Comparing New and Traditional Methodologies for Production Scheduling in Open Pit Mining

Marcos Goycoolea
School of Business, Universidad Adolfo Ibañez, Chile

Daniel Espinoza
Department of Industrial Engineering, Universidad de Chile, Santiago, Chile

Eduardo Moreno
Faculty of Engineering and Sciences, Universidad Adolfo Ibañez, Chile

Orlando Rivera
Industrial Engineering and Operations Research, Universidad Adolfo Ibañez, Chile

ABSTRACT: Strategic open pit mine planning methodologies have traditionally broken up the production scheduling process into two steps. First, push-backs (or phases) are determined. Second, bench-phases are scheduled in time by defining which extracted material should be treated as ore, and which as waste, and deciding how extracted ore should be processed. Traditionally, this last step has been carried out using heuristics. In this article we describe a mixed integer programming (MIP) formulation with which this last step can be optimized, and compare it to Whittle's Milawa algorithm. Our results confirm that MIP techniques for this problem have come to the point where they can solve real-sized problem instances, computing solutions that are better both in terms of objective function and modelling detail.

INTRODUCTION

In the early 1960s, Thys Johnson [7] proposed a mathematical programming model for solving the strategic open pit mine planning problem. Solving it has proved computationally daunting, however: the economic block models used to describe ore bodies are typically made up of millions of blocks, and production schedules usually span decades, resulting in complex problem instances having hundreds of millions of variables. Since Johnson's early work a number of authors have built on his model and methodology, with important advances in recent years (See for example, [4, 3, 1, 2, 5]). Despite this progress, mathematical programming approaches have faced two main difficulties. On the one hand, though the size of problems that have been tackled has greatly increased, most approaches still need to simplify Johnson's model in one way or another. Most consider only a single destination per block, or a limited number of constraints. On the other hand, it is not clear how the solutions produced by pure mathematical programming approaches can be made feasible in practice. This is due to the complex spatial constraints that need to be imposed in order ensure minimum width requirements for equipment to move and operate in the mine. Solutions produced by these approaches tend to scatter the extraction of small numbers of blocks throughout the deposit, making solutions unrealistic.

In order to circumvent these problems in practice, mine planners have instead relied on an approach known as the Nest-Pit method. This is a heuristic technique that greedily breaks down the problem into two main steps: First, it determines a set of push-packs (or phases) to be used as guides to construct a production schedule. Second, it schedules the blocks in each push-back, bench by bench. In this approach, blocks are not extracted individually, but rather, blocks in a same bench-phase are concurrently extracted. The Nested-Pit method is the natural result of two seminal papers from the early 1960s: That of Lerchs and Grossman [10], who proposed a technique for creating a sequence of nested pits used to guide the creation of push-backs; and that of Lane [8, 9], who proposed a dynamic programming technique for scheduling the timing of the bench-phase extraction, and the destination of the individual blocks once extracted. To our knowledge, all commercial software packages for strategic mine planning use the Nested-Pit method.

In this article we propose a MIP model for dealing with this last step of the Nested-Pit method. With the exception of Lane's algorithm and its variants, which both simplify the problem and are not known to be exact optimization algorithms, there is little if any literature on solving this problem to optimality. In fact, this problem is generally assumed to be intractable. For example, the user's manual of the NPV Scheduler software package warns its readers that "General NPV maximization problem is intractable for real mines. The methods that solve this problem are applicable to academic examples with less than 100,000 blocks and uniform slopes, and even then the computation time is counted in days." (OES Overview, Maintenance Release 21 (SP 3)). The MIP model that we propose is different to others that have been proposed in the literature in that we work with a pre-defined set of Push-backs. In addition, this study is the first to compare the results obtained by this model with those obtained with a major commercial software package on exactly the same instances. Our goal is not only to produce solutions with higher (and provably optimal) values, but also, to show that mathematical programming makes it easy to impose additional constraints that are important to model for operational reasons.

NOTATION

Let B represent the set of blocks, and D the destinations to which blocks can be sent. We assume destinations can be waste-dumps or processing-locations (mills, leech-pads, etc.), but not stockpiles. Let T = {1,...,T} represent the time periods. For each $b \in$B, $d \in$D, and $t \in$T, let $p^t{}_{b,d}$ represent the estimated profit obtained by sending b to d in time t. For each $b \in$B let q_b be its tonnage.

Let P = P1,...,Pn{P_1,...,P_n} represent the set of pre-defined push-backs. Assume that these pushbacks are disjoint sets of blocks. In practical applications, these pushbacks will typically be obtained by taking differences between sets of nested pits computed by a parametric run of the Lerchs Grossman algorithm. Assume that benches are numbered from 1 to H, with bench 1 being the topmost bench. For each $h \in \{1,...,H\}$, let B(h) represent the blocks in bench h. Given $p \in \{1,...,n\}$ and $h \in \{1,...,H\}$ we say that I = B(h) $\cap$ P_p is an increment. Let I index all of the increments. For each $i \in$I let h_i and p_i represent the index of the corresponding bench and push-back, respectively, and let q_i represent the total tons of increment i.

Let A$\subseteq$I×I represent a precedence relationships between increments. A will be used to model sequencing requirements by imposing that if $(i_1,i_2) \in$A, then increment i_2 must be extracted no later than increment i_1. Define A as follows: If i_1 and i_2 are non-empty, then (i_1,i_2) $\in$A if either h_{i1}

$$\max \sum_{b\in\mathcal{B}} \sum_{d\in\mathcal{D}} \sum_{t\in\mathcal{T}} p^t_{b,d} y_{b,d,t} \tag{1a}$$

s.t.

$$x_{i,t} = \sum_{d\in D} y_{b,d,t} \qquad \text{for all } i\in\mathcal{I},\ b\in\mathcal{B}(i),\ t\in\mathcal{T} \tag{1b}$$

$$\sum_{t\in\mathcal{T}} x_{i,t} \le 1 \qquad \text{for all } i\in\mathcal{I} \tag{1c}$$

$$\sum_{t=1}^{\tau} x_{i_1,t} \le \sum_{t=1}^{\tau} x_{i_2,t} \qquad \text{for all } (i_1,i_2)\in\mathcal{A},\ \tau\in\mathcal{T} \tag{1d}$$

$$\sum_{b\in\mathcal{B}} \sum_{d\in\mathcal{D}} q_b y_{b,d,t} \le M_t \qquad \text{for all } t\in\mathcal{T} \tag{1e}$$

$$\sum_{b\in\mathcal{B}} q_b y_{b,d,t} \le U^d_t \qquad \text{for all } d\in\mathcal{D},\ t\in\mathcal{T} \tag{1f}$$

$$\sum_{i\in\mathcal{I}} q_i x_{i,t+1} \le (1+\alpha) \sum_{i\in\mathcal{I}} q_i x_{i,t} \qquad \text{for all } t=1,\ldots,T-1 \tag{1g}$$

$$\sum_{i\in\mathcal{I}} q_i x_{i,t+1} \ge (1-\beta) \sum_{i\in\mathcal{I}} q_i x_{i,t} \qquad \text{for all } t=1,\ldots,T-1 \tag{1h}$$

$$\sum_{b\in\mathcal{B}} q_b y_{b,d,t+1} \le (1+\alpha) \sum_{b\in\mathcal{B}} q_b y_{b,d,t} \qquad \text{for all } t=1,\ldots,T-1,\ d\in\mathcal{D}^* \tag{1i}$$

$$\sum_{b\in\mathcal{B}} q_b y_{b,d,t+1} \ge (1-\beta) \sum_{b\in\mathcal{B}} q_b y_{b,d,t} \qquad \text{for all } t=1,\ldots,T-1, d\in\mathcal{D}^* \tag{1j}$$

$$x_{i,t}, y_{b,d,t} \ge 0 \qquad \text{for all } i\in\mathcal{I}, b\in\mathcal{B}, d\in\mathcal{D}, t\in\mathcal{T} \tag{1k}$$

$$x_{i_1,t} > 0 \text{ implies } \sum_{k=1}^{t} x_{i_2,k} = 1 \qquad \text{for all } (i_1,i_2)\in\mathcal{A},\ \forall t\in\mathcal{T} \tag{1l}$$

Figure 1. Mathematical programming formulation of the open pit mine production scheduling (OPMPS) problem

$< h_{i2}$, or, if $h_{i1} = h_{i2}$ and $p_{i1} < p_{i2}$. That is, increments must be extracted from top to bottom, and then in the order prescribed by the push-backs.

For each t∈T let M_t represent the mining capacity and U_t^d the processing capacity, of destination d∈D at time t∈T. We assume capacities are expressed in tons.

A MIXED INTEGER PROGRAMMING (MIP) MODEL

Let variables $x_{i,t}$ indicate the proportion of increment i∈I extracted in time t∈T. For each b∈B; d∈D and t∈T let variable $y_{b,d,t}$ indicate the proportion of block b sent to destination d in time t. In Figure 1 we present a mathematical programming formulation of the production scheduling problem, given a pre-defined set of push-backs:

Note that model OPMPS is an extension of a model recently proposed by Bienstock and Zuckerberg. In formulation (OPMPS) the objective function (1a) maximizes solution NPV.

Table 1. Description of the different instances of the test set used in our computational study

Instance	Blocks	Increments	Periods	Annual Discount Rate
Marvin	53,271	56	20	0.10
Guallatari	2,000,000	322	60	0.05
Palomo	1,000,000	67	50	0.10
McLaughlin	2,140,342	173	20	0.15
Chaiten	300,000	266	25	0.10
Tacora	4,000,000	233	60	0.10

Constraints (1b) impose consistency between the x and y variables, ensuring that all blocks in a same increment are extracted in the same proportion in all time periods. Constraints (1c) impose the total amount extracted of each increment during the considered time horizon does not exceed the available material in the increment. Constraints (1d) impose the precedence relationships. Constraints (1e) impose upper bounds on the total tonnage that can be extracted per time period. Constraints (1f) impose upper bounds on the total tonnage that can be sent to each destination per time period. Constraints (1g) and (1h), and (1i) and (1j), known as volume-flow constraints, impose a limit on how much extraction and production can vary from period to period. This constraint is parameterized by $-1<\alpha,\beta<1$, and $D^*\subseteq D$. Constraints (1k) impose non-negativity of the variables. Constraints (1l) impose the integrality conditions on the x variables. That is, they impose that in order for a block to be partially (or totally) extracted, all of its predecessors must be totally extracted. If an OPMPS solution only satisfies constraints (1l) on pairs of increments $(i_1,i_2)\in A$ such that $h_{i1}\neq h_{i2}$ we say that the solution is *partially-integral*.

COMPUTATIONAL ANALYSIS

In this study we consider six strategic mine planning instances: Marvin, McLaughlin, Chaiten, Guallatari, Tacora and Palomo. Two of these instances, Marvin and McLaughlin, can be downloaded from the MineLib website [6]. The others are anonymous data sets provided by our industry research partners. Statistics pertaining to all instances are described in Table 1. The names of the instances provided by our partners, and some basic statistics, have been changed for confidentiality reasons. All instances have but two destinations: mill and waste-dump. Our analyses were made with Dassault's Whittle 4.4.0, and with OMP. OMP, which stands for Open Mine Planner, is an academic production scheduling software developed by the authors of this article that includes a customized mixed integer programming solver for OPMPS. This solver uses the Bienstock-Zuckerberg algorithm [1,2] to solve linear programming relaxations of OPSP at the root and in each node of a branch-and-bound algorithm. This branch-and-bound algorithm is combined with the TopoSort heuristic presented in [5].

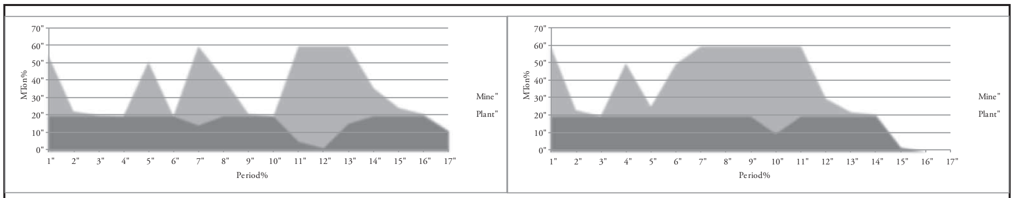
To create our problem instances we applied the following procedure on each data-set using Whittle. We created 36 pitshells with revenue factors ranging from 0.3 to 1.0. For each data set we created an operational scenario (Cashflow ore selection method) with four push-backs (automatic push-back definition using Milawa NPV and the "fast" calculation mode).

For each problem instance we produced two production schedules with the Whittle Cut-off Optimization tool, starting from our pre-defined push-backs. The first solution was generated with the Milawa NPV scheduling algorithm, and the second with Milawa Balanced. In these runs we did not use the maximum lead, minimum lead and maximum benches per period constraints. From our runs we found that Milawa NPV produces feasible OPMPS solutions that ignore volume flow constraints (1g)–(1j). It is not exactly clear what Milawa Balanced attempts to do. The reference manual simply states that the algorithm attempts to find a solution with "an improved throuput balance," which it does by seeking to "maximise the usage of production facilitites early in the life of the mine instead of maximising NPV". This suggests that it ignores constraints (1g), (1h), (1j), and imposes (1i) with α = 0.0 for the processing destination (i.e., not for waste). Our computational experiments show that that Milawa Balanced produces solutions of OPMPS that are partially integral.

For each problem instance we also produced two production schedules with OMP. The first solution ignores constraints (1g) and (1h). We called this the OMP NPV solution. Since the user's manual is not precise about what Milawa Balanced does, we opted to be more conservative and to impose (1g)–(1j) with α =0, β=1. By setting these parameters in this way we hope to achieve solutions that are balanced, as defined by the Whittle User's Manual. For this we defined D^* as the singleton set containing only the mine processing destination. This imposes that the tons extracted and sent to processing must decreasing over time. We call this the OMP Balanced solution. All instances were ran for 72 hours each, and the best solution obtained in that time was recorded. All instances of Marvin and McLaughling were solved to optimality within hours. The remaining instances did not finish solving in the allotted time. However, the OMP solver obtained good solutions (< 3% of optimality) for all instances except Palomo in less than ten minutes each.

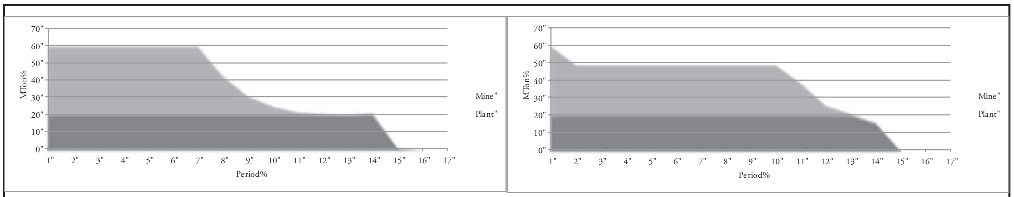
To begin we plot in Figures 2–5 the tonnage graphs corresponding to the solutions obtained for the Marvin and Palomo instances. These graphs illustrate the total number of tons sent in each problem to each destination (processing plant or waste dump). As can be seen in the graphs, all of the NPV solutions have horrible throughput balances. For example, after year 10 both the Milawa NPV and OMP NPV solutions find that it is optimal to not saturate the processing plant capacity in order to extract more material. Milawa Balanced effectively manages to build solutions that produce in a non-increasing rate. OMP Balanced solutions, in addition, manage to effectively impose a non-increasing extraction rate to the solutions. It is interesting to note that in all of the OMP Balanced solutions, the waste extracted in each period is also non-increasing, even though this is not explicitly required by the corresponding OPMPS formulation. This is a natural consequence of imposing that the total extracted and processed tons are non-increasing, but there could be instances where this is not observed. It is also interesting that the OMP solutions tend to span fewer years than the corresponding Milawa solutions. Moreover, in the Palomo Data Set it is remarkable that the OMP Balanced solution can produce a solution with such higher value, spanning such a shorter amount of time and moving so much less material. Moreover, it is surprising how much the number of tons extracted varies from year to year in the Whittle Balanced solution. This makes the Milawa Balanced solution of this problem very impractical.

The NPV of the solutions obtained using Whittle and OMP are presented in Table 2. The solution values are normalized with respect to the best known-upper bound of the solution value. This makes it easier to see how close solutions are to optimality, and to make comparisons. In this table we also present the ratio of the best-known NPV and Balanced solution values. As can be seen, OMP NPV is able to obtain solutions within 3% of optimality in all instances. Milawa NPV also does very well, failing only in Guallatari and Palomo, where it finds solution with a gap of approximately 15% each. OMP Balanced also manages to obtain high-valued solutions, all within 6% of optimality. Milawa Balanced again has great difficulty with Guallatari and Palomo. Milawa



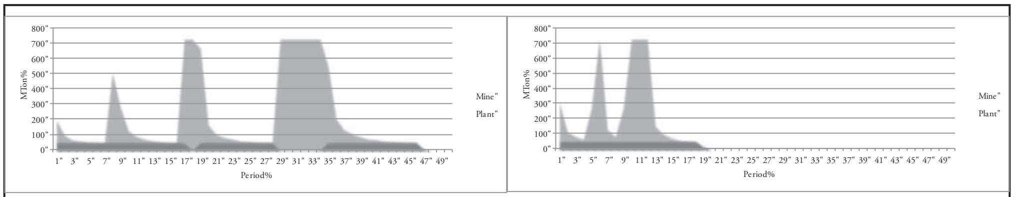
(a) Milawa NPV (b) OMP NPV

Figure 2. Tonnage graphs for NPV solutions of the Marvin data set



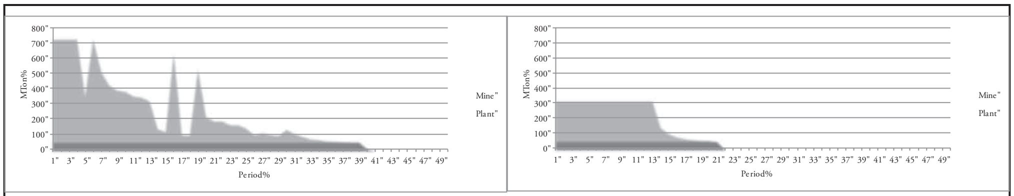
(a) Milawa Balanced (b) OMP Balanced

Figure 3. Tonnage graphs for balanced solutions of the Marvin data set



(a) Milawa NPV (b) OMP NPV

Figure 4. Tonnage graphs for NPV solutions of the Palomo data set



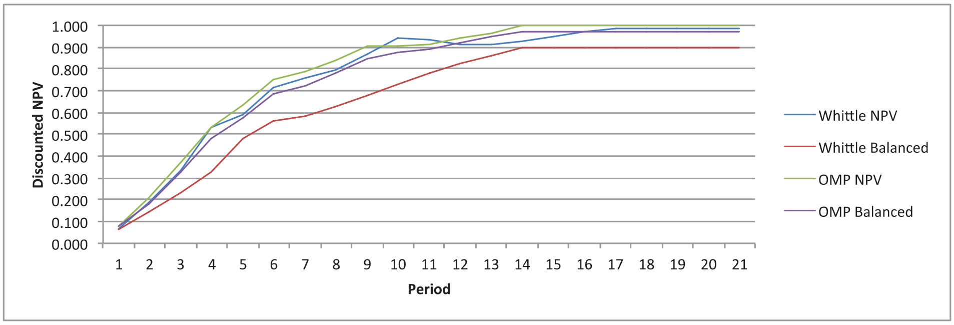
(a) Milawa Balanced (b) OMP Balanced

Figure 5. Tonnage graphs for balanced solutions of the Palomo data set

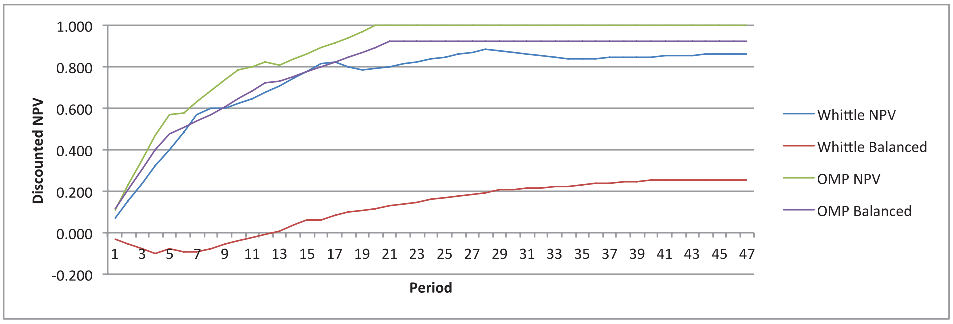
Balanced was not used to solve McLaughlin because, for some reason unknown to us, the algorithm requires an upper bound on the production tonnage, which we did not have for this instance. It is interesting to note that in Chaiten, Milawa obtained a Balanced solution with objective function value higher than that of OMP. It should be noted that this solution is only partially feasible, but it is still a very good result. This table highlights the value of using exact optimization to solve these instances. Since Whittle does not provide bounds on the quality of its solutions, one might

Table 1. NPV of the solutions obtained

	Whittle Milawa			OMP		
Instance	**NPV**	**Balanced**	**Balanced/NPV**	**NPV**	**Balanced**	**Balanced/NPV**
Marvin	98.61	91.78	89.95	100.00	99.47	97.38
Guallatari	83.76	63.79	64.74	98.27	96.72	98.15
Palomo	85.60	25.89	25.03	99.24	96.19	93.00
McLaughlin	99.73	—	—	100.00	99.98	98.89
Chaiten	97.61	98.99	99.35	99.62	94.22	94.57
Tacora	96.96	93.35	95.69	97.26	94.62	97.00
Geo. Mean	93.48	67.49	67.34	99.06	96.84	96.48



(a) Marvin



(b) Palomo

Figure 6. PV graphs of the Marvin and Palomo data sets

think that the solutions that Milawa obtains are the best possible. However, not only does OMP obtain significantly better balanced solutions, but also, the OMP Balanced/NPV ratio shows that it is possible to obtain balanced solutions that achieve nearly the same value as their unbalanced counterparts.

A graphical analysis is presented in Figure 6, where we plot the accumulated NPV of the solutions over time. We rescale the Balanced and NPV solutions so that they can compared against each other. As can be seen in the graphs, the OMP solutions obtain significantly higher values during the first few years of the mining operation than the corresponding Milawa solutions. This makes them financially much more attractive. In addition, it can be seen that on a year-by-year basis the OMP Balanced solutions are very competitive the Whittle NPV solutions.

The computational analyses in this paper highlight the value, both in terms of objective function and in modelling detail, that can be achieved by using Mixed Integer Programming approaches over traditional heuristics. Though the case studies that we consider are simple, it is not difficult to see that OPMPS can be even further enriched by adding additional constraints. Examples might be more advanced sequencing rules that impose a maximum number of open phases, or minimum/maximum lags between benches, blending requirements, use of stockpiles, or yearly production targets as defined by the user.

ACKNOWLEDGMENTS

This work was partially funded by the government of Chile through FONDECYT grant 110674 (M.G., D.E. and O.R.).

REFERENCES

[1] D. Bienstock and M. Zuckerberg. A new lp algorithm for precedence constrained production scheduling. Optimization Online, 2009.
[2] D. Bienstock and M. Zuckerberg. Solving LP relaxations of large-scale precedence constrained problems. Proceedings from the 14th conference on Integer Programming and Combinatorial Optimization (IPCO). Lecture Notes in Computer Science 6080, pages 1–14, 2010.
[3] N. Boland, I. Dumitrescu, G.Froyland, and A.M. Gleixner. LP-based disaggregation approaches to solving the open pit mining production scheduling problem with block processing selectivity. Computers and Operations Research, 36:1064–1089, 2009.
[4] L. Caccetta and S.P. Hill. An application of branch and cut to open pit mine scheduling. Journal of Global Optimization, 27:349–365, 2003.
[5] R. Chicoisne, D. Espinoza, M. Goycoolea, E. Moreno, and E. Rubio. A new algorithm for the open-pit mine production scheduling problem. Operations Research, 60(3):517–528, 2012.
[6] D. Espinoza, M. Goycoolea, E. Moreno, and A.M. Newman. Minelib: a library of open pit mining problems. Annals of Operations Research, pages 1–22, 2012.
[7] T.B. Johnson. Optimum open pit mine production scheduling. PhD thesis, Operations Research Department, Universiry of California, Berkeley, May 1968.
[8] K.F. Lane. Choosing the optimum cutoff grade. Colorado school of mines quarterly, 59(4):811–829, 1964.
[9] K.F. Lane. The economic definition of ore: cutoff grade in theory and practice. Mining Journal Books Limited, London, 1988.
[10] H. Lerchs and I.F. Grossmann. Optimum design of open-pit mines. Transactions, LXVIII:17–24, 1965.