A network-flow-based procedure for scheduling trains in an underground mine

Mario Ahumada
Faculty of Engineering and Science
Universidad Adolfo Ibáñez
Santiago, Chile
mario.ahumada2007@alumnos.uai.cl

Eduardo Moreno
Faculty of Engineering and Science
Universidad Adolfo Ibáñez
Santiago, Chile
eduardo.moreno@uai.cl

***Abstract*—We present an applied combinatorial problem that arises in a large-scale underground mine that uses trains to extract material from the interior of the mine. The mine requires the trains to be scheduled throughout the day to maximize the amount of extracted material. The main difficulty of the problem is that a single-track tunnel connects the interior and the exterior of the mine, which is, in practice, the bottleneck of the extraction process. We propose a network-flow based algorithm for this purpose. The algorithm iteratively routes a train in the network, solving a min-cost flow problem over a three-dimensional network representing the position and load of the train at a given time. Computational comparisons show that the proposed algorithm extracts 16.5% more material than the current methodology utilized in the mine.**

Keywords*—*Underground mining, train scheduling, network flow.

I. Introduction

This work is motivated by the operation of the underground mine El Teniente, which is located inside the Andes mountains, in the central zone of Chile. El Teniente is the worlds largest underground copper mine, with over 450 thousand metric tons of copper produced every year. Since 1940, the mine has been extracted using a method called "panel caving" and "block caving", which extract the material and sends it, using gravity, to a lower transportation level after passing through a crusher, where trains extract the material to the outside of the mountain (see Figure 1).

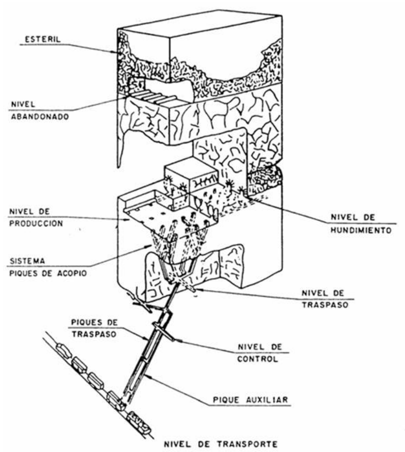

Fig. 1. Underground mine levels (from [5])

This rail network connects several extraction points (*pits*) inside the mine and has a single-track tunnel of 2.3 kilometers to the outside of the mine, where the material is dropped into *depots*. Because the pits are filled with material continuously, trains are required to visit a depot several times during the day but are sufficiently spaced for the pit to have enough material for the train. One of the bottlenecks in production is the capacity of trains to remove the material from the inside of the mine. This is because the tunnel has a single rail track, so trains *block* each other when they are entering or exiting the interior of the mine. The goal of this work is to provide a practical tool that helps schedule trains by considering the blocking constraints and maximizing the material extracted from the interior of the mine.

A first approach to model this problem is as a vehicle-routing-problem model (see [1]). In particular, it is possible to model the pit production as a time window (see [2] and [3]) because a pit should be visited after it has enough material to fill the train and before it reaches its capacity, imposing a time-window for its visit. However, this model does not consider the blockage between vehicles that occurs inside the tunnel, and adding these constraints results in a problem that is even more difficult to solve. Therefore, it is not practical to implement this model, due to the periodicity of the problem. A second idea is to use a flow-based model with binary variables similar to the one proposed in [4]. However, these MIP models require several hours of computation to produce a schedule. It is of interest to the final users to have a quick tool that allows them to evaluate different scenarios and to re-schedule the trains if a rail track fails or if a pit stops producing material. Hence, it is proposed a network-flow based heuristic to solve the problem in a few seconds.

II. Instance description

To explain the details and constraints of the problem, we present in this section the particular instance that we wish to solve. This example contains 10 pits P, with each of them having a production rate, working hours and a maximum capacity of material than can be accumulated in the pit. There are also two different types of materials that are produced in the pits: coarse and fine ore, and the type of material in a given pit can change over time. The material should be extracted to the depots located outside the mine. A schematic diagram of the rail network, pits and depots in our example is presented

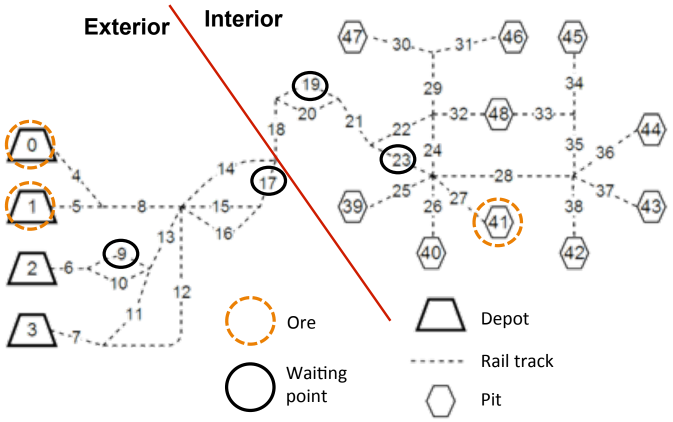

Fig. 2. Schematic diagram of the mine and its rail network

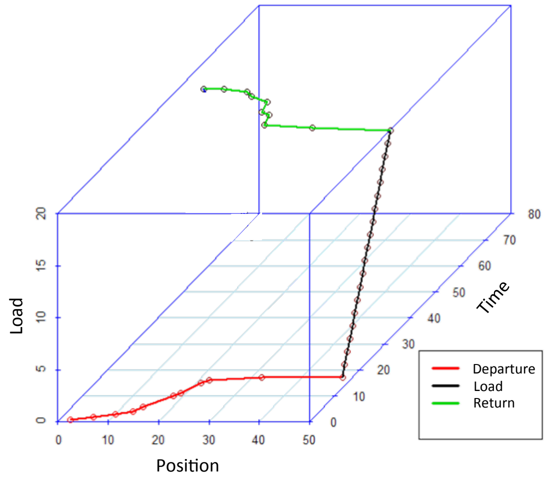

Fig. 3. Example of a trip from depot 1 to pit 48 over the flow network

in Figure 2. In the figure, pits are numbered from 39 to 48, depots from 1 to 4 and the rest of the numbers correspond to rail track. The network also has a few positions specially designed as waiting points for trains, and they are numbered 9, 17, 19 and 23.

There are also 7 trains (5 of them for coarse ore, and 2 for fine ore) composed of 19 wagons that can transport up to 69 metrics tons of coarse material or 80 tons of fine material. Trains move, on average, at a speed of 10 km/hr.

III. A Network Flow Model for Scheduling a Single Train

The proposed algorithm is based on sequential schedules of single trains, where a schedule of a round-trip of a train is obtained by solving a min-cost network flow problem. We first discretize the time (in minutes) and the load of each train (in wagons). We construct a network model in which node N corresponds to a triplet (p, t, c), where p represents a location in the mine (a pit, a depot, or a rail track), t represents a time period (in minutes), and c represents the load of the train. We also add dummy nodes s and t for the source and sink of the flow, respectively. Then, we construct the following sets of arcs:

Movement arcs $\langle (u, t_1, c), (v, t_2, c) \rangle$: these arcs correspond to the movement of the train inside the mine from position u to a neighbor position $v \neq u$. The (discretized) time $t_2 - t_1$ required to move from u to v is computed according to the average speed of the train and the distance between these two positions.

Waiting arcs $\langle (u, t_1, c), (u, t_1 + 1, c) \rangle$: these arcs correspond to a train waiting at a given position u for one time step.

Loading arcs $\langle (u, t_1, c), (t, t_2, c + 1) \rangle$: these arcs correspond to a train loading material at pit u. The time $t_2 - t_1$ is the time required to load a full wagon of the train. These arcs exist only if u corresponds to a pit.

Start/end arcs $\langle s, (u, t_1, 0) \rangle$ and $\langle (v, t_2, c), t \rangle$: these arcs represent the start and the end of a train round trip, respectively. Points u and v are depots of the same type of material (coarse or fine) that the train transports.

All arcs have lower and upper bounds of zero and one unit of flow, respectively. The unitary costs of each arc are the following: cost -1 (i.e., benefit 1) for the loading arcs, cost $\alpha(t_2 - t_1)$ for movements arcs and costs β and γ for waiting arcs, when its corresponding positions are waiting-points and non-waiting-points, respectively. By setting $0 < \beta \ll \gamma \ll \alpha \ll 1$ correctly, a min-cost flow of one unit from s to t corresponds to a train trip from a depot to a pit and then back to a depot, after maximizing its load in the minimum possible time. If the train needs to wait, it prefers to do so at a waiting point in the rail network. An example of a train trip is presented in Figure 3.

Note that we can only solve this problem for one train. If two trains want to travel across the tunnel in opposite directions, then the flow representing each train can be split into two half flows separated by one time step, and in this way, both flows can simultaneously traverse the arcs corresponding to the tunnel.

IV. A Heuristic Algorithm for Scheduling Several Trains

The network flow model solves the problem for scheduling a single round-trip of a train over the network. To obtain a daily schedule for several trains, the principle of the algorithm is to solve the problem sequentially for one train at a time, modifying the network at each iteration. To consider the interaction of several trains and the availability of material at each pit, we define two sets of "forbidden" arcs $\mathcal{F}_1$ and $\mathcal{F}_2$, where the flow over these arcs should be zero. The first set $\mathcal{F}_1$ of arcs is determined by the material available at each pit: if at time t the pit u will have no more than c wagons of material available, then we forbid arcs $\langle (u, t, c), (u, t + \Delta t, c + 1) \rangle$. Hence, no train will load more than c wagons of material from that pit. Movements of other trains, previously scheduled, determine the second set of forbidden arcs $\mathcal{F}_2$. This means that if a previous train is moving using a rail track, then we forbid the use of this rail track in that period, and also in the previous and following periods. More precisely, if a previous iteration sent flow through arc $\langle (u, t_1, c), (v, t_2, c) \rangle$, then we add to $\mathcal{F}_2$ the arcs $\langle (u, t_1, c), (v, t_2 - \delta, c) \rangle$, $\langle (u, t_1, c), (v, t_2 + \delta, c) \rangle$ and $\langle (v, t_1, c), (u, t_2, c) \rangle$. These first two arcs allow us to maintain

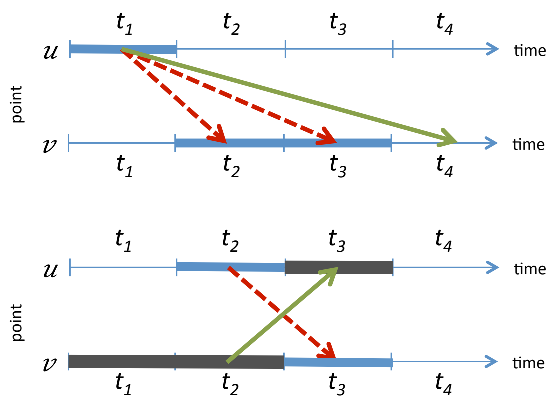

Fig. 4. Example of blocked arcs added to $\mathcal{F}_2$

a safety time δ between two consecutive trains, and the last arc allows us to avoid the case of two trains in opposite directions, crossing each other (see Figure 4). Note that a flow over this modified network will avoid the arcs used by the previously scheduled trains and will consider the material extracted by previous trains.

Finally, we present the iterative steps of our algorithm:

1) Choose an available train at time T. If no train is available, increase T to $T+1$.
2) Check the potentially available material for time $\hat{t} > T$ and update the set $\mathcal{F}_1$.
3) Fix the upper bound of the flow over arcs in $\mathcal{F}_1 \bigcup \mathcal{F}_2$ to zero.
4) Solve the min-cost flow problem for one unit of flow from s to t.
5) Add to $\mathcal{F}_2$ the arcs corresponding to the previous flow.
6) Discount the extracted material from the corresponding pits.
7) Return to step 1, to route the next available train in time T.

These steps are repeated until the full day has been scheduled.

V. Computational Comparison

A. Example of a daily schedule

An example of a daily schedule is plotted in Figure 5. The upper figure shows the positions over time of seven trains in the network, and the lower figure shows the material available at three pits over time. It can be seen that all trains go to a specific pit (positions 39–47), load material, and then come back to the depot (positions 1–4), in accordance with the constraints of the problem. Note that even if trains can, in theory, go to several pits before returning to the depot, this does not happen in practice. It also can be seen that the trains are synchronized with each other to traverse the tunnel connecting the interior and exterior of the mine (position 18). The figure also shows that trains prefer to wait at the waiting points (mostly at positions 17 and 19).

Some interesting patterns of the obtained solution can be observed from this figure. For example, note that the train represented by the magenta line waits at position 19 before

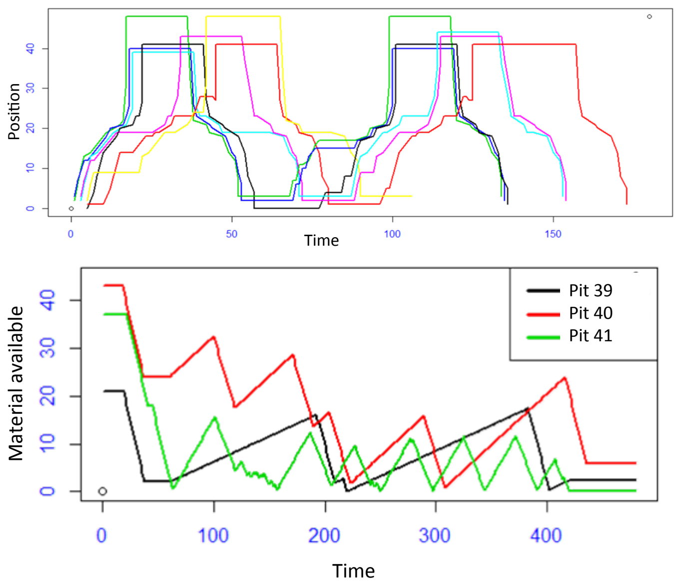

Fig. 5. Example of the trains positions and material available over time for a daily schedule

entering the zone with the pits, even if the target pit and the path to this pit are available. This is because this train needs to wait for the train represented by the cyan line before returning, so it prefers to wait at a waiting point. Another interesting case are the trains represented by the red and black lines. Both trains can only transport fine material, which is only available at pit 41. Hence, the red train waits at the depot zone before starting its trip to avoid interfering with the other trains. Also note that during the red train's second visit to the pit, it stays a long time. This is because the pit does not have enough material to fill the 19 wagons of the train, so the train stay at the pit until enough material is produced. This can be seen also in the second figure, where the green-line representing pit 41 has a different pattern in the corresponding period of time (minutes 120–160). Note that this behaviour is an exception, and in other pits, trains wait until pits have enough material before visiting them.

B. Comparison with the current methodology

We benchmark our algorithm against the current methodology for computing the daily schedule, which is based on simulations. We compare our algorithm for real parameter values (pit production rates, available tracks, type of trains, working hours and other parameters) for each day of the month of April 2013. The computational time required to obtain a daily schedule is between 1 and 3 minutes, which is considerably lower than the 20-30 minutes required by the current methodology. The operational results of our algorithm are presented in Table I.

TABLE I. Average values obtained using April 2013 data.

	Cycle Time (minute)	Extracted Material (Tons.)	Train Trips
Current methodology	104	121,000	88
Flow algorithm	84	141,000	103
Variation	-19.2%	+16.5%	+17.0%

It can be seen that our algorithm improves the total material extracted by an average of 16.5%. This improvement is obtained by reducing the time that trains are waiting. In fact, the average time of a round trip is reduced by 19.2%, allowing 17% more trips during a day.

VI. Conclusion

We formulated a network-flow-based algorithm to solve a trip scheduling problem for the train network of an underground mine. The algorithm iteratively solves a min-cost network flow to route a single train and removes at each iteration the arcs from the network that get blocked by the previously scheduled trains. Computational experiments show that the scheduled proposed by the algorithm can extract 16.5% more material than the current methodology. Additionally, from a practical point of view, the proposed approach is very fast to solve, allowing computation and evaluation of different scenarios in a few seconds.

Acknowledgment

The authors would like to thank the financial support of grants Anillo ACT-88 and Fondecyt 1130681. We also thank Enrique Rubio and Redco Mining Consulting for providing the data of the problem.

References

[1] P. Toth and D. Vigo, *The vehicle routing problem*. Society for Industrial and Applied Mathematics, 2002.

[2] M. L. Fisher, K. O. Jörnsten, and O. B. Madsen, "Vehicle routing with time windows: Two optimization algorithms," *Oper. Res.*, vol. 45, no. 3, pp. 488–492, 1997.

[3] O. Bräysy and M. Gendreau, "Vehicle routing problem with time windows, part i: Route construction and local search algorithms," *Transp. Sci.*, vol. 39, no. 1, pp. 104–118, 2005.

[4] Q. Lu, M. Dessouky, and R. C. Leachman, "Modeling train movements through complex rail networks," *ACM Trans. Model. Comput. Simul.*, vol. 14, no. 1, pp. 48–75, 2004.

[5] V. C. de Proyectos, "Informe de principales decisiones," Codelco Chile, Tech. Rep., 2011.