

Early start and late start algorithms to improve the solution time for long-term underground mine production scheduling

by E. Topal*

Synopsis

Mixed integer programming (MIP) has been used for optimizing production schedules of mines since the 1960s. The major problem in the long-term production scheduling for an entire orebody is that the number of integer variables needed to formulate an MIP model is too large to solve the formulation. This number may reach well over one hundred thousand. To overcome this difficulty, this paper presents two new algorithms to reduce the size of the problem. These algorithms assign an earliest and latest possible start date for each machine placement, eliminating the integer variables that correspond to machine placement before its early start date and after its late start date. A case study based on Kiruna Mine, the second largest underground mine in the world, is summarized in the paper. It shows substantial improvement in the solution time required using the new algorithms. This increased efficiency in the solution time of the MIP model allows it to be applied to Kiruna Mine, with the potential to increase substantially the net present value (NPV) of the project.

Introduction

Mining projects start with the exploration stage, drilling and sampling is undertaken to locate the orebody and define its grade. A block model of the orebody is developed to represent its geological and economic attributes. The geological attributes include the grades of the different minerals comprising the orebody, and their mineralogy, density and tonnage. The geological block model can be converted to an economic block model by applying operational costs and commodity prices to the blocks.

The next step in a mining project is to determine the best possible mining strategy based on the answers to the following questions:

- Is the optimal mining method by surface mining, underground mining or a combination of the two?
- If the optimal mining method is surface mining, what is the break-even depth below the surface (ultimate pit limits)?
- If the optimal mining method is underground mining, what type of underground mining method(s) should be used?

For a surface operation, only the ore inside the ultimate pit boundary can be economically mined using open pit methods under the defined economic conditions. Outside the ultimate pit boundary, underground methods can often be used. After determining the ultimate pit boundary, the next step is to develop the production schedule for the mine in such a way as to maximize the company's profit.

Generally, the number of blocks are too numerous to allow a feasible multi-time period production schedule to be developed. In order to reduce the size of the problem and facilitate a solution, it is common to divide the ultimate pit into smaller volumes for use in optimizing the extraction sequence. Each of these smaller volumes is called a 'pushback', 'incremental cut', or 'phase'. These pushbacks are used as a guide in determining annual production schedules for the mine life. Unfortunately, there are no such methods available for reducing the problem size for underground mining method applications (Topal, 2003).

Review of MIP applications and solution time

Production planning, in a general sense, is one of the most important areas in which operations research (OR) techniques are applied. The OR method most commonly applied to production scheduling problems is known as linear programming (LP). An LP programming model consists of a linear objective function of the form:

* *Senior Lecturer, School of Engineering and Senior Researcher, CRC Mining, The University of Queensland Brisbane, Australia.*

© *The Southern African Institute of Mining and Metallurgy, 2008.* *SA ISSN 0038–223X/3.00 + 0.00. Paper received Apr. 2007; revised paper received Oct. 2007.*

Early start and late start algorithms to improve the solution time

Maximized or (*Minmize*) $Z = c_1x_1 + c_2x_2 + c_3x_3 + \ldots\ldots\ldots\ldots + c_nx_n$

Subject to a set of linear constraints, without loss of generality, of the following form:

$$a_{11}x_1 + a_{12}x_2 + \ldots\ldots\ldots\ldots a_{ln}x_n \leq b_1$$
$$a_{21}x_1 + a_{22}x_2 + \ldots\ldots\ldots\ldots a_{2n}x_n \leq b_2$$
$$\vdots \quad \vdots \quad \vdots \quad \vdots$$
$$a_{m1}x_1 + a_{m2}x_2 + \ldots\ldots\ldots\ldots a_{mn}x_n \leq b_m$$

and a set of non-negativity restrictions: $x_1, x_2, \ldots\ldots\ldots\ldots, x_n \geq 0$

In this formulation, Z is the value of the objective function, e.g., profit or cost, x_j are decision variables, i.e., they represent values to be determined, a_{ij} and c_j are constants whose values are dictated by the nature of the problem and b_i is the right-hand side constant value such as available amount of resource i. In certain situations, the decision variables must assume integer values. When this restriction is added to the problem, the model is called an integer programming (IP) model. Such a requirement arises naturally when the variables represent entities such as packages, people, and trucks that cannot be divided. Integers, specifically, binary variables are used to capture yes/no decisions such as whether or not to mine the ore from a particular block at a given time period. For example, if the block i is mined in a given time period, x_i assumes the value of one and zero otherwise. If some variable values are allowed to be continuous while others must assume integer values, the resulting model is a mixed integer program (MIP).

Mixed integer programming models are recognized as having significant potential for optimizing production scheduling for surface and underground mines. However, MIP model solution times depend mainly on the number of integer variables and also on the number of constraints in the model. MIP solution time often increases exponentially as the number of integer variables increases. The Branch-and-Bound Algorithm is the most frequently used solution technique for MIP problems. Worst case performance for this algorithm involves evaluating each candidate problem. Therefore solving large problems becomes exponentially more complex with increasing problem size. Fortunately, in practice all nodes are usually not examined because some branches are fathomed early.

For mining application, if there are say 100 000 production blocks that need to be scheduled over a period of 5 years, it would require 500 000 binary variables to generate the MIP model. This would make it very difficult or even impossible to solve. In order to overcome this problem in surface mine applications, the production scheduling model is generated for each pushback and if the model is still too big to solve, some aggregation techniques are applied to the block within the pushback. Tolwinski (1998) used an algorithm that combines the production blocks within the ultimate pit boundary into elementary units called 'atoms' and builds a tree of potential solutions for the scheduling. Each atom is characterized by its location on a bench, the pushback to which it belongs and the quantities of attributes that it contains. Next, pushbacks are generated by combining these atoms using the Lerch and Grossman's (LG) algorithms (Lerch and Grossmann, 1965). Finally, the method generates the production schedule by solving a series of optimization problems that contains all the optimal solutions, employing dynamic programming. The number of these problems equals the number of target variables, i.e., output levels, strip ratios, blending requirements, plus one (maximization of the NPV). The size of the atoms will affect the optimality of the final production schedule dramatically.

Ramazan (2007) proposed a new type of algorithm called 'Fundamental Tree Algorithms' based on linear programming to aggregate the production blocks of the model without violating optimality. A set of combined blocks is termed a 'Fundamental Tree' if these blocks have the following properties: (i) can be mined without violating the slope constraints; (ii) the total economic value of combined blocks as a fundamental tree must be positive; and (iii) a fundamental tree (FT) cannot be partitioned into smaller trees without violating (i) and (ii). Each fundamental tree is treated as a mining block having a certain tonnage, metal content and quality parameters. Since the binary variables are assigned to the FTs instead of blocks, the number of integer variables and constraints are dramatically reduced. However, the number of trees to be scheduled will increase with the size of the deposit. The required number of binary variables to schedule these trees would make solution time quite unreasonable for the large size deposit. Application of this approach to a case study reduced the number of blocks within the ultimate pit limits from 38 457 to 5 512 and the application of the MIP scheduling model to the trees improved the overall project NPV by ~7% higher than the best NPV generating schedule obtained among three commonly used traditional software packages including MINTEC's M821V, Earthworks' NPV scheduler, and Whittle's Milawa schedulers in the Four-X program. Ramazan and Dimitrakopoulos (2004) proposed an MIP scheduling formulation that allows the waste block variables to be defined as linear rather than binary. The per cent reduction in the number of binary variables was 100*(number of waste blocks)/(total number of blocks), which is significant in open pit mines and makes the MIP formulation soluble within a reasonable time.

Underground mine production scheduling has attracted more attention in the last 10 years. Trout (1995) formulated and attempted to solve a mixed integer multi-period production scheduling model for underground stoping operations for base metals. Although the model produced a solution that is considerably better than what is currently realized in practice, solution time exceeded 200 hours, without a guarantee of optimality. Carlyle and Eaves (2001) used an integer programming model to plan a production schedule for a sub-level stoping operation at Stillwater Mining Company. The model provided near-optimal solutions, for a 10-quarter planning period, to maximize revenue from mining platinum and palladium. However, the authors did not describe any special techniques to expedite solution time.

McIsaac (2005) formulated the scheduling of underground mining of a narrow-veined poly-metallic deposit utilizing MIP. The deposit was divided into eleven zones and scheduled over quarterly time periods. The production schedules were generated for each zone, rather than for the individual stopes within these zones. The model was solved in 30 minutes with 1 200 variables (the number of integer variables were not specified in the paper) using Microsoft

Early start and late start algorithms to improve the solution time

Excel (Fontline's Xpress as a solver). Although the model size and the complexity of the model were reduced by limiting the number of variables and just scheduling the zones over quarterly time periods, there is no method available to increase the solution efficiency of the MIP model.

Topal (2003) and Topal *et al.*, (2003), generated a long-term production scheduling MIP model for a sub-level caving operation and successfully applied it to Kiruna Mine, one of the largest underground mine in the world. The mine was divided into ten production areas, each of which has its own group of ore passes and ventilation shafts; this group is also known as a shaft group. These production areas are about 400 to 500 m in length and contain from 1 to 3 million tons of ore and waste. One or two 25 ton capacity load haul dump units (LHDs) operate on each sub-level within each production area to transport the ore from the cross-cuts to the ore passes. The place where each LHD operates is also called a machine placement and is about 200 to 500 m in length. Each machine placement contains 10 to12 production blocks, which are the same height as the mining sub-level (about 28.5 m) and extend from the hangingwall to the footwall. Figure 1 defines the shaft group, machine placements, production areas, production blocks, and LHDs.

The model determined which section of the ore to mine, and when to start mining them so as to minimize deviation from the planned production quantities, while adhering to the geotechnical and machine availability constraints. The model solved for multi-period production scheduling and is detailed in Appendix I (Topal, 2003; (Kuchta *et al.*, 2004). Because the model size is large, two new algorithms are developed herein in order to reduce the size of the model, without violating optimality. This paper proposes early and late start algorithms for optimizing production scheduling in underground mining.

Overview of underground mining operations

Underground mining operation is much more complex to schedule and manage than surface mining. Due to the reduced flexibility in underground mining operations imposed by geotechnical, equipment and space constraints, underground operation requires more constraints and binary variables in order to generate the schedule. The following section explains the constraints imposed by sub-level caving operations.

Continuity of mining requirement

Once a machine placement has started to be mined, mining that machine placement must progress continuously for each production block within the machine placement and in a specific order, until all available ore has been removed. The continuity requirement for the machine placement located at the 820 m level from Machine Placement 29_30 can be seen in Figure 2 a. Once mining has started for this machine placement, it must continue for the rest of the 15 time periods.

Vertical sequencing requirement between levels

To prevent undermining between levels, mining at a particular machine placement cannot be started until 50% of the machine placement on the level above has been mined. As seen in Figure 2 b at level 849, mining can start at Machine Placement 16_18 if the Level 820 Machine Placement 16_18 Production Block 3 is being mined and so on (assuming there are 5 production blocks at each machine placement).

Horizontal sequencing requirement

To prevent blast damage to adjacent locations, machine placements to the left and right on the same level must begin

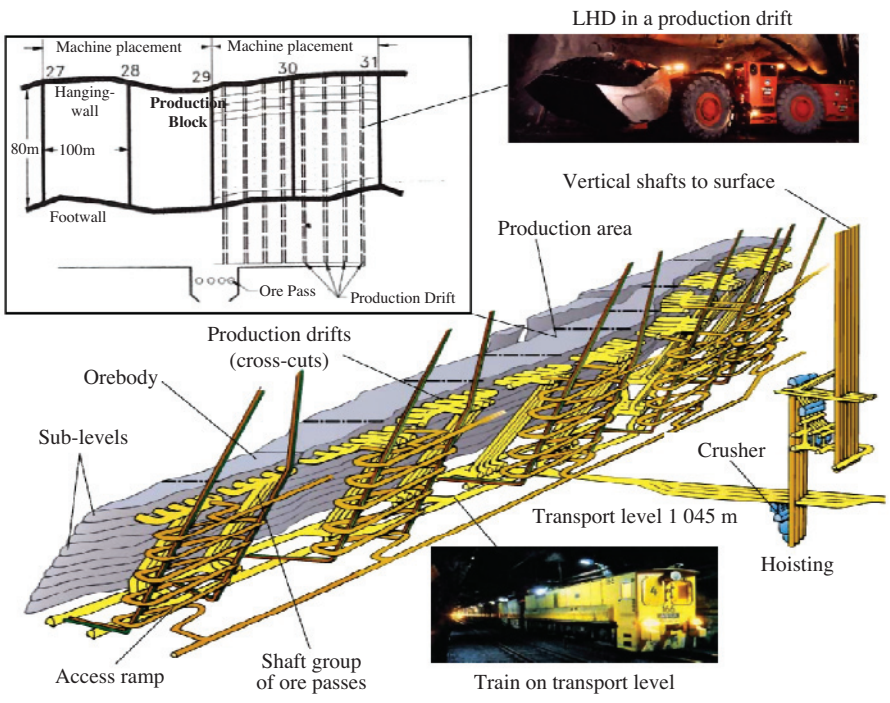


Figure 1—The 1 045 m main transportation level at the Kiruna Mine and representation of some definitions

Early start and late start algorithms to improve the solution time

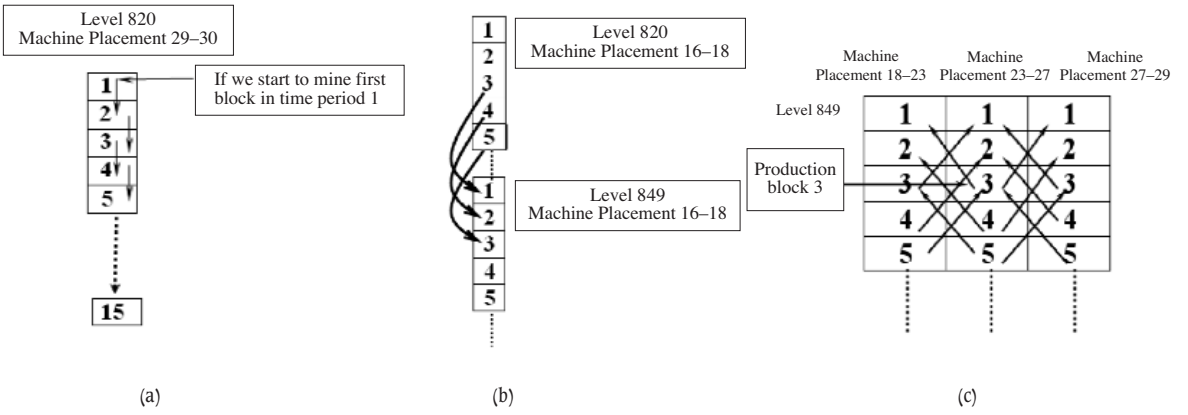


Figure 2—Geotechnical mining requirements for an underground sublevel caving operation

to be mined after no more than 50% of the given machine placement has been mined. As seen in Figure 2 c at Level 849, if the Machine Placement 23_27 is being mined, the adjacent Machine Placements 18_23 and 27_29 have to start being mined if Machine Placement 23_27 has reached Production Block 3, assuming the total number of blocks at Machine Placement 23_27 is 5.

Machine limitation for each shaft group

Each shaft group has 3 or 4 ore passes, one of which is used for mine development. Only 2 or 3 load haul dump (LHD) units can simultaneously be active in each shaft group.

In reality, this lack of geotechnical and machine flexibility would help to reduce the number of binary variables in the scheduling model without violating optimality.

Increase the efficiency of the MIP model: earliest start and latest start time algorithms

Algorithm 1: Early start algorithm

Because of vertical sequencing requirements between levels, horizontal sequencing requirements between adjacent machine placements and the fact that every shaft group is constrained by a maximum number of simultaneously operable LHD units, there are restrictions on the earliest time at which a machine placement can start to be mined.

The logic of the early start algorithm follows:

- a: Machine placement to which we assign an early start date
- a_o: Machine placement constrained by vertical sequencing to a
- a_l: Machine placement constrained by left horizontal sequencing to a
- a_r: Machine placement constrained by right horizontal sequencing to a
- n_v: Parameter representing the number of active machine placements allowed for shaft group v
- b_a: Parameter representing the number of production blocks contained in machine placement a
- m_v: Dynamic array containing the max{n_v} greatest completion times for each shaft group v for machine placements for which the early start dates have been set, ordered from greatest to least.

The function member(X,y) to denote the yth (greatest) member of set X.

- *Step 0*—Initialize by setting the earliest start date (ES) to 1 for all machine placements a and correspondingly initialize all elements of the dynamic array m to 1.
  Loop through the sorted list of sublevels starting with the uppermost sublevel
  For each sublevel do
  Set the early start date for all machine placements on the current sublevel according to the vertical sequencing constraints
  For each machine placement on current sublevel do
- *Step 1*—Set the early start date for a affected by the vertical sequencing constraint owing to a_o and the corresponding number of production blocks within machine placement a_o:
  Set ES(a) = max (ES(a), ES(a_o) + 50%*(b_{a_o})).
  Next machine placement
  Set the early start date according to the shaft group constraint
  For each shaft group do
  For each machine placement in the current shaft group on the current sublevel do
- *Step 2*—Set the early start date for a affected by the shaft group constraint owing to m_v and n_v:
  Set ES(a) = max (ES(a), member(m_v, nv)).
  Next machine placement
  Next shaft group
  Set the early start date for all machine placements on the current sublevel according to the left and right horizontal sequencing constraints. Repeat until no changes are made.
  Repeat
  Set changes_made = false
  For each machine placement on current sublevel do
- *Step 3*—Set the early start date for a affected by the left and right sequencing constraints owing to a_l and a_r, respectively:
  If ES(a_l) > (ES(a) +50%*(b_a))
  Set ES(a) = ES(a_l) – 50%*(b_a).
  Set changes_made = true
  If ES(a_r) > (ES(a) +50%*(b_a))
  Set ES (a) = ES(a_r) – 50%*(b_a).
  Set changes_made = true

Early start and late start algorithms to improve the solution time

Next machine placement
Until changes_made = False

- *Step 4*—Update both the completion time for machine placement α and the corresponding array m such that the array contains the max{n_v}greatest completion times for the machine placements for which the early start dates have been set.
  Set ES(a) = ES(a) + b_a
  If ES(a) > min(ES() ε m_v)
  Let $m_v = m_v$/member(m_v, n_v) ∪ ES(a)
  Next sublevel

Note that we could improve the efficiency of this algorithm (at the expense of exposition!) by executing the loop in Step 3 only until changes_made has been set to true for the first time, and then resetting the affected early start dates.

Algorithm 2: Late start algorithm

The dependency resulting from vertical and horizontal sequencing requirements and the limited number of active LHDs at a machine placement precludes underlying machine placements from being mined. A late start date can be assigned to each machine placement to prevent it from starting to be mined so late that adjacent and underlying machine placements eventually become 'locked in', precluding production of the required tonnages. The logic of the algorithm follows, and should be repeated for each machine placement to which we wish to assign a late start date.

- *Step 0*—Initialize by setting the late start date (LS) to T (the maximum number of time periods in the planning horizon) for all machine placements.
- *Step 1*—Define the boundaries of influence for machine placement a. That is, determine all Production Blocks b within adjacent machine placements a on the same sublevel that can never be mined as a result of not mining a. Denote this set ϑ_a.
- *Step 2*—Given a is never mined during the time horizon, determine the Production Blocks b within machine placements adjacent to a that can be mined in each time period t while adhering to the vertical and horizontal sequencing constraints. Denote this set ω_{at}.
- *Step 3*—Given the Production Blocks b that can be mined (from Step 2), calculate the cumulative available tonnage $\sum_{u=1}^{t} \phi_{uk}$ of each ore type k for every time period t.
- *Step 4*—Determine the cumulative required production tonnage $\sum_{u=1}^{t} d_{uk}$ of each ore type k for each time period t. Then, for each time period t = 1...T, check whether: $\sum_{u=1}^{t} \phi_{uk} < \sum_{u=1}^{t} d_{uk}$.

If the condition is true, stop. Time period t-1 defines an upper bound on the latest start date for a.

To determine whether this upper bound is tight, check whether:

$$\sum_{q=1}^{u} \phi_{qk} \geq \sum_{q=1}^{u} d_{qk} \quad \forall u = t + 1,...,T \quad \forall k.$$

If $\exists$ u, k for which $\sum_{q=1}^{u} \phi_{qk} < \sum_{q=1}^{u} d_{qk}$, set the latest start date to, $t' = t$–2 and check again.

Continue to move the late start date back by a single time period until:

$$\sum_{q=1}^{u} \phi_{qk} \geq \sum_{q=1}^{u} d_{qk} \quad \forall u = t' + 1,...,T \quad \forall k$$

Let t' be the late start date for a.

A late start (LS) for machine placements adjacent to a, i.e., a_l and a_r, owing to horizontal sequencing constraints is set as follows:

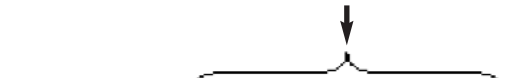
LS(a_l) = LS(a) + 50%*(b_a).
LS(a_r) = LS(a) + 50%*(b_a).

Examples

The following hypothetical example illustrates the early and late start algorithms. Figure 3 represents a section with two underground sublevels for a typical sublevel caving mine. Machine Placement 8 (MP8) is selected to determine the earliest start time and Machine Placement 3 (MP3) is selected for the latest start time algorithm. Table I presents the available tonnage for the B1 ore type (high quality of ore) from each production block for each machine placement.

Early start algorithm is applied to the example as follows:

- *Step 0*—Initialize by setting the earliest start date (ES) to 1 for all machine placements a
  Let: ES(MPj) =1 $\forall j$ = 1..10.
  For each sublevel do
  For each machine placement on current sublevel do
- *Step 1*—Set the early start date for MP8 affected by the vertical sequencing constraint:
  ES(MP8) = max(ES(MP8), ES(MP3) + 50%*(b_{MP3})) = max(1, 1+ 3) = 4.



MP1	MP2	MP3	MP4	MP5
PB1	PB1	PB1	PB1	PB1
PB2	PB2	PB2	PB2	PB2
PB3	PB3	PB3	PB3	PB3
PB4	PB4	PB4	PB4	PB4
PB5		PB5		PB5
		PB6		

Sublevel 2

MP6	MP7	MP8	MP9	MP10
PB1	PB1	PB1	PB1	PB1
PB2	PB2	PB2	PB2	PB2
PB3	PB3	PB3	PB3	PB3
PB4	PB4	PB4	PB4	PB4
PB5	PB5	PB5	PB5	PB5

Figure 3—Illustration of two sublevels of a mine; sublevel 1 has 5 machine placements (MPs) 1–5 and sublevel 2 contains 5 machine placements (MPs) 6–10 and each MPs includes different production blocks (PBs)

Early start and late start algorithms to improve the solution time

Table I

Tonnage of B1 ore contained in each production block

	PBs	B1 Tonnage		PBs	B1 Tonnage
MP1	PB1	40*+	MP6	PB1	40
	PB2	75+		PB2	35
	PB3	45		PB3	40
	PB4	40		PB4	45
	PB5	80		PB5	70
	PBs	**B1 Tonnage**		**PBs**	**B1 Tonnage**
MP2	PB1	110*+	MP7	PB1	30
	PB2	90+		PB2	50
	PB3	85		PB3	60
	PB4	120		PB4	140
	PB5	135			
	PBs	**B1 Tonnage**		**PBs**	**B1 Tonnage**
MP3	PB1	80+	MP8	PB1	90
	PB2	100		PB2	110
	PB3	120		PB3	90
	PB4	150		PB4	70
	PB5	100		PB5	75
	PB6	60			
	PBs	**B1 Tonnage**		**PBs**	**B1 Tonnage**
MP4	PB1	60*+	MP9	PB1	40
	PB2	50+		PB2	45
	PB3	50		PB3	50
	PB4	20		PB4	120
	PB5	100			
	PBs	**B1 Tonnage**		**PBs**	**B1 Tonnage**
MP5	PB1	50*+	MP10	PB1	35
	PB2	40+		PB2	45
	PB3	30		PB3	85
	PB4	40		PB4	70
	PB5	70		PB5	65

- *Step 2*—Set the early start date for MP8 affected by the shaft group constraint:
  ES(MP8) = max[ES(MP8), member{ES(MP2) + (b_{MP2}), ES(MP3) + (b_{MP3}), ES(MP4)+ (b_{MP4}), 3}]=max[4, member{(1+4, 1+6, 1+4), 3}] = max(4, 5) = 5.
  Next machine placement
  Repeat
  Set changes_made = false
- *Step 3*—Set the early start date for MP8 affected by the left and right sequencing constraints, respectively:
  Is ES(MP7) > (ES(MP8) + ceil(50%*(b_{MP8})))?
  Because 1 < 1 + 3, the above condition is false.
  ⇨ Set ES(MP8) = 5.
  Is ES(MP9) > (ES(MP8) + ceil(50%*(b_{MP8})))?
  Because 1 < 1 + 3, the above condition is false.
  ⇨ Set ES(MP8) = 5.
  Next machine placement
  Until changes_made = False
  Because changes_made = false, the early start for machine placement 8 (MP8) a is determined to be 5.

The late start algorithm is applied to the example as follows:

- *Step 0*—Select a machine placement to which to assign a latest start date (MP3).
- *Step 1*—Determine the production blocks in set ϑ_a, i.e., the production blocks within machine placement a that will never be mined if MP3 is not mined. Those production blocks have value of zero in Table II.
- *Step 2*—Determine the production blocks in the set ω_{at}, i.e., the set of production blocks within machine placement a that can be mined during time period t given MP3 is never mined during the time horizon. Table II presents available production blocks with in its machine placement along the time periods.
- *Step 3*—Given the production blocks that can be mined assuming machine placement 3 is never mined (from Step 2), the available cumulative tonnage of B1 ore type k for every time period t ($\sum_{u=1}^{t} \phi_{uk}$) is displayed in column 3, Table III.
- *Step 4*—The required cumulative demand for B1 ore for each time period t, $\sum_{u=1}^{t} d_{uk}$, appears in column 2, Table III.

For example, the available B1 tonnage for the first time period is calculated using the numbers with an accompanying asterisk in Table III as follows:

40 + 110 + 60 + 50 = 260 tons.

As is seen in Table III, by comparing column 2 with column 3, if we do not mine MP3 by time period 3, the total required B1 production is 750 tons and the total available B1

Table II

The production blocks that can be mined and the corresponding time period(s) in which they can be mined

Time period	MP1	MP2	MP3	MP4	MP5	MP6	MP7	MP8	MP9	MP10
1	PB1	PB1	0	PB1	PB1	0	0	0	0	0
2	PB1 PB2	PB1 PB2	0	PB1 PB2	PB1 PB2	0	0	0	0	0
3	PB1 PB2 PB3	PB1 PB2	0	PB1 PB2	PB1 PB2 PB3	PB1	PB1	0	PB1	PB1
4	PB1 PB2 PB3 PB4	PB1 PB2	0	PB1 PB2	PB1 PB2 PB3 PB4	PB1 PB2	PB1 PB2	0	PB1 PB2	PB1 PB2
5	PB1 PB2 PB3 PB4	PB1 PB2	0	PB1 PB2	PB1 PB2 PB3 PB4	PB1 PB2 PB3	PB1 PB2	0	PB1 PB2	PB1 PB2 PB3

Table III

Comparison between cumulative available ore and required ore for B1 ore type

Time period (months)	Required B1 production (Tons)	Available B1 ore type (Tons)
1	250	260 *
2	500	515
3	750	735
4	1000	990
5	1250	1115

Table IV

Comparison between cumulative available ore and required ore for B1 ore type assuming MP3 starts to be mined in time period 2

Time period (Months)	Required B1 production (Tons)	Available B1 ore type (Tons)
1	250	260
2	500	595+
3	750	1050
4	1000	1605
5	1250	2230

tonnage is 735 tons. Because the required B1 production is greater than the available B1 tonnage, time period 2 is an upper bound to the time at which MP3 must start to be mined. To ensure that this is a tight upper bound, we must calculate the available ore given that Machine Placement 3 starts to be mined in time period 2. The result of these calculations appears in Table IV.

For example, the amount of ore available to time period 2 is calculated as the sum of the following quantities: (i) the amount of ore available in time period 1, (ii) the additional ore available from mining production block 2 of machine placements adjacent to MP3, and (iii) the amount of ore in Production Block 1 of MP3. These quantities are, respectively, 260 + (75 + 90 + 80 + 50) + 40 = 595 tons, and are denoted by a $_+$ in Table IV. Following this logic, starting to mine MP3 in time period 2 will make 1050 tons of B1 ore available to time period 3, 1,605 tons of B1 ore available to time period 4 and 2230 tons of B1 ore available to time period 5. By comparing column 2 with column 3, Table IV, we see that the cumulative available ore tonnage given MP3 starts to be mined in time period 2 exceeds the required amount of B1 ore from time period 3 to the end of the horizon. Hence, time period 2 is a tight (exact) upper bound on the late start date for MP3.

Implementation of the algorithms to the Kiruna Mine

The algorithms were developed using AMPL programming language. The size of the MIP model with respect to the number of binary integer variables reduced from 2088 to 496 using both early start and late start algorithms for 36 time periods. These algorithms enable scheduling the mine in less than 100 seconds for 36 time periods. Figure 4 represents the

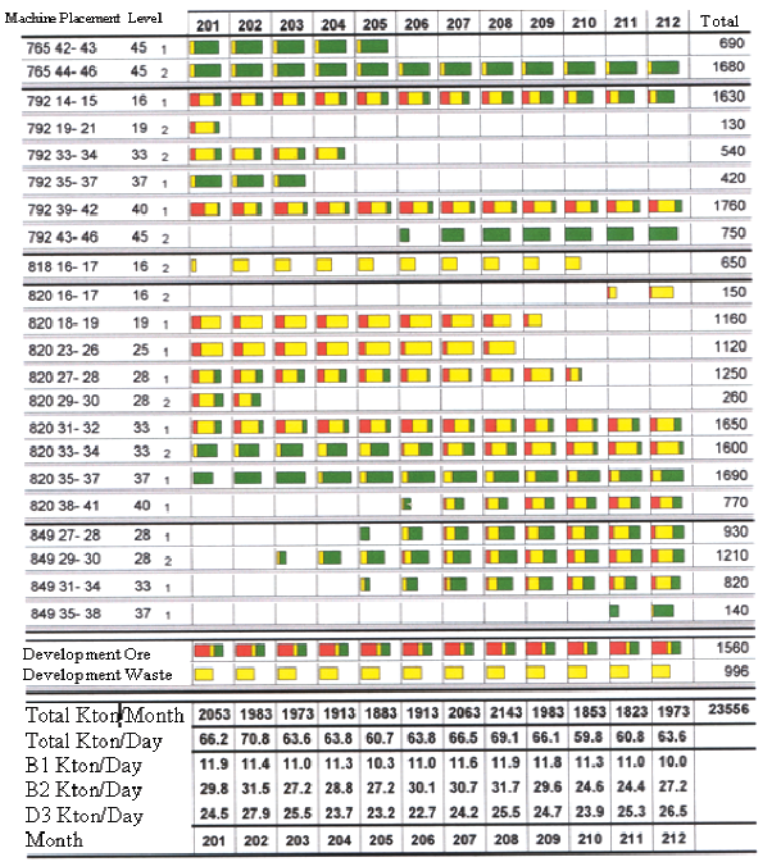


Figure 4—Illustration of first year production scheduling of the Kiruna Mine

first year production schedule for the mine. The different colours represent the three different ore types, the first two columns specify the Machine Placement and the shaft group number, and subsequent columns specify the year and month of the schedule. The total monthly production can be seen at the bottom.

Conclusions

MIP model solution times depend on the number of integer variables and also on the number of constraints in the model. The solution time increases exponentially as the number of integer variables increases. For mining application, the number of binary variables required to formulate production scheduling as an MIP model is usually too great to achieve a solution in a reasonable time. In order to overcome this problem, this paper presents a state-of-art method for reducing the size of the production scheduling model for underground mining operations. Specifically, two new algorithms are developed to determine the earliest and latest start time for each machine placement to reduce the number of integer variables. Before the earliest and after the latest start time for a specific machine placement the binary variable values would be equal to zero in the optimal solution.

It has been shown that these two algorithms considerably reduce the MIP problem size for a sublevel caving operation and enable us to achieve production scheduling for a large-scale underground mining operation such as Kiruna Mine.

Acknowledgements

The author would like to thank LKAB for opportunity to work on this challenging project.

References

Carlyle, M. and Eaves, B.C. Underground planning at Stillwater Mining Company, *Interfaces*, 2001. pp. 50–60.

Kuchta, M., Newman, A., and Topal, E. Implementing a Production Schedule at LKAB's Kiruna Mine, *Interface*, 2004. vol. 34, no. 2, pp. 124–134.

Lerchs, H. and Grossmann, I.F. Optimum Design of open-pitmine, *Transaction on CIM LXVII*, 1965. pp. 47–54.

McIssac, G. Long-term Planning of an Underground Mine Using Mixed-integer Linear Programming, *CIM Bulletin*, 2005. vol. 98.

Ramazan, S. The new fundamental tree algorithm for production scheduling of open pit mines. *European Journal of Operations Research*, 2007. vol. 177, issue 2, pp. 1153–1166.

Ramazan, S. and Dimitrakopoulos, R. Recent applications of operations research and efficient MIP formulations in open pit mining. *SME, transaction*, 2004, vol. 316, pp. 73–78.

Tolwinki, B. Scheduling Production for Open Pit Mines. *27th Applications of Computers and Operations Research in Minerals Industries*, 1998, pp. 168–175.

Topal, E. Advanced Underground Mine Scheduling Using Mixed Integer Programming, PhD dissertation, Colorado School of Mines, Golden , Colorado, 2003.

Topal, E., Newman, A., and Kuchta, M. Extensions to an Efficient Optimization Model for Long-Term Production Planning at LKAB's Kiruna Mine. *Applications of Computers and Operations Research in Minerals Industries*, South African Institute of Mining and Metallurgy, 2003.

Trout, L.P. Underground Mine Production Scheduling Using Mixed Integer Programming', *25th International APCOM Symposium Proceedings*, Brisbane, Australia, 1995, pp. 395–400.

MIP model from sublevel caving operation

Appendix I

A. Indices

a	machine placement
b,b'	production block
k	ore type, i.e., B1, B2, D3
t	time period (month)
v	shaft group, i.e., 1…10

B. Sets

Ω_b =	set of eligible time periods in which production block b can be mined (restricted by production block location and the start time of other relevant production block)
A_a =	set of production blocks within machine placement a
$BlockV_b$ =	set of production blocks whose access is restricted vertically by production block b
$BlockR_b$ =	set of production blocks whose access is restricted by right adjacency to production block b
$BlockL_b$ =	set of production blocks whose access is restricted by left adjacency to production block b
$BlockS_v$ =	set of machine placements within a shaft group v

C. Parameters

r_{bk} =	amount of ore type k, in production block b (tons)
d_{kt} =	demand for ore type k in time period t (tons)
$Early_b$ =	earliest start time for production block b
$Late_b$ =	latest start time at which production block b
T =	length of the planning horizon
LHD_v =	the maximum number of simultaneously operational LHDs in each shaft group v

$$P_{abv} = \begin{cases} 1, \text{ If block } b \text{ of machine placement } a \text{ is in shaft group } v \\ 0, \text{ othewwise} \end{cases}$$

D. Decision variables

$$y_{bt} = \begin{cases} 1, \text{ If start mining production block } b \text{ in time period } t \\ 0, \text{ othewwise} \end{cases}$$

du_{kt} =	amount mined above the desired demand of ore type k in time period t (tons)
dd_{kt} =	amount mined below the desired demand of ore type k in time period t (tons)

E. Formulation

Objective Function:

$$\text{Min} \sum_{k,t} (du_{kt} + dd_{kt})$$

Subject to:

$$\sum_{b} r_{bk} * y_{bt} - du_{tk} + dd_{tk} = d_{tk} \quad \forall k,t \in \Omega_b \qquad [1]$$

$$\sum_{t\in\Omega_b} y_{bt} \geq y_{b't'} \quad \forall b,b' \in Block\,\mathrm{V}_b, t' \in \Omega_{b'} \qquad [2]$$

$$\sum_{t\in\Omega_b} y_{bt} \geq y_{b't'} \quad \forall b,b' \in Block\,\mathrm{R}_b, t' \in \Omega_{b'} \qquad [3]$$

$$\sum_{t\in\Omega_b} y_{bt} \geq y_{b't'} \quad \forall b,b' \in Block\,\mathrm{L}_b, t' \in \Omega_{b'} \qquad [4]$$

$$\sum_{a\in S_v} \sum_{b\in Aa} \sum_{u\in\Omega_b | u\leq t} P_{abv} * y_{bu} \leq LHD_v \quad \forall v \qquad [5]$$

$$y_{bt} = 1 \text{ for } \forall b \mid Early_b = Late_{ab} \qquad [6]$$

$$\sum_t y_{bt} \leq 1 \text{ for } \forall b \mid Late_b > T \qquad [7]$$

$$\sum_t y_{bt} = 1 \text{ for } \forall b \mid Late_b \leq T \qquad [8]$$

$$du_{tk}, dd_{tk} \geq 0 \quad \forall t,k$$
$$y_{bt}\ binary \quad \forall b,t \qquad [9]$$

The objective function minimizes the deviation from the production targets for each ore type (B1, B2 and D3) so that the mills can meet their respective production demands. Constraints [1] calculates the tons of B1, B2, and D3 ore mined per time period and the corresponding deviations from the specified production levels. Constraints [2] comprises the vertical sequencing constraints between mining sublevels.

Constraints [3] and [4] enforce horizontal sequencing constraints between adjacent production blocks. Constraints [5] ensures that no more than the allowable number of LHDs is active within a shaft group. Constraints [6] places the production blocks that are currently being mined into the production schedule. Constraints [7] sets packing constraints, and ensures that a production block starts to be mined no more than once during the time horizon if its late start date occurs beyond the length of the time horizon. Constraints [8] sets partitioning constraints, and requires that a production block starts being mined at some point during the time horizon if its late start date falls within the time horizon. Constraints [9] enforces non-negativity and integrality of the variables, as appropriate. ◆

