**Les Cahiers du GERAD** ISSN: 0711–2440

# Progressive hedging applied as a metaheuristic to schedule production in open-pit mines accounting for reserve uncertainty

A. Lamghari
R. Dimitrakopoulos

G–2014–40

June 2014

Les textes publiés dans la série des rapports de recherche *Les Cahiers du GERAD* n'engagent que la responsabilité de leurs auteurs.

La publication de ces rapports de recherche est rendue possible grâce au soutien de HEC Montréal, Polytechnique Montréal, Université McGill, Université du Québec à Montréal, ainsi que du Fonds de recherche du Québec – Nature et technologies.

Dépôt légal – Bibliothèque et Archives nationales du Québec, 2014.

The authors are exclusively responsible for the content of their research papers published in the series *Les Cahiers du GERAD*.

The publication of these research reports is made possible thanks to the support of HEC Montréal, Polytechnique Montréal, McGill University, Université du Québec à Montréal, as well as the Fonds de recherche du Québec – Nature et technologies.

Legal deposit – Bibliothèque et Archives nationales du Québec, 2014.

**GERAD** HEC Montréal
3000, chemin de la Côte-Sainte-Catherine
Montréal (Québec) Canada H3T 2A7

**Tél. : 514 340-6053**
Téléc. : 514 340-5665
info@gerad.ca
www.gerad.ca

# Progressive hedging applied as a metaheuristic to schedule production in open-pit mines accounting for reserve uncertainty

**Amina Lamghari**

**Roussos Dimitrakopoulos**

*GERAD & COSMO – Stochastic Mine Planning Laboratory, Department of Mining and Materials Engineering, McGill University, Montréal (Québec) Canada, H3A 2A7*

`amina.lamghari@mcgill.ca`
`roussos.dimitrakopoulos@mcgill.ca`

**June 2014**

**Les Cahiers du GERAD**

**G–2014–40**

Copyright © 2014 GERAD

**Abstract:** Scheduling production in open-pit mines is characterized by uncertainty about the metal content of the orebody (the reserve) and leads to a complex large-scale mixed-integer stochastic optimization problem. In this paper, a two-phase solution approach based on Rockafellar and Wets' progressive hedging algorithm (PH) is proposed. PH is used in phase I where the problem is first decomposed by partitioning the set of scenarios modeling metal uncertainty into groups, and then the sub-problems associated with each group are solved iteratively to drive their solutions to a common solution. In phase II, a strategy exploiting information obtained during the PH iterations and the structure of the problem under study is used to reduce the size of the original problem, and the resulting smaller problem is solved using a sliding time window heuristic based on a fix-and-optimize scheme. Numerical results show that this approach is efficient in finding near-optimal solutions and that it outperforms existing heuristics for the problem under study.

**Key Words:** Open-pit mine production scheduling, strategic planning, progressive hedging method, Lagrangean relaxation, sliding time window heuristic.

## 1 Introduction

Scheduling production for open-pit mines is an important and critical issue in surface mine planning as it determines the raw materials to be produced yearly over the life of the mine and can have a huge impact on the economic value of a mining operation. In formulating the mine production scheduling problem (MPSP), the mineral deposit is typically represented as a three-dimensional array of blocks, each of which represents a volume of material that can be mined and then sent either to a processing facility to recover the metal it contains, to a stockpile for possible future processing, or to a waste dump. Each block has a weight, a metal content, and an economic value. Blocks with a metal content above a certain cut-off grade are referred to as ore blocks and are to be processed, while those whose metal content is below the cut-off grade are waste. The metal content is estimated using information obtained from drilling, while the economic value represents the value of the metal recovered less the mining, the processing, and the selling costs. MPSP seeks to determine the mining sequence of the blocks; i.e., deciding which blocks to mine in each period of the life of the mine, so that the highest possible net present value of the mine is achieved. There are constraints specifying physical and operational limitations that must be taken into account. In basic terms, a block can be mined at most once (reserve constraints) and only after the blocks overlying it have been mined (slope constraints). Both extraction equipment and processing facilities have capacities that cannot be exceeded in any period (mining and processing constraints, respectively).

Metal uncertainty (also referred to as reserve uncertainty) is an inherent part of mine production scheduling, as the metal content required for the decision-making is known only from limited drilling information. Ignoring metal uncertainty; that is, solving a deterministic model using a single estimate for the blocks' metal content, can lead to incorrect assessments of the the production forecasts, the final pit limits, and the net present value (Ravenscroft, 1992; Dowd, 1994; Dimitrakopoulos et al., 2002; Menabde et al., 2007; Albor and Dimitrakopoulos, 2010; Asad and Dimitrakopoulos, 2013; Marcotte and Caron, 2013). One should therefore consider metal uncertainty in the scheduling process, which gives rise to the stochastic MPSP considered in this paper.

Most of the literature on the stochastic MPSP has been devoted to designing heuristic-based solution methods, as exact methods are only effective for small problem instances with a few thousand blocks (Boland et al., 2008; Ramazan and Dimitrakopoulos, 2013), while realistic instances involves typically tens to hundreds of thousands blocks. Early work proposed a simulated annealing algorithm (Godoy, 2002). A three-phase constructive approach that involves generating nested pits, grouping them in mining phases, and generating a schedule based on these phases has been developed by Albor and Dimitrakopoulos (2010). The method used in Lamghari and Dimitrakopoulos (2012) is based on Tabu search, and hybrid approaches embedding variable neighborhood descent and a very large-scale neighborhood search mechanism have been developed in Lamghari et al. (2013) and Lamghari and Dimitrakopoulos (2013), respectively.

In this paper, we propose a novel heuristic-based method to efficiently address the stochastic MPSP. It is based on the progressive hedging algorithm (PH) introduced by Rockafellar and Wets (1991) for optimization problems under uncertainty. In PH, uncertainty is modelled by a limited number of scenarios that reflect the information available about the uncertain parameters of the problem under study (i.e., metal uncertainty in the case of this paper). Each scenario has a known probability of occurrence and provides possible values of the uncertain parameters. PH decomposes the problem according to the scenarios and solves the sub-problem for each scenario separately. This step provides multiple solutions that are tailored to the scenarios and thus might be different from each other. Each solution indicates what should be done if a specific scenario is realized. But, what is needed is an *implementable* solution; that is, a non scenario-specific solution that will be feasible and can be implemented regardless of which scenario is realized and that is also well-hedged against uncertainty. The essence of the PH algorithm is to introduce *implementability* constraints to drive the solutions of the sub-problems towards a single *implementable* solution. Such constraints are relaxed and dualized in the objective function; that is, a penalty term measuring their violation is added to each sub-problem's objective function to minimize the differences between the scenario-solution (the sub-problem solution) and an estimation of the implementable solution. PH iterates between updating the estimation of the implementable solution, adjusting the penalties, and solving the sub-problems until an implementable solution is obtained. At this point, the method is said to have converged, and for continuous stochastic problems, it converges to a global optimum.

A limitation of PH is that convergence is guaranteed only in the convex case. When applied to mixed-integer stochastic problems, such as the one at hand, convergence might be difficult to obtain and, if PH converges, it converges to a local optimum. The approach proposed in Løkketangen and Woodruff (1996) to alleviate the convergence problem proceeds in 2 phases. Phase I applies PH, but with a different stopping criterion. Rather than waiting for full convergence, PH terminates when a fixed number of iterations or a fixed amount of CPU time is reached. The variables that have converged to that point (i.e., those that have the same values in all sub-problem's solutions) are fixed in the original problem. One thus obtains a smaller mixed-integer stochastic problem, referred to as the restricted problem, which is solved in phase II to generate an implementable solution. Because the restricted problem is of reduced size, it won't take as much time to solve as the original problem would.

This approach was successfully used to solve stochastic lot-sizing problems (Haugen et al., 2001) and stochastic network design problems (Crainic et al., 2011). However, applying it in the context of the stochastic MPSP addressed in this paper is computationally expensive because the sub-problems associated with the scenarios involve a large number of binary variables and are themselves difficult to solve. Moreover, preliminary tests showed that although the restricted problem is of smaller size compared to the original problem, it remains a large-scale problem and requires long computational times if solved with an exact method, as is typically done in the literature. To overcome these difficulties, we made the following refinements to improve the performance of the approach proposed in Løkketangen and Woodruff (1996):

- At phase I, instead of having each sub-problem associated with a single scenario, the sub-problems are comprised of multiple scenarios. Grouping scenarios and solving multi-scenario sub-problems not only will reduce the number of sub-problems to be solved at each iteration of phase I (PH), and hence the time required to complete phase I, but it should also result in faster convergence. Similar ideas were recently used by Crainic et al. (2014) in the context of stochastic network design. While Crainic et al. (2014) use different strategies to group the scenarios, we use here a random strategy.
- We take advantage of the structure of the problem to better exploit the information that becomes available during the iterations of PH and fix additional variables (in addition to those that have converged). This substantially reduces the size of the restricted problem to be solved during phase II, and consequently the computational time required to solve it.
- Rather than solving the restricted problem using an exact method, we use an efficient heuristic; namely, a sliding time window heuristic (STWH) based on a fix-and-optimize scheme.

We provide numerical results to evaluate the efficiency of the proposed solution approach as well as an analysis that shows the advantages and disadvantages of increasing the size of the groups of the scenarios during the first phase of the algorithm. We also compare the proposed solution approach to the sequential heuristic proposed in Lamghari et al. (2013), which is based on a time-decomposition strategy rather than a scenario-decomposition strategy. The results indicate that although it requires longer computational times, the method proposed in this paper finds solutions that are 48% to 71% better than those generated by the sequential heuristic in Lamghari et al. (2013). We also compare the proposed solution approach to an approach where STWH is used alone. In this comparison, it is found that combining PH and STWH leads to a substantial reduction of the computational times by a factor of 154 to 243 (71 to 112 minutes compared to 12 days), while in terms of solution quality, the two methods are comparable.

Throughout the paper, we consider the "classical" and basic variant of the MPSP, where the cut-off grade determining whether an extracted block is to be processed or sent to the waste dump is fixed, and only reserve, slope, mining, and processing requirements are taken into account. Additional requirements may be added via constraints or as penalties in the objective function to reflect other concerns such as variable cut-off grade, grade blending, or stockpiling. This leads to different variants of the stochastic MPSP; however, the algorithm proposed in this paper is a general-purpose algorithm and should be applicable to any of these variants.

In the next section, a mathematical formulation of the stochastic MPSP considered in this paper is given. The solution procedure based on the progressive hedging strategy is described in Section 3. Numerical results are reported in Section 4. The paper concludes with Section 5, where directions for future research are summarized.

## 2 Mathematical formulation

The stochastic MPSP described in the previous section is formulated as a two-stage stochastic programming model (Birge and Louveaux, 1997). This model is described in detail in Lamghari and Dimitrakopoulos (2012), and we only briefly recall it here. The following notation is used:

### Sets and indices:

$T$: Set of time periods over which blocks are being scheduled, indexed by $t = 1, \dots, h$.

$N$: Set of blocks considered for scheduling, indexed by $i = 1, \dots, n$.

$P_i$: Set of predecessors of block $i$; i.e., blocks that have to be mined to have access to block $i$, indexed by $p$.

$\Omega$: Set of scenarios used to model metal uncertainty, indexed by $s = 1, \dots, S$.

### Variables:

$$x_i^t = \begin{cases} 1 & \text{if } i \text{ is mined during period } t, \\ 0 & \text{otherwise.} \end{cases}$$

$d_s^t$: Surplus in ore production during period $t$ under scenario $s$.

### Parameters:

$W^t$: Maximum amount of rock (ore and waste) that can be mined during period $t$ (mining capacity in tonnes).

$\Theta^t$: Maximum amount of ore that can be processed during period $t$ (processing capacity in tonnes).

$w_i$: Weight of block $i$ in tonnes (tonnage).

$$\theta_{is} = \begin{cases} 1 & \text{if } i \text{ is ore under scenario } s, \\ 0 & \text{otherwise.} \end{cases}$$

$v_{is}^t = \frac{v_{is}}{(1+d_1)^t}$: Discounted economic value of block $i$ if mined and processed during period $t$, and if scenario $s$ occurs ($v_{is}$ being the undiscounted economic value and $d_1$ the discount rate per period).

$c^t = \frac{c}{(1+d_2)^t}$: Unit surplus cost incurred if the total ore mined during period $t$ exceeds the processing capacity $\Theta^t$ ($c$ is the undiscounted surplus cost and $d_2$ represents the risk discount rate).

The two-stage stochastic programming model SMPSP can be summarized as follows:

$$\max \frac{1}{S} \left\{ \sum_{s\in\Omega} \sum_{t\in T} \sum_{i\in N} v_{is}^t x_i^t - \sum_{s\in\Omega} \sum_{t\in T} c^t d_s^t \right\} \tag{1}$$

$$s.t. \quad \sum_{t\in T} x_i^t \le 1 \qquad \forall i \in N \tag{2}$$

$$x_i^t - \sum_{\tau=1}^{t} x_p^\tau \le 0 \qquad \forall i \in N, p \in P_i, t \in T \tag{3}$$

$$\sum_{i\in N} w_i x_i^t \le W^t \qquad \forall t \in T \tag{4}$$

$$\sum_{i\in N} \theta_{is} w_i x_i^t - d_s^t \le \Theta^t \qquad \forall t \in T, s \in \Omega \tag{5}$$

$$x_i^t \in \{0,1\} \qquad \forall i \in N, t \in T \tag{6}$$

$$d_s^t \geq 0 \qquad \forall t \in T, s \in \Omega. \tag{7}$$

The objective function includes two terms to maximize the expected net present value of the mining operation and to minimize the expected recourse cost incurred whenever the processing constraints are violated due to metal uncertainty. In this paper, the scenarios used are realizations of a spatial random field with an equal probability of occurrence and hence the coefficient $\frac{1}{S}$ represents the probability that scenario $s$ occurs (Goovaerts, 1997).

Constraints (2) guarantee that each block can be mined at most once during the horizon (reserve constraints). The mining precedence (slope constraints) is enforced by constraints (3). Constraints (4) impose an upper bound $W^t$ on the amount of rock (waste and ore) mined during each period $t$ (mining constraints). Constraints (5) are related to the requirements on the processing levels (processing constraints). The target is to have the total amount of ore mined during any period $t$ and under any scenario $s$ be less than or equal to $\Theta^t$; otherwise, the surplus penalty cost is equal to $c^t d_s^t$. Constraints (6)–(7) enforce integrality and non-negativity conditions on the variables. Note that the variables $x_i^t$ specifying the mining sequence are the first-stage decision variables, and the variables $d_s^t$ measuring the surplus in ore production are the second-stage decision variables.

The two-stage stochastic model (1)–(7) is NP-hard since it contains the *constrained maximum closure problem* as a special case (Hochbaum and Chen, 2000; Bienstock and Zuckerberg, 2010). If the instance size is not large, it can be solved exactly, but this is not typically the case in real-world applications, justifying the use of heuristic-based methods. The next section presents a heuristic solution method based on the progressive hedging strategy.

## 3 Solution method based on the progressive hedging strategy

The proposed solution method consists of two main phases. In the first phase, a modified version of the baseline PH algorithm is used. It iteratively generates and solves an optimization sub-problem for each group of scenarios until a stopping criterion is met. In phase II, information obtained during the PH iterations is used to identify the earliest time and the latest time in which each block can be extracted. This allows us to fix many variables in the original problem SMPSP. The resulting restricted problem is solved to obtain an *implementable* solution.

In what follows, a step-by-step description of our adaptation of PH is first provided. The method used to solve the restricted problem is then described.

### 3.1 Scenario decomposition

The scenarios modelling metal uncertainty are partitioned into groups, which are then used to define the sub-problems. The scenarios within each group are chosen randomly.

Let $G$ be the set of groups, indexed by $g$. Denote the partition by $\Omega = (\Omega_1, \ldots, \Omega_{|G|})$, where $\Omega_g \subseteq \Omega$ $\forall g \in G$, $\cup_{g\in G}\Omega_g = \Omega$, and $\Omega_g \cap \Omega_{g'} = \emptyset$ $\forall g, g' \in G$ and $g \neq g'$. The first stage variables $x_i^t$ are subscripted with a group index. This can be seen as creating a copy $x_{ig}^t \in \{0,1\}$ of each $x_i^t$ for each group $g$ in order to allow the mining decisions to depend on the group, and yields the following model:

$$\max \frac{1}{S}\left\{\sum_{g\in G}\sum_{s\in\Omega_g}\sum_{t\in T}\sum_{i\in N} v_{is}^t x_{ig}^t - \sum_{g\in G}\sum_{s\in\Omega_g}\sum_{t\in T} c^t d_s^t\right\} \tag{8}$$

$$s.t. \qquad \sum_{t\in T} x_{ig}^t \leq 1 \qquad \forall i \in N, g \in G \tag{9}$$

$$
\begin{aligned}
x_{ig}^t - \sum_{\tau=1}^{t} x_{pg}^{\tau} \leq 0 \qquad & \forall i \in N, p \in P_i, t \in T, g \in G && (10)\\
\sum_{i \in N} w_i x_{ig}^t \leq W^t \qquad & \forall t \in T, g \in G && (11)\\
\sum_{i \in N} \theta_{is} w_i x_{ig}^t - d_s^t \leq \Theta^t \qquad & \forall t \in T, g \in G, s \in \Omega_g && (12)\\
x_{ig}^t = x_{ig'}^t \qquad & \forall i \in N, t \in T, g, g' \in G, g \neq g' && (13)\\
x_{ig}^t \in \{0,1\} \qquad & \forall i \in N, t \in T, g \in G && (14)\\
d_s^t \geq 0 \qquad & \forall t \in T, g \in G, s \in \Omega_g. && (15)
\end{aligned}
$$

Whereas the objective function (8) as well as constraints (9)–(12) and (14)–(15) are self-explanatory given the previous discussion in Section 2, constraints (13) deserve some explanation. They are the so-called *implementability constraints*, and are used to guarantee an *implementable* solution; that is, a solution that will be feasible and can be implemented regardless of which scenario is realized. Denote this solution by $\overline{x} = (\overline{x}_i^t)$. Constraints (13) can then be rewritten as follows:

$$
\begin{aligned}
x_{ig}^t = \overline{x}_i^t \qquad & \forall i \in N, t \in T, g \in G && (16)\\
\overline{x}_i^t \in \{0,1\} \qquad & \forall i \in N, t \in T. && (17)
\end{aligned}
$$

Following the decomposition scheme proposed in Rockafellar and Wets (1991), constraints (16) are relaxed using an augmented Lagrangean strategy (Bertsekas, 1982), which yields the following objective function:

$$
\max \sum_{g \in G} \frac{|\Omega_g|}{S} \Bigg\{ \frac{1}{|\Omega_g|} \sum_{s \in \Omega_g} \sum_{t \in T} \sum_{i \in N} v_{is}^t x_{ig}^t - \frac{1}{|\Omega_g|} \sum_{s \in \Omega_g} \sum_{t \in T} c^t d_s^t - \sum_{t \in T} \sum_{i \in N} \lambda_{ig}^t (x_{ig}^t - \overline{x}_i^t) - \frac{1}{2}\rho \sum_{t \in T} \sum_{i \in N} (x_{ig}^t - \overline{x}_i^t)^2 \Bigg\}. \qquad (18)
$$

The Lagrangean multipliers $\lambda_{ig}^t$ are associated with the relaxed constraints (16), $\rho$ is a penalty ratio, and $\frac{|\Omega_g|}{S}$ is the probability associated with group $g$ (recall that we consider that the scenarios are equiprobable, and group $g$ is composed of $|\Omega_g|$ scenarios). Given that constraints (17) require that variables $\overline{x}_i^t$ are binary, the objective function (18) becomes, after rearranging the terms:

$$
\max \sum_{g \in G} \frac{|\Omega_g|}{S} \Bigg\{ \sum_{t \in T} \sum_{i \in N} \left( \frac{1}{|\Omega_g|} \sum_{s \in \Omega_g} v_{is}^t - \lambda_{ig}^t - \frac{1}{2}\rho + \rho \overline{x}_i^t \right) x_{ig}^t - \frac{1}{|\Omega_g|} \sum_{s \in \Omega_g} \sum_{t \in T} c^t d_s^t + \sum_{t \in T} \sum_{i \in N} \lambda_{ig}^t \overline{x}_i^t - \frac{1}{2}\rho \sum_{t \in T} \sum_{i \in N} \overline{x}_i^t \Bigg\}. \qquad (19)
$$

If $\overline{x} = (\overline{x}_i^t)$ is fixed to a given value, then the model is decomposable according to the groups. Denote $\widetilde{v_{ig}^t} = \frac{1}{|\Omega_g|} \sum_{s \in \Omega_g} v_{is}^t - \lambda_{ig}^t - \frac{1}{2}\rho + \rho \overline{x}_i^t$. The sub-problem $SP_g$ associated with group $g$ can be expressed as follows:

$$
\begin{aligned}
& \max \sum_{t \in T} \sum_{i \in N} \widetilde{v_{ig}^t} x_{ig}^t - \frac{1}{|\Omega_g|} \sum_{s \in \Omega_g} \sum_{t \in T} c^t d_s^t && (20)\\
& s.t. \qquad (9)\text{–}(12) \textit{ and } (14)\text{–}(15).
\end{aligned}
$$

In this paper, $\overline{x} = (\overline{x}_i^t)$, henceforth referred to as the *inclusive schedule*, is fixed as in Rockafellar and Wets (1991); that is, the average function given the group probabilities is used. Because the scenarios are equiprobable, and the probability associated with group $g$ is $\frac{|\Omega_g|}{S}$, this means that:

$$
\overline{x}_i^t = \frac{|\Omega_g|}{S} \sum_{g \in G} x_{ig}^t \qquad \forall i \in N, t \in T. \qquad (21)
$$

The penalties, $\lambda_{ig}^t$ and $\rho$, are also adjusted as in Rockafellar and Wets (1991). The strategy used is inspired by the augmented Lagrangean method (Bertsekas, 1982) and is as follows:

$$\lambda_{ig}^t := \lambda_{ig}^t + \rho(x_{ig}^t - \overline{x}_i^t) \qquad \forall i \in N, t \in T, s \in \Omega \tag{22}$$

$$\rho := \alpha\rho \tag{23}$$

where $\alpha$ is a constant greater than or equal to 1 ($\alpha \geq 1$) to guarantee a slow increase in the penalty term. Sensitivity of the algorithm to parameter $\alpha$ is explored in Section 4.

## 3.2 Sub-problem solution method

Each iteration of PH requires solving $G$ sub-problems, each associated with a group of scenarios (problems $SP_g$ described in the previous section). This is done using the sequential heuristic proposed in Lamghari et al. (2013). In brief, the heuristic separates the problem into smaller sub-problems, each associated with a period $t \in T$. The sub-problems are solved sequentially in increasing order of $t$, and their solutions are combined to generate a solution for the original problem. Moreover, logical implications of the reserve, slope and mining constraints are used to reduce the number of decision variables in the formulation of the sub-problems to make them easier to solve. The need to resort to a sequential approach is a consequence of the large number of binary variables in the original formulation.

## 3.3 Phase II solution method

As indicated in the introduction, PH might not converge, and thus a second phase is required to generate a solution for the original problem SMPSP ((1)–(7)) described in Section 2. This second phase consists of solving a restricted problem obtained from SMPSP by considering only a subset of variables; the other variables being fixed using a strategy that exploits the information obtained during the PH iterations and the structure of the problem. In what follows, we explain which variables are fixed and to which values they are fixed.

Recall that at each iteration of PH, the *inclusive schedule* $\overline{x} = \left(\overline{x}_i^t\right)$ is computed using the following formula, where $\left(x_{ig}^t\right)$ represents the solution of the sub-problem $SP_g$ associated with group $g$ obtained at the current iteration:

$$\overline{x}_i^t = \sum_{g \in G} \frac{|\Omega_g|}{S} x_{ig}^t \qquad \forall i \in N, t \in T.$$

It is clear that any $\overline{x}_i^t$ can only take values in the interval $[0, 1]$ since $x_{ig}^t$ are binaries and $\sum_{g \in G} \frac{|\Omega_g|}{S} = 1$. Furthermore, the larger the number of fractional components $\overline{x}_i^t$ is, the less consensus there is among the scenarios.

Once PH terminates, let $\overline{bestx}$ be the best *inclusive schedule* obtained so far. By best it is meant the *inclusive schedule* with the most consensus among the scenarios (or in other words, with the fewest components having fractional values).

Partition the set of blocks into two disjoint subsets:

- $N_1 = \{i \in N : \overline{bestx}_i^t \in \{0, 1\}$ for all $t \in T\}$: the set of blocks for which a consensus has been obtained among the scenarios, or in other words, all scenarios agree that these blocks have to be mined in a specific period.
- $N_2 = \{i \in N :$ there exists a period $t \in T$ such that $0 < \overline{bestx}_i^t < 1\}$: the set of the remaining blocks or those for which a consensus has not been obtained.

All variables associated with blocks in $N_1$ are fixed to their values in $\overline{bestx}$.

Now consider a block $i \in N_2$. Even if the scenarios do not agree on which period block $i$ should be mined in, they might agree on which periods block $i$ should not be mined in. This observation is used to specify a "time window" for each block $i \in N_2$. This is done as follows.

Let $E_i = \min\left\{t \in T : \overline{bestx}_i^t > 0\right\}$ and $L_i = \max\left\{t \in T : \overline{bestx}_i^t > 0\right\}$. Variables $x_i^t$ such that $t \in [1, E_i[\cup]L_i, h]$ are fixed to the value 0, while those such that $t \in [E_i, L_i]$ are not fixed. This means that block $i$ can be scheduled no earlier than $E_i$ nor later than $L_i$. Preliminary experiments indicate that this strategy gives better results than the alternative that consists of fixing only variables associated with blocks in $N_1$. In addition, using $\overline{bestx}_i^t$ instead of the inclusive schedule obtained during the last iteration of PH improves the results.

Although the restricted problem is somewhat smaller than the original problem, it can be very large when the size of the set $N_2$ is large (i.e., when, after the iterations of PH, consensus has been obtained for only a few blocks). Consequently, solving it using an exact method might be time consuming. Instead, we propose using a sliding time window heuristic (STWH) that divides the set of time periods into three disjoint but consecutive subsets ($T_1$, $T_2$ and $T_3$). When solving the restricted problem, all variables associated with periods in the first subset, $T_1$, are fixed to feasible values; those associated with periods in the second subset, $T_2$, are restricted to be binary; and finally, those associated with periods in the last subset, $T_3$, are relaxed to be continuous. The algorithm proceeds in a sequential manner, where the subsets $T_1$, $T_2$, and $T_3$ are updated as follows:

1. Set $\tau := 1$, $T_1 := \emptyset$, $T_2 := \{\tau\}$, and $T_3 := \{\tau + 1, \ldots, h\}$
2. Solve the resulting problem
3. Fix all variables associated with period $\tau$ to the optimal values just found
4. Set $T_1 := T_1 \cup \{\tau\}$
5. If $T_1 = T$, stop. Otherwise, set $\tau := \tau + 1$, $T_2 := \{\tau\}$, $T_3 := T - \{T_1 \cup T_2\}$ and go to step 2.

The approach used to solve the restricted problem is based on integer linear programming techniques. Approaches based on such techniques have been used in the past to solve the deterministic version of the open-pit mine production scheduling problem and are also used in mine planning software (see Caccetta and Hill (2003) and Bley et al. (2010), for instance). The closest approach to ours is the one in Cullenbine et al. (2011). The authors also use a STWH, but in combination with Lagrangian relaxation. For periods in $T_3$, not only do they relax the integrality constraints as it is done in this paper, but they also relax all the other constraints (i.e., slope, mining and processing constraints) to reduce computational effort. Such a strategy is not used here because the restricted problem is of small size and can be solved in a reasonable amount of time without relaxing the other constraints of the problem, as can be seen from the results in Section 4.

### 3.4 The Algorithm

A template for the algorithm based on the progressive hedging strategy and integrating the elements that have been presented in the previous sections is given below (Algorithm 1). In our implementation, phase I terminates when full convergence is obtained or when a fixed number of iterations (*nIter*) have been performed, but one can consider any other stopping criterion; for example, a fixed amount of CPU time or when the convergence reaches a pre-specified threshold value.

## 4 Numerical results

The solution method proposed in this paper is tested on an instance generated from a copper deposit comprising 26,021 mining blocks, 13 time periods, and 20 scenarios for the metal content. All blocks $i$ are $20 \times 20 \times 10$ meters in size and weigh $w_i = 10,800$ tons each. Table 1 reports the economic parameters used to compute the blocks' economic values and the recourse costs.

The production capacities are identical in all periods. For each period $t$, the mining capacity and the processing capacity are set to 27 million tonnes and 4.7 million tonnes, respectively ($\frac{\text{Total amount of rock}}{\text{number of periods}}$ plus a margin of 20% and $\frac{\text{Total expected amount of ore}}{\text{number of periods}}$ plus a margin of 5%, respectively).

The algorithm outlined in Section 3.4 (Algorithm 1) is implemented in C++ and run on an Intel(R) Xeon(R) CPU E7-8837 computer (2.67 GHz) with 1 TB of RAM running under Linux. To speed up the

**Algorithm 1** based on the progressive hedging strategy

```
Initialization
    $n$ and $h$, the number of blocks and the number of periods, respectively
    $countIter := 0$, the number of iterations of PH performed so far
    $nIter$, the maximum number of iterations of PH allowed
    $\overline{bestx} := \emptyset$, the best inclusive schedule obtained so far
    $C := 0$, the number of variables that have converged so far
    $\lambda_{ig}^{t} := 0$, the initial value of the Lagrangean multipliers
    $\alpha \geq 1$, a parameter of the algorithm used to adjust the value of the penalty ratio $\rho$
    Partition the scenarios into groups
    for each group $g$ do
        Solve the sub-problem $SP_g$ described in Section 3.1 using the method summarized in Section 3.3
        and the original economic values
    end for
    Compute the values of the components of the inclusive schedule $\overline{x} = \left(\overline{x}_i^t\right)$ using equation (21)
    Update $\overline{bestx}$ and $C$
    Initialize the value of the penalty ratio, $\rho$
Search
    Phase I: Looking for similarities among the scenario groups
    while ($C \neq nh$ and $countIter < nIter$) do
        countIter := countIter+1
        for each group $g$ do
            Adjust the Lagrangean multipliers according to equation (22)
            Update the modified economic values ($\widetilde{v_{ig}^{t}} = \frac{1}{|\Omega_g|}\sum_{s\in\Omega_g} v_{is}^{t} - \lambda_{ig}^{t} - \frac{1}{2}\rho + \rho\overline{x}_i^t$)
            Solve the sub-problem $SP_g$
        end for
        Update $\overline{x}$, $\overline{bestx}$, and $C$
        Adjust the penalty ratio $\rho$ according to equation (23)
    end while
    Phase II: Getting an implementable schedule
    Solve the restricted problem (the SMPSP in Section 2, but where some variables are fixed according to the
    strategy described in Section 3.4).
```

Table 1: Economic parameters used to compute the objective function coefficients.

| Parameter | Value |
|---|---|
| Mining cost | \$1/t |
| Processing cost | \$9/t |
| Metal price | \$2/lb |
| Selling cost | \$0.3/lb |
| Undiscounted surplus cost for ore ($c$) | \$15/t |
| Discount rate ($d_1$) | 10% |
| Risk discount rate ($d_2$) | 10% |

algorithm, an OpenMP parallel implementation of the initialization phase and the first phase is used. It is based on a simple master-worker strategy. The master operates as a central memory, which manages the search. Each worker processor deals with one sub-problem. It updates the associated penalties (Lagrangean multipliers), computes the modified economic values, solves the sub-problem, and communicates the solution to the master. When all sub-problems are solved, the master computes the *inclusive schedule* and sends it to the worker processors to update the penalties.

Version 12.2 of CPLEX is used to solve the sub-problems associated with the periods within the sequential heuristic described in Section 3.2, and to solve the restricted problem of the second phase introduced in

Section 3.3. CPLEX is also used to solve the linear relaxation of the two-stage stochastic problem (1)–(7) presented in Section 2 to obtain an upper bound on the optimal value, which is used to assess the quality of the solutions produced by the proposed algorithm. In all numerical experiments, the predual parameter of CPLEX is set to 1; that is, the dual linear programming problem is passed to the optimizer. This is a useful technique for problems with more constraints than variables, such as the one considered in this paper. Unless otherwise specified, all other CPLEX parameters are set to their default values since preliminary experiments indicated that these settings yield better results.

In what follows, the results of the experiments conducted to determine appropriate parameter values for the first phase of the algorithm are first discussed. This is followed by a comparison of the performance of the algorithm considering the different alternatives for each of the two phases (i.e., in phase I, varying the size of the groups, and in phase II, using the sliding time window heuristic versus using an exact method).

## 4.1 Parameter calibration

As indicated earlier, at each iteration of PH, the value of the penalty ratio $\rho$ used to compute the modified economic values is adjusted by multiplying it by a parameter $\alpha \geq 1$. As in Crainic et al. (2011), the initial value of $\rho$ is set to $1+\log(1+D)$, $D$ being the inconsistency level; i.e., the number of variables that have not converged after the initialization phase. To identify an appropriate value for the parameter $\alpha$, we considered the case where the groups are comprised of 1 scenario (the case where convergence is the most difficult to obtain), we fixed the number of iterations of PH to 30, and we ran tests using 4 different values for $\alpha$: 1, 1.1, 1.2, and 1.3. The sliding time window heuristic (STWH) was used to solve the restricted problem of phase II. The results of these tests are reported in Table 2. The column headings are defined as follows:

- $\alpha$, the value used for the parameter $\alpha$.
- $\%Convergence = \frac{|\{(i,t):\overline{bestx}_i^t \in \{0,1\}|}{nh} \times 100$, the percentage of binary variables that have converged after the first phase of the algorithm.
- $\%Gap = \frac{Z_{LR}-Z^*}{Z_{LR}} \times 100$, the gap between $z_{LR}$, the linear relaxation optimal value, and $z^*$, the value of the solution found by the proposed algorithm. The smaller the value of $\%Gap$ is, the better the solution is.
- $Time$, the solution time in minutes. Note that the time reported for phase I includes the time needed to initialize PH with the procedure described in Section 3.4.

Table 2: Sensitivity of the solution method to the parameter $\alpha$.

| | | | ***Time (minutes)*** | | |
|---|---|---|---|---|---|
| $\boldsymbol{\alpha}$ | ***%Convergence*** | ***%Gap*** | **Phase I** | **Phase II** | **Total** |
| 1 | 74.28 | 0.256 | 258.40 | 636.62 | 895.03 |
| 1.1 | 75.60 | 0.256 | 298.02 | 326.47 | 624.49 |
| 1.2 | 83.22 | 0.263 | 271.77 | 54.97 | 326.74 |
| 1.3 | 90.14 | 0.535 | 261.60 | 2.42 | 234.01 |

As one can expect, the value of the parameter $\alpha$ has almost no effect on the computational time required for the iterations of PH (phase I), but it does have a significant effect on the convergence rate and the time required to solve the restricted problem (phase II). By using a large value for $\alpha$, a higher penalty is associated with violation of the implementability constraints (16), which accelerates convergence. The advantage of having a large rate of convergence is that a large number of variables are fixed in phase II. The restricted problem is thus considerably smaller, which results in a significant reduction of the time needed to solve it. It is worth mentioning, however, that when variables converge, they do not necessarily converge to optimal values, since we are dealing with a mixed-integer stochastic problem. This explains why, in terms of solution quality, smaller values of $\alpha$ yield better results. The results indicate that the value 1.2 is the most appropriate value if one is looking for a trade-off between solution time and solution quality. Thus this value is used in all further experiments.

The number of iterations of PH performed before solving the restricted problem (i.e., before phase II) is another parameter of phase I. We considered 4 values for this parameter (*nIter*): 0, 10, 20, 30. The value 0 means that the algorithm skips phase I and moves directly to phase II once the initialization phase is completed (i.e., once the sub-problems are solved using the original economic values). Results obtained when considering the four aforementioned values for *nIter* are summarized in Table 3, which has the same structure as Table 2, except that the first column indicates the number of iterations of PH performed. Again, the restricted problem is solved using STWH and the time reported for phase I includes the time needed for the initialization phase.

Table 3: Sensitivity of the solution method to the parameter $nIter$.

| | | | ***Time (minutes)*** | | |
|---|---|---|---|---|---|
| ***nIter*** | ***%Convergence*** | ***%Gap*** | **Phase I** | **Phase II** | **Total** |
| 0 | 72.74 | 0.238 | 9.11 | 641.97 | 651.08 |
| 10 | 74.10 | 0.257 | 98.96 | 455.82 | 554.78 |
| 20 | 76.90 | 0.267 | 185.38 | 322.32 | 507.71 |
| 30 | 83.22 | 0.263 | 271.77 | 54.97 | 326.74 |

From the results, one can see that the total solution time decreases as *nIter* increases. This is explained by the fact that the second phase requires less time because more variables are fixed in the restricted problem (c.f. column "*%Convergence*"), and this decrease outweighs the increase in the time needed to complete the first phase (increase due to performing more iterations of PH).

There is no guarantee that a better solution will be obtained by waiting for a higher rate of convergence (i.e., by increasing the number of PH iterations). Actually, quite the opposite is more likely to happen, as can be observed from the values of *%Gap*. Thus, in all further experiments, *nIter* is fixed to 10. Note that with this value, the total solution time is larger as compared to $nIter = 30$, but it is acceptable, considering that the solution is better.

## 4.2 Effect of grouping scenarios

In this section, the effects on convergence, solution time, and solution quality of grouping scenarios are analyzed. Five sizes for the groups are considered: 3, 5, 7, 10, and 20. For each size, the scenarios within the groups are chosen randomly. Recall that the instance considered in this paper contains 20 scenarios. Thus, size 3 signifies that 7 sub-problems are solved at each iteration of the initialization phase and of the first phase. The number of sub-problems reduces to 4, 3 and 2 for sizes 5, 7 and 10, respectively. Finally, with size 20, there is only one sub-problem, which is equivalent to the original two-stage stochastic problem in Section 2. Consequently, for this size, neither phase I nor phase II will be necessary since the algorithm will provide an implementable solution at the initialization phase. The interest of considering size 20 is that this will allow us to compare the algorithm proposed in this paper to the sequential heuristic proposed in Lamghari et al. (2013) and briefly described in Section 3.2.

For each group size, we ran tests using $\alpha = 1.2$ and $nIter = 10$. The restricted problem of phase II was solved using STWH. The results are summarized in Table 4, where, as in Tables 2 and 3, we report the values of *%Convergence* and *%Gap* (as defined previously), the time required to complete phase I, the time spent solving the restricted problem (phase II), and the total solution time. Times are given in minutes.

As expected, increasing the size of the groups results in a faster convergence and is computationally more efficient. Considering the 20 scenarios at once is the fastest (only 7 minutes). This good performance is, as explained earlier, due to the fact that full convergence (an implementable solution) is obtained at the initialization phase and thus neither phase I nor phase II are necessary (they are not completed). When considering groups of 10 scenarios each, full convergence is not achieved after 10 iterations of PH, but a rate of convergence of 96.41% is obtained, requiring a total of 70 minutes. Then in phase II, most variables are fixed, and consequently, solving the restricted problem is straightforward and required less than 2 minutes. It can be seen that decreasing the size of the groups leads to longer solution times. It is worth noting that

Table 4: Effect of grouping scenarios during Phase I.

| | | | *Time (minutes)* | | |
|---|---|---|---|---|---|
| ***Group Size*** | ***%Convergence*** | ***%Gap*** | **Phase I** | **Phase II** | **Total** |
| 3 | 86.81 | 0.292 | 83.06 | 29.56 | 112.62 |
| 5 | 92.38 | 0.345 | 81.93 | 5.13 | 87.06 |
| 7 | 94.11 | 0.402 | 77.91 | 3.11 | 81.02 |
| 10 | 96.41 | 0.514 | 70.28 | 1.59 | 71.87 |
| 20 | 100 | 0.993 | 7.71 | 0 | 7.71 |

the size of the groups does not have a significant impact on the time required to solve the sub-problems. This is in part explained by the fact that the size of the sub-problems is not really affected by the introduction of additional scenarios. In fact, the number of variables and constraints increases very slightly when the number of scenarios is increased ($T$ continuous variables $d_s^t$ and the associated constraints (12) are added for every additional scenario $s$).

The results in Table 4 also reinforce the observations made in Sections 4.1 and 4.2: when the rate of convergence is small, solving the restricted problem requires a bit of extra time because it is of larger size, but this extra time allows an increase in solution quality.

Now, if we compare the sequential heuristic in Lamghari et al. (2013) (size 20) to the algorithm proposed in this paper (sizes 3, 5, 7, and 10), we can see that the latter outperforms the sequential heuristic (improves the gap by 48% to 71%), but it runs 10 to 15 times longer than does the sequential heuristic. It seems thus fair to conclude that the sequential heuristic is the best choice if fast optimization is required, but the opposite is true if more time is allowed. Note that although the solution times of the proposed algorithm are somewhat long compared to those of the sequential heuristic, they are still considerably smaller than those required by the commercial solver CPLEX to solve the two-stage stochastic formulation (1)–(7) (at most 9 hours to find a near-optimal solution compared to 34 hours required by CPLEX to solve only the linear relaxation of the two-stage stochastic model).

## 4.3 Effect of combining PH with STWH

In this section, we first evaluate whether using the sliding time window heuristic (STWH) instead of an exact method in phase II improves the performance of the proposed algorithm. We then examine the value-added of PH; that is, whether STWH is efficient if it is not combined with PH.

STWH is compared to the Branch-and-Cut algorithm implemented in CPLEX, henceforth identified as BCA, and the results are given in Table 5. The 4 alternatives described in the previous section for grouping scenarios in the first phase are considered (i.e., sizes 3, 5, 7, and 10). The values of *%Convergence* obtained for each group are reported in the second column, the next two columns give the *%Gap* of the objective values obtained by each method (with respect to the upper bound provided by CPLEX), while the last two columns report CPU times, in minutes, spent by STWH and BCA to solve the restricted problem (i.e., to complete phase II). The times required to complete phase I are not reported because they are similar whether STWH or BCA is used, and have already been reported in Table 4. The maximum run time for BCA is set to 16 hours (960 minutes) because it was observed that BCA solution times exceeded 1 day for group size 3.

Table 5: Effect of using STWH in phase II instead of BCA implemented in CPLEX.

| | | *%Gap* | | *Time Phase II (minutes)* | |
|---|---|---|---|---|---|
| ***Group Size*** | ***%Convergence*** | **STWH** | **BCA** | **STWH** | **BCA** |
| 3 | 86.81 | 0.292 | 0.271 | 29.56 | 960 |
| 5 | 92.38 | 0.345 | 0.301 | 5.13 | 742.06 |
| 7 | 94.11 | 0.402 | 0.377 | 3.11 | 54.68 |
| 10 | 96.41 | 0.514 | 0.485 | 1.59 | 9.05 |

First note that for size 3, BCA was not able to solve the restricted problem to optimality within the 16 hours allowed, and thus BCA might provide a better solution if it were given more time, but for the other sizes, BCA provided optimal solutions. With the 16-hour time limit, the results in Table 5 indicate that using STWH improves the performance of the algorithm considerably. STWH is 6 to 33 times faster, and any slight improvements in solution quality derived from using BCA are outweighed by the increase in solution time.

Finally, to examine the value-added of PH, the problem is solved with STWH without combining it with PH. As one would expect, this approach is computationally expensive. It requires 12 days compared to tens of minutes when PH is used. The solution obtained after 12 days is slightly better than the one obtained when combining PH and STWH, but the difference is not significant (% Gap is 0.264 compared to 0.292). This clearly highlights the significant benefit of combining PH with STWH.

## 5 Conclusions

This paper explores the development of an efficient optimization approach to address the large and complex problems faced by the mining industry when scheduling production in open-pit mines under metal uncertainty. We have proposed a two-phase solution approach based on the progressive hedging strategy (PH). PH is used in phase I where the problem is first decomposed by partitioning the set of scenarios modeling metal uncertainty into groups, and then the sub-problems associated with each group are solved iteratively to drive their solutions to a common solution. In phase II, a strategy exploiting information obtained during the PH iterations and the structure of the problem under study is used to reduce the size of the original problem, and the resulting smaller problem is solved to generate an implementable solution.

Through computational experiments, we have shown that the proposed algorithm performs very well in terms of solution quality. We have provided an analysis that shows the advantages and disadvantages of increasing the size of the groups of the scenarios during the first phase of the algorithm. This analysis indicates that increasing the size of the groups has a positive impact on the solution time, but it negatively impacts the solution quality. For the second phase, we have compared two alternate solution methods: a sliding time window heuristic (STWH) and the branch-and-cut algorithm implemented in the commercial solver CPLEX (BCA). The results indicate that, with respect to solution quality, the two methods are comparable with a slightly better performance for BCA, but STWH has a significant superior performance to that of BCA in terms of solution time. The results also indicate that STWH is efficient only if combined with the progressive hedging algorithm (PH); i.e., only if used within the algorithm proposed in this paper. With respect to the sequential heuristic previously proposed by the authors, the algorithm proposed here dominates in terms of solution quality. Its weakness is that it requires longer solution times; however, these solution times are considerably smaller compared to those required by the commercial solver CPLEX to solve the problem directly; i.e., to solve the two-stage stochastic model in Section 2.

As mentioned earlier, this paper represents ongoing efforts to efficiently address the stochastic MPSP. Future work may consider investigating whether the algorithm would be as successful or not in solving variants of the MPSP including more operational constraints as it is in solving the "classical" variant considered in this paper. Indeed, it is a general-purpose algorithm and should be applicable to any of these variants. Other research avenues include considering other strategies for updating the penalties within PH and other methods for solving the sub-problems.

## References

Albor, F., Dimitrakopoulos, R., 2010. Algorithmic approach to pushback design based on stochastic programming: Method, application and comparisons. IMM Transactions, Mining Technology, 119(2): 88-101. 119 (2), 88–101.

Asad, M., Dimitrakopoulos, R., 2013. Implementing a parametric maximum flow algorithm for optimal open pit mine design under uncertain supply and demand. Journal of the Operational Research Society 64, 185–197.

Bertsekas, D., 1982. Constrained optimization and Lagrange multipliers methods. Academic Press, New York.

Bienstock, D., Zuckerberg, M., 2010. Solving LP relaxations of large-scale precedence constrained problems. Lecture Notes in Computer Science 6080, 1–14.

Birge, J., Louveaux, F., 1997. Introduction to stochastic programming. Springer.

Bley, A., Boland, N., Fricke, C., Froyland, G., 2010. A strengthened formulation and cutting planes for the open pit mine production scheduling problem. Computers & Operations Research 37 (9), 1641–1647.

Boland, N., Dumitrescu, I., Froyland, G., 2008. A multistage stochastic programming approach to open pit mine production scheduling with uncertain geology. Optimization OnlineLast accessed: January 31, 2012. URL http://www.optimization-online.org/DB_FILE/2008/10/2123.pdf

Caccetta, L., Hill, S., 2003. An application of branch and cut to open pit mine scheduling. Journal of Global Optimization 27 (2-3), 349–365.

Crainic, T., Fu, X., Gendreau, M., Rei, W., Wallace, S.W., 2011. Progressive hedging-based metaheuristics for stochastic network design. Networks 58 (2), 114–124.

Crainic, T., Hewitt, M., Rei, W., 2014. Scenario grouping in a progressive hedging-based meta-heuristic for stochastic network design. Computers & Operations Research 43, 90–99.

Cullenbine, C., Wood, R., Newman, A., 2011. A sliding time window heuristic for open pit mine block sequencing. Optimization Letters 5, 365–377.

Dimitrakopoulos, R., Farrelly, C., Godoy, M., 2002. Moving forward from traditional optimization: grade uncertainty and risk effects in open pit mine design. IMM Transactions 111, A82–A88.

Dowd, P., 1994. Risk assessment in reserve estimation and open-pit planning. Transactions of the Institution of Mining and Metallurgy 103, A148–A154.

Godoy, M., 2002. The effective management of geological risk. Ph.D. thesis, University of Queensland, Qld, Australia.

Goovaerts, P., 1997. Geostatistics for Natural Resources Evaluation. Oxford University Press, New York.

Haugen, K., Løkketangen, A., Woodruff, D. L., 2001. Progressive hedging as a metaheuristic applied to stochastic lot-sizing. European Journal of Operational Research 132, 116–122.

Hochbaum, D., Chen, A., 2000. Improved planning for the open-pit mining problem. Operations Research 48, 894–914.

Lamghari, A., Dimitrakopoulos, R., 2012. A diversified tabu search approach for the open-pit mine production scheduling problem with metal uncertainty. European Journal of Operational Research 222, 642–652.

Lamghari, A., Dimitrakopoulos, R., 2013. A network-flow based algorithm for scheduling production in multi-processor open-pit mines accounting for metal uncertainty. Les Cahiers du GERAD, G–2013–63, HEC Montréal.

Lamghari, A., Dimitrakopoulos, R., Ferland, J., 2013. A variable descent neighborhood algorithm for the open-pit mine production scheduling problem with metal uncertainty. Journal of the Operational Research Society. URL http://dx.doi.org/10.1057/jors.2013.81

Løkketangen, A., Woodruff, D., 1996. Progressive hedging and tabu search applied to mixed integer (0,1) multistage stochastic programming. Journal of Heuristics 2, 111–128.

Marcotte, D., Caron, J., 2013. Ultimate open pit stochastic optimization. Computers & Geosciences 51, 238–246.

Menabde, M., Froyland, G., Stone, P., Yeates, G., 2007. Mining schedule optimization for conditionally simulated orebodies. Orebody Modelling and Strategic Mine Planning, The Australasian Institute of Mining and Metallurgy, Spectrum Series 14, 379–384.

Ramazan, S., Dimitrakopoulos, R., 2013. Production scheduling with uncertain supply: A new solution to the open pit mining problem. Optimization and Engineering 14, 361–380.

Ravenscroft, P., 1992. Risk analysis for mine scheduling by conditional simulation. Transactions of the Institution of Mining and Metallurgy, Section A: Mining Technology, A104–A108.

Rockafellar, R., Wets, R., 1991. Scenarios and policy aggregation in optimization under uncertainty. Mathematics of Operations Research 16 (1), 119–147.