃ by 1;
10         Goto 4;
11     end
12 end
```

Where we say that a path p^* does not have conflicts with a set of paths P if using formulation (5.2.1) and putting all $f(p) = 1 \quad \forall p \in P \cup \{p^*\}$ all the restrictions are respected.

5.2.3 Relaxation and approximation

Once we solved the problem above, we do not have an integral multicommodity flow, hence the solution is not feasible. In order to get a feasible solution from the fractional flow, we use the following approximations:

fill path one by one in decreasing order of flow or just branch and bound

6 Experimental results

6.1 Arc formulation and flow formulation

Is the branch and bound process already implemented in Gurobi ? in this case just ask to solve the reduced master LP first as an IP, if it fails do one step of the column generation method and then ask again after updating the variables if the IP is feasible.

Check the percentage of integer solution in the relaxed formulation.

6.2 Approximation comparison

6.3 Comparison with reinforcement learning approach

7 On the mixed RL-CO approach

7.1 Ideas

7.2 Theoretical approach

7.3 Results

8 Conclusion

References

Ahuja, R. K., Magnanti, T. L., & Orlin, J. B. (1993). *Network Flows: Theory, Algorithms, and Applications*. Prentice hall.

Dai, W., Zhang, J., & Sun, X. (2017). On solving multi-commodity flow problems: An experimental evaluation. *Chinese Journal of Aeronautics*, *30*(4), 1481 – 1492.
URL `http://www.sciencedirect.com/science/article/pii/S100093611730122X`

Dufour Charles, G. E. (2019). Comparison of standard combinatorial optimization with reinforcement learning techniques on rescheduling problems. `https://github.com/dufourc1/SemesterProjectMA3`.

Skutella, M. (2008). An introduction to network flows over time. In *Bonn Workshop of Combinatorial Optimization*.