European Journal of Operational Research 253 (2016) 843–855

Contents lists available at ScienceDirect

European Journal of Operational Research

journal homepage: www.elsevier.com/locate/ejor

Innovative Applications of O.R.

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

Amina Lamghari*, Roussos Dimitrakopoulos

COSMO – Stochastic Mine Planning Laboratory, Department of Mining and Materials Engineering, McGill University, FDA Building, 3450 University Street, Montreal, Quebec H3A 2A7, Canada

ARTICLE INFO

Article history:
Received 19 May 2014
Accepted 5 March 2016
Available online 11 March 2016

Keywords:
Open-pit mine production scheduling
Progressive hedging method
Lagrangian relaxation
Sliding time window heuristic
Metaheuristics

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.

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. The expense of setting up mining operations, the volatility of the markets, and the in-situ spatial variability of the deposit grades are all factors that make profit margins tight and investments risky. There is thus a clear interest in careful planning that can reduce risk and establish the most economically efficient mine production schedule possible that enables the mine operator to meet production targets and make the best possible return on investment. Even a 1 percent increase in the efficiency of the ore removal scheme can significantly increase the profitability of an operation that is worth hundreds of millions of dollars, as mining companies recognize and as the body of literature on the topic attests (see for example Dimitrakopoulos (2011); Newman, Rubio, Caro, Weintraub, and Eurek (2010); Whittle (2015), and the references therein).

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'

* Corresponding author. Tel.: +1 514 733 8489.
E-mail addresses: amina.lamghari@mcgill.ca (A. Lamghari), roussos.dimitrakopoulos@mcgill.ca (R. Dimitrakopoulos).

http://dx.doi.org/10.1016/j.ejor.2016.03.007

metal content, can lead to incorrect assessments of the the production forecasts, the final pit limits, and the net present value (Albor & Dimitrakopoulos, 2010; Asad & Dimitrakopoulos, 2013; Dimitrakopoulos, Farrelly, & Godoy, 2002; Dowd, 1994; Marcotte & Caron, 2013; Menabde, Froyland, Stone, & Yeates, 2007; Ravenscroft, 1992). 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, Dumitrescu, & Froyland, 2008; Ramazan & 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, Dimitrakopoulos, and Ferland (2013) and Lamghari and Dimitrakopoulos (2013), respectively. Behrang, Hooman, and Clayton (2014) used a clustering approach to reduce the number of binary variables and thus make larger instances computationally tractable by exact methods, while Chatterjee (2014) proposed a sequential heuristic where the sub-problems are solved using lagrangian relaxation techniques. Lagos, Espinoza, Moreno, and Amaya (2011) compared three approaches for optimization under uncertainty, namely Value-at-Risk, Conditional Value-at-Risk and a robust optimization approach. Marcotte and Caron (2013) studied the effect of metal uncertainty on the pit limits, and Fricke, Velletri, and Wood (2014) developed an analytic framework to quantify the impact of uncertainty and support strategic mine planning decisions.

In this paper, a novel heuristic-based method is proposed 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-problems 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-problems 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 will not take as much time to solve as the original problem would.

This approach was successfully used to solve stochastic lot-sizing problems (Haugen, Løkketangen, & Woodruff, 2001) and stochastic network design problems (Crainic, Fu, Gendreau, Rei, & S.W., 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, Hewitt, and Rei (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.

Numerical results are provided 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. The proposed solution approach is then compared to the sequential heuristic proposed in Lamghari et al. (2013), which is based on a time-decomposition strategy rather than a scenario-decomposition strategy, to a solution approach where STWH is used alone, and to a branch-and-cut approach. The results of these tests indicate the superiority of the proposed solution approach over the other approaches and show that combining PH and STWH leads to an efficient algorithmic framework that offers an excellent trade-off between solution quality and solution time (near-optimal solutions, within less than 1 percent of optimality on average, are obtained in a few minutes up to a few hours). Finally, we demonstrate on benchmark instances the benefits of solving a stochastic model rather than its deterministic counterpart. In this comparison, it

is found that the objective function value is 1–27 percent higher when the stochastic approach is used.

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 & Louveaux, 2011). 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 $t = 1, \ldots, h$.

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

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.

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

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).

Θ^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).

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

$v_{is}^t = \frac{v_{is}}{(1+d_1)^t}$: 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 Θ^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:

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

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

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

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

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

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

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

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 Θ^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 (Bienstock & Zuckerberg, 2010; Hochbaum & Chen, 2000). 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 subproblem 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 modeling 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:

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

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

$$x_{ig}^t - \sum_{\tau=1}^{t} x_{pg}^\tau \le 0 \qquad \forall i\in N, p\in P_i, t\in T, g\in G \tag{10}$$

$$\sum_{i\in N} w_i x_{ig}^t \le W^t \qquad \forall t\in T, g\in G \tag{11}$$

$$\sum_{i\in N} \theta_{is} w_i x_{ig}^t - d_s^t \le \Theta^t \qquad \forall t\in T, g\in G, s\in\Omega_g \tag{12}$$

$$x_{ig}^t = x_{ig'}^t \qquad \forall i\in N, t\in T, g, g'\in G, g\ne g' \tag{13}$$

$$x_{ig}^t \in \{0,1\} \qquad \forall i\in N, t\in T, g\in G \tag{14}$$

$$d_s^t \ge 0 \qquad \forall t\in T, g\in G, s\in\Omega_g. \tag{15}$$

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 $\bar{x} = (\bar{x}_i^t)$. Constraints (13) can then be rewritten as follows:

$$x_{ig}^t = \bar{x}_i^t \qquad \forall i\in N, t\in T, g\in G \tag{16}$$

$$\bar{x}_i^t \in \{0,1\} \qquad \forall i\in N, t\in T. \tag{17}$$

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

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

The Lagrangian multipliers λ_{ig}^t are associated with the relaxed constraints (16), ρ 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 $\bar{x}_i^t$ are binary, the objective function (18) becomes, after rearranging the terms:

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

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

$$\max \sum_{t\in T}\sum_{i\in N}\widetilde{v_{ig}^t} x_{ig}^t - \frac{1}{|\Omega_g|}\sum_{s\in\Omega_g}\sum_{t\in T} c^t d_s^t$$
$$s.t. \quad (9)-(12) \text{ and } (14)-(15). \tag{20}$$

In this paper, $\bar{x} = (\bar{x}_i^t)$, 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:

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

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

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

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

where α is a constant greater than or equal to 1 ($\alpha \ge 1$) to guarantee a slow increase in the penalty term. Sensitivity of the algorithm to parameter α 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* $\bar{x} = (\bar{x}_i^t)$ 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:

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

It is clear that any $\bar{x}_i^t$ 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 $\bar{x}_i^t$ 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 (c.f. results in Section 4.3). 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 τ 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, Boland, Fricke, and Froyland (2010), for instance). The closest approach to ours is the one in Cullenbine, Wood, and Newman (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 prespecified threshold value.

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 Lagrangian multipliers
$\alpha \geq 1$, a parameter of the algorithm used to adjust the value of the penalty ratio ρ
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 3.2 and the original economic values
end for
Compute the values of the components of the *inclusive schedule* $\bar{x} = (\bar{x}_i^t)$ using equation (21)
Update $\overline{bestx}$ and C
Initialize the value of the penalty ratio, ρ

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 Lagrangian multipliers according to equation (22)
        Update the modified economic values ($\widetilde{v_{ig}^t} = \frac{1}{|\Omega_g|}\sum_{s\in\Omega_g} v_{is}^t - \lambda_{ig}^t - \frac{1}{2}\rho + \rho\bar{x}_i^t$)
        Solve the sub-problem SP_g
    end for
    Update $\bar{x}$, $\overline{bestx}$, and C
    Adjust the penalty ratio ρ 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 3.3).

4. Numerical results

The solution method proposed in this paper is tested on the instances introduced in Lamghari and Dimitrakopoulos (2012). There are two datasets, D1 and D2, each consisting of 5 different instances from actual copper and gold deposits, respectively. Table 1 provides details about the instances. While the number of blocks and the number of periods differ from one instance to another,

Table 1
Characteristics of the instances in the two datasets.

Dataset	Instance	Number of blocks (N)	Number of periods (T)	Number of scenarios (S)
D1	C1	4273	3	20
Metal type: copper	C2	7141	4	20
Block size: $20 \times 20 \times 10$ meters	C3	12,627	7	20
Block weight: 10, 800 tons	C4	20,626	10	20
	C5	26,021	13	20
D2	G1	18,821	5	20
Metal type: gold	G2	23,901	7	20
Block size: $15 \times 15 \times 10$ meters	G3	30,013	8	20
Block weight: 5, 625 tons	G4	34,981	9	20
	G5	40,762	11	20

Table 2
Economic parameters used to compute the objective function coefficients.

Parameters	D1	D2
Mining cost	\$1/t	\$1/t
Processing cost	\$9/t	\$15/t
Metal price	\$2/lb	\$29/g
Selling cost	\$0.3/lb	\$0.2/g
Undiscounted surplus cost for ore (c)	\$15/t	\$17/t
Discount rate (d_1)	10 percent	10 percent
Risk discount rate (d_2)	10 percent	10 percent

the number of scenarios used to model metal uncertainty is similar in all instances. More specifically, 20 equiprobable scenarios representing the mineral deposits were generated from a limited amount of drilling information using the geostatistical techniques of conditional simulation, which can be seen as a complex Monte Carlo simulation framework able to reproduce all available data and information as well as spatial statistics of the data. Further details about these techniques can be found in Boucher and Dimitrakopoulos (2009); Chiles and Delfiner (2012); Goovaerts (1997); Horta and Soares (2010); Maleki and Emery (2015); Rossi and Deutsch (2014). Twenty scenarios are sufficient to capture metal uncertainty, as previous work, such as that of Albor and Dimitrakopoulos (2009), indicates that after about 15 simulated representations of an orebody, stochastic schedules converge to both a stable final physical schedule and stable production forecasts. The reason for this is that while simulated scenarios represent a mineral deposit at the support-scale of mining blocks with a volume of a few cubic meters, the production schedule of a mine groups several thousand of these blocks in a single time period, subject to certain constraints. Therefore, since the support-scale of a mine's schedule is orders of magnitude larger than that of the simulated representations of the mineral deposit being scheduled, the stochastic schedule becomes insensitive to additional scenarios after a relatively small number of scenarios. Note that some papers addressing the same problem considered in this paper use only 5–10 scenarios (Boland et al., 2008; Menabde et al., 2007).

Table 2 reports the economic parameters used to compute the blocks' economic values and the recourse costs, which are based on real-life data provided by our industrial partners.

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, 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 (Lagrangian 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. To reduce the time required to complete phase I, the optimality tolerance parameter of CPLEX is set to 1 percent when solving the sub-problems associated with the periods. This parameter is set to its default value when solving the restricted problem of the second phase. 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, namely the Branch-and-Cut algorithm implemented in CPLEX). Finally, results showing the expected gain from solving the proposed stochastic model rather than its deterministic counterpart are presented. Note that when calibrating parameters, only the 5 instances in the dataset D1 are used. All 10 instances summarized in Table 1 are used in the remaining tests.

4.1. Parameter calibration

As indicated earlier, at each iteration of PH, the value of the penalty ratio ρ 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 ρ 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 α, 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 α: 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 3. The column headings are defined as follows:

- *Instance*, the name of the instance.
- α, the value used for the parameter α.
- $\%Convergence = \frac{|\{(i,t):\overline{bestx}_i^t \in \{0,1\}|}{nh} \times 100$, the percentage of binary variables that have converged after the first phase of the algorithm.
- Z^*, the value of the solution found by the proposed algorithm in dollars.
- $\%Gap = \frac{Z_{LR}-Z^*}{Z_{LR}} \times 100$, the gap between z_{LR}, the linear relaxation optimal value of the two-stage stochastic problem (1)–(7) presented in Section 2, and Z^* as defined above. 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.

As one can expect, the value of the parameter α 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 α, 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 α yield in general 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 4, which has the same structure as Table 3, except that the second 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.

From these 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). Note also that 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. On average, the gap is 0.1392 percent when $nIter = 10$ versus 0.1401 percent, 0.1420 percent, and 0.1949 percent when $nIter = 0$, $nIter = 20$ and $nIter = 30$, respectively. Note that the difference in gap does not appear significant, but given the scale of the problem it represents a difference in value of hundreds of thousands of dollars, as can be seen from the values of Z^* in the fourth column of the table.

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 instances considered in this paper contain 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 for the copper instances (dataset D1) and for the gold instances (dataset D2) are summarized in Tables 5 and 6, respectively. As in Tables 3 and 4, we report the values of *%Convergence*, of Z^*, and of *%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 are given in minutes in the last three columns.

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 18 minutes on average, considering the 10 instances in the two datasets). 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 on average, a rate of convergence of 91.40 percent is obtained, requiring a total of 166 minutes. Then in phase II, most variables are fixed, and consequently, solving the restricted problem is straightforward and required on average less than 8 minutes. It can be seen that decreasing the size of the groups leads to a smaller rate of convergence, and consequently longer total solution times. It is worth noting that the size of the groups does not have a significant impact on the time required to solve the sub-problems (Phase I). 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 Tables 4 and 5 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 in terms of solution quality (on average, the gap is 1.547 percent when the sequential heuristic is used versus 0.538 percent, 0.566 percent, 0.656 percent, and 0.812 percent when the algorithm proposed in this paper is used with group sizes 3, 5, 7, and 10, respectively). It should be noted that, although the differences in the gap appear small, for the context

Table 3
Sensitivity of the solution method to the parameter α.

Instance	α	%Convergence	Z^*($)	%Gap	Time (minutes)		
					Phase I	Phase II	Total
C1	1	39.35	165,523,000	0.010	3.07	0.10	3.17
	1.1	43.29	165,526,000	0.008	3.24	0.08	3.31
	1.2	49.56	165,515,000	0.014	3.18	0.04	3.22
	1.3	61.25	165,497,000	0.025	2.86	0.01	2.87
C2	1	48.60	199,137,000	0.024	15.44	0.61	16.05
	1.1	52.18	199,136,000	0.025	14.55	0.40	14.95
	1.2	57.31	199,141,000	0.022	13.71	0.20	13.91
	1.3	69.89	198,949,000	0.119	11.22	0.03	11.25
C3	1	61.16	229,066,000	0.087	45.79	30.13	75.92
	1.1	63.48	229,067,000	0.087	52.10	18.26	70.36
	1.2	68.44	229,062,000	0.089	45.73	5.29	51.02
	1.3	79.92	228,789,000	0.208	39.92	0.44	40.36
C4	1	69.16	252,418,000	0.315	162.19	282.31	444.50
	1.1	70.96	252,359,000	0.338	165.05	235.00	400.05
	1.2	78.65	252,036,000	0.466	158.99	26.72	185.71
	1.3	86.80	251,295,000	0.758	126.19	1.09	127.28
C5	1	74.42	244,933,000	0.266	238.25	751.53	989.78
	1.1	75.83	244,969,000	0.251	229.51	422.29	651.79
	1.2	83.88	244,645,000	0.383	203.65	42.27	245.91
	1.3	90.11	244,281,000	0.531	172.22	2.39	174.61

Table 4
Sensitivity of the solution method to the parameter *nIter*.

Instance	nIter	%Convergence	Z^*($)	%Gap	Time (minutes)		
					Phase I	Phase II	Total
C1	0	39.02	165,522,000	0.010	0.16	0.12	0.28
	10	39.44	165,527,000	0.007	1.17	0.10	1.27
	20	43.34	165,524,000	0.009	2.25	0.07	2.32
	30	49.56	165,515,000	0.014	3.18	0.04	3.22
C2	0	46.70	199,129,000	0.029	0.60	0.89	1.49
	10	48.19	199,139,000	0.024	5.07	0.60	5.67
	20	52.83	199,128,000	0.029	9.83	0.40	10.23
	30	57.31	199,141,000	0.022	13.71	0.20	13.91
C3	0	59.51	229,075,000	0.083	1.70	26.64	28.34
	10	60.81	229,059,000	0.090	16.91	27.65	44.56
	20	63.32	229,082,000	0.080	37.20	19.24	56.44
	30	68.44	229,062,000	0.089	45.73	5.29	51.02
C4	0	67.43	252,424,000	0.312	4.55	384.27	352.83
	10	68.74	252,417,000	0.315	49.34	338.92	388.26
	20	72.08	252,297,000	0.363	103.86	174.37	278.23
	30	78.65	252,036,000	0.466	158.99	26.72	185.71
C5	0	72.88	244,933,000	0.266	8.16	864.29	872.44
	10	73.79	244,947,000	0.260	73.76	764.80	838.56
	20	75.84	245,023,000	0.229	177.57	446.31	623.88
	30	83.88	244,645,000	0.383	203.65	42.27	245.91

of the problem studied in this paper, they are meaningful because they represent millions of dollars, as can be seen from the values of Z^* in Tables 5 and 6. Considerable economic gains ranging between 2 and 6 million dollars are achieved if the proposed algorithm is used instead of the sequential heuristic. In terms of solution time, although the solution times of the proposed algorithm are somewhat long compared to those of the sequential heuristic (on average, the proposed algorithm runs 9–18 times longer than does the sequential heuristic), they are reasonable and still considerably smaller than those required by the commercial solver CPLEX to solve the stochastic integer program, as can be seen from the numerical results presented next.

Instead of solving the two-stage stochastic formulation (1)–(7), we solved a slightly different but equivalent formulation where the binary decision variables are defined as follows:

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

that is, it is $(x_i^t - x_i^{t-1})$ that specifies whether block i is mined in period t or not. This way of defining the variables has been proposed by Caccetta and Hill (2003) and has been shown to be computationally more efficient if one has to solve the problem using branch-and-cut methods, which is the case in the next experiments. With this definition of the variables, the two-stage stochastic model becomes:

$$\max \frac{1}{S}\left\{\sum_{s\in\Omega}\sum_{t\in T}\sum_{i\in N} v_{is}^t (x_i^t - x_i^{t-1}) - \sum_{s\in\Omega}\sum_{t\in T} c^t d_s^t\right\} \tag{24}$$

$$s.t. \quad x_i^{t-1} \le x_i^t \quad \forall i \in N, t \in T \tag{25}$$

$$x_i^t \le x_p^t \quad \forall i \in N, p \in P_i, t \in T \tag{26}$$

$$\sum_{i\in N} w_i (x_i^t - x_i^{t-1}) \le W^t \quad \forall t \in T \tag{27}$$

Table 5
Effect of grouping scenarios during Phase I - Copper instances (dataset D1).

Instance	Group size	%Convergence	Z^*($)	%Gap	Time (minutes)		
					Phase I	Phase II	Total
C1	3	71.04	165,523,000	0.010	0.81	0.03	0.85
	5	82.82	165,516,000	0.014	0.71	0.02	0.72
	7	85.26	165,496,000	0.026	0.77	0.02	0.78
	10	96.54	165,147,000	0.237	0.76	0.01	0.77
	20	100.00	164,079,000	0.882	0.07	0.00	0.07
C2	3	72.93	199,135,000	0.026	3.11	0.18	3.29
	5	83.16	199,133,000	0.027	3.30	0.07	3.37
	7	90.03	199,038,000	0.074	2.96	0.04	3.00
	10	94.36	198,521,000	0.334	2.39	0.03	2.42
	20	100.00	196,189,000	1.504	0.30	0.00	0.30
C3	3	75.79	229,001,000	0.116	14.93	3.75	18.68
	5	84.74	228,932,000	0.145	15.83	1.54	17.36
	7	87.82	228,899,000	0.160	14.72	0.66	15.38
	10	94.09	228,330,000	0.408	12.55	0.27	12.81
	20	100.00	227,279,000	0.867	1.18	0.00	1.18
C4	3	82.77	252,033,000	0.467	44.58	36.46	81.04
	5	89.00	251,672,000	0.609	52.59	5.37	57.97
	7	93.28	251,240,000	0.780	45.82	1.13	46.94
	10	96.80	250,886,000	0.920	30.93	0.64	31.57
	20	100.00	249,596,000	1.429	2.93	0.00	2.93
C5	3	86.31	244,860,000	0.296	70.52	57.74	128.26
	5	91.13	244,784,000	0.327	80.41	12.52	92.93
	7	93.63	244,528,000	0.431	67.70	3.29	70.99
	10	96.40	244,117,000	0.598	51.51	1.31	52.82
	20	100.00	242,818,000	1.127	5.49	0.00	5.49

Table 6
Effect of grouping scenarios during Phase I - Gold instances (dataset D2).

Instance	Group size	%Convergence	Z^*($)	%Gap	Time (minutes)		
					Phase I	Phase II	Total
G1	3	58.88	409,094,000	0.528	64.60	17.18	81.78
	5	66.98	409,050,000	0.539	107.01	9.29	116.29
	7	73.63	409,173,000	0.509	131.99	2.17	134.16
	10	84.57	408,941,000	0.565	102.67	0.95	103.63
	20	100.00	406,236,000	1.223	15.09	0.00	15.09
G2	3	63.02	440,556,000	0.777	139.68	133.23	272.91
	5	72.02	440,586,000	0.770	190.05	76.62	266.66
	7	76.87	440,737,000	0.736	210.56	38.40	248.96
	10	85.72	440,072,000	0.886	182.37	3.24	185.61
	20	100.00	434,445,000	2.154	30.54	0.00	30.54
G3	3	68.28	475,829,000	0.876	233.36	220.37	453.73
	5	75.98	475,819,000	0.878	339.46	81.19	420.64
	7	79.39	474,915,000	1.066	371.71	60.20	431.91
	10	85.75	474,842,000	1.082	382.16	20.73	402.89
	20	100.00	470,706,000	1.943	39.64	0.00	39.64
G4	3	71.44	483,017,000	1.026	304.21	474.77	778.97
	5	78.79	482,942,000	1.042	413.44	172.43	585.87
	7	83.40	481,973,000	1.240	546.31	95.57	641.89
	10	87.95	480,782,000	1.484	482.82	16.32	499.14
	20	100.00	477,333,000	2.191	52.79	0.00	52.79
G5	3	76.52	461,840,000	1.262	285.84	981.08	1266.92
	5	83.48	461,614,000	1.310	379.46	185.70	565.15
	7	87.89	460,563,000	1.535	407.17	73.99	481.16
	10	91.87	460,223,000	1.607	413.03	29.65	442.69
	20	100.00	457,702,000	2.146	34.29	0.00	34.29

$$\sum_{i\in N}\theta_{is}w_i(x_i^t - x_i^{t-1}) - d_s^t \le \Theta^t \quad \forall t \in T, s \in \Omega \tag{28}$$

$$x_i^0 = 0 \quad \forall i \in N \tag{29}$$

$$x_i^t \in \{0,1\} \quad \forall i \in N, t \in T \tag{30}$$

$$d_s^t \ge 0 \quad \forall t \in T, s \in \Omega. \tag{31}$$

CPLEX 12.2 was used to solve the formulation (24)–(31) with a time limit of 26 hours, which is larger than the largest computational time required by the proposed algorithm considering all tested instances and group sizes. The results of these tests are summarized in Table 7. For each instance, we recall its size (columns N and T) and then we give the value of the gap between Z_{LR}, the linear relaxation optimal value, and the value of the solution found by CPLEX within the time limit, as well as the time required to find this solution. A dash ("-") indicates that no feasible solution was obtained within the 26-hour time limit, while the symbol "$*$" indicates that the solution found by CPLEX is optimal. We also give in this table the results obtained by the proposed algorithm when group sizes 3, 5, 7, and 10 are considered, which we refer to as PA-G3, PA-G5, PA-G7, and PA-G10, respectively. Table 8

Table 7
Comparing the proposed algorithm to the branch-and-cut algorithm implemented in CPLEX.

			%Gap					Time (minutes)				
	N	*T*	CPLEX	PA-G3	PA-G5	PA-G7	PA-G10	CPLEX	PA-G3	PA-G5	PA-G7	PA-G10
C1	4273	3	0.005*	0.010	0.014	0.026	0.237	0.91	0.85	0.72	0.78	0.77
C2	7141	4	0.011*	0.026	0.027	0.074	0.334	30.67	3.29	3.37	3.00	2.42
C3	12,627	7	0.074	0.116	0.145	0.160	0.408	1560.00	18.68	17.36	15.38	12.81
C4	20,626	10	–	0.467	0.609	0.780	0.920	–	81.04	57.97	46.94	31.57
C5	26,021	13	–	0.296	0.327	0.431	0.598	–	128.26	92.93	70.99	52.82
G1	18,821	5	0.376	0.528	0.539	0.509	0.565	1560.00	81.78	116.29	134.16	103.63
G2	23,901	7	–	0.777	0.770	0.736	0.886	–	272.91	266.66	248.96	185.61
G3	30,013	8	–	0.876	0.878	1.066	1.082	–	453.73	420.64	431.91	402.89
G4	34,981	9	–	1.026	1.042	1.240	1.484	–	778.97	585.87	641.89	499.14
G5	40,762	11	–	1.262	1.310	1.535	1.607	–	1266.92	565.15	481.16	442.69

Table 8
Comparing the computational times of the proposed algorithm and the time required by CPLEX to solve the linear relaxation (LR).

			Time (minutes)				
	N	*T*	LR	PA-G3	PA-G5	PA-G7	PA-G10
C1	4273	3	0.27	0.85	0.72	0.78	0.77
C2	7141	4	6.40	3.29	3.37	3.00	2.42
C3	12,627	7	137.67	18.68	17.36	15.38	12.81
C4	20,626	10	1102.61	81.04	57.97	46.94	31.57
C5	26,021	13	2015.51	128.26	92.93	70.99	52.82
G1	18,821	5	175.79	81.78	116.29	134.16	103.63
G2	23,901	7	1351.41	272.91	266.66	248.96	185.61
G3	30,013	8	3051.09	453.73	420.64	431.91	402.89
G4	34,981	9	3843.19	778.97	585.87	641.89	499.14
G5	40,762	11	10560.22	1266.92	565.15	481.16	442.69

provides a comparison of solution times between the proposed algorithm and the linear relaxation.

Results presented in Table 7 clearly show the limitations of exact methods and the need for heuristic approaches, such as the one proposed in this paper, to deal with instances of realistic size. For the smallest instance, C1, the solution times of the proposed algorithm and CPLEX are comparable. For the other 9 instances, the proposed algorithm is clearly faster and its running time has a significantly smaller growth rate. CPLEX was able to solve the small instances C1 and C2 to optimality, while it could not find the optimal solution for the larger instances C3 and G1 within the 26-hour time limit. CPLEX could not even find a feasible solution for instances with more than 20,000 blocks (the largest instances C4, C5, G2, G3, G4, and G5). A near-optimal solution was obtained by the proposed algorithm for all tested instances, within significantly less time than the time required by CPLEX to solve the linear relaxation of the problems (cf. Table 8).

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 9. For each instance, 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, while the next four columns give the objective function values obtained by each method (Z^*) and the *%Gap* with respect to the upper bound provided by CPLEX. The last two columns report CPU times, in minutes, spent by STWH and BCA solving 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 Tables 5 and 6. The maximum run time for BCA and STWH is set to 10 hours (600 minutes), except for the instance G5 when the group size 3 is used in phase I, which is allowed 20 hours (1200 minutes) as it seems to be the hardest to solve. The best results obtained for each instance and each group size are indicated in bold.

Within the time limit, the results in Table 9 indicate that using STWH improves the performance of the algorithm considerably. More specifically, there are three cases that depend on the size of the restricted problem to be solved. When the size of the restricted problem is small (c.f. results for C1 and C2), BCA and STWH are comparable. However, when the size is large (c.f. results for G4 group size 3 and G5 group sizes 3 and 5), STWH dominates BCA both in terms of solution time and solution quality. In particular, for the largest instance G5, when group size 3 is used, the gap is reduced to 1.262 percent when STWH is used compared to 25.516 percent when BCA is used. For the remaining cases, STWH is 6 to 47 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 10 instances considered in this paper were solved with STWH without combining it with PH. In what follows, we refer to this solution approach as PSTWH (Pure STWH). The time limit for PSTWH was set to 26 hours, and the results obtained are summarized in Table 10, which has the same structure as Table 7. As in Table 7, a dash ("-") indicates that no feasible solution was obtained within the time limit.

As one would expect, PSTWH is computationally expensive. For instances with more than 20,000 blocks (C4, C5, G2, G3, G4, and G5), its performance is quite similar to the performance of CPLEX when solving the two-stage stochastic formulation (24)–(31), in the sense that both methods were not able to find a feasible solution within the 26-hour time limit. PSTWH also failed to solve the instance G1 within the time limit. For the smaller instances (C1, C2, and C3), PSTWH requires in general more time than the proposed algorithm. Although PSTWH obtains slightly better solutions, the difference in gap is not significant, especially if the proposed algorithm is used with group size 3 (on average, % Gap is 0.033 compared to 0.051). These results clearly highlight the significant benefit of the proposed algorithm combining PH with STWH.

4.4. Effect of accounting for metal uncertainty

The last part of the numerical tests addresses the question of the value of including metal uncertainty in the optimisation process. To this end, we use the so-called *Value of the Stochastic Solution* (VSS), which measures the expected gain from solving a stochastic model rather than its deterministic counterpart (Birge & Louveaux, 2011). More specifically, the problem

Table 9
Effect of using STWH in phase II instead of BCA implemented in CPLEX.

			Z^*($)		%Gap		Time phase II (minutes)	
	Group size	%Convergence	STWH	BCA	STWH	BCA	STWH	BCA
C1	3	71.04	165,523,000	**165,530,000**	0.010	**0.005**	**0.03**	0.04
	5	82.82	165,516,000	**165,518,000**	0.014	**0.013**	0.02	**0.01**
	7	85.26	165,496,000	**165,505,000**	0.026	**0.020**	0.02	**0.01**
	10	96.54	165,147,000	**165,151,000**	0.237	**0.234**	0.01	**0.005**
C2	3	72.93	199,135,000	**199,157,000**	0.026	**0.015**	0.18	**0.10**
	5	83.16	199,133,000	**199,145,000**	0.027	**0.020**	0.07	**0.05**
	7	90.03	199,038,000	**199,059,000**	0.074	**0.064**	0.04	**0.02**
	10	94.36	**198,521,000**	**198,521,000**	**0.334**	**0.334**	0.03	**0.01**
C3	3	75.79	229,001,000	**229,046,000**	0.116	**0.096**	**3.75**	50.65
	5	84.74	228,932,000	**228,980,000**	0.145	**0.125**	**1.54**	8.21
	7	87.82	228,899,000	**228,923,000**	0.160	**0.150**	**0.66**	4.27
	10	94.09	228,330,000	**228,448,000**	0.408	**0.357**	0.27	**0.11**
C4	3	82.77	252,033,000	**252,134,000**	0.467	**0.427**	**36.46**	600.00
	5	89.00	251,672,000	**251,785,000**	0.609	**0.565**	**5.37**	146.01
	7	93.28	251,240,000	**251,312,000**	0.780	**0.751**	**1.13**	7.21
	10	96.80	250,886,000	**250,989,000**	0.920	**0.879**	**0.64**	1.34
C5	3	86.31	244,860,000	**244,938,000**	0.296	**0.264**	**57.74**	600.00
	5	91.13	244,784,000	**244,895,000**	0.327	**0.281**	**12.52**	579.88
	7	93.63	244,528,000	**244,657,000**	0.431	**0.379**	**3.29**	70.02
	10	96.40	244,117,000	**244,185,000**	0.598	**0.571**	**1.31**	2.17
G1	3	58.88	409,094,000	**409,719,000**	0.528	**0.376**	**17.18**	292.94
	5	66.98	409,050,000	**409,694,000**	0.539	**0.382**	**9.29**	51.74
	7	73.63	409,173,000	**409,488,000**	0.509	**0.432**	**2.17**	4.98
	10	84.57	408,941,000	**409,473,000**	0.565	**0.436**	**0.95**	1.01
G2	3	63.02	440,556,000	**441,241,000**	0.777	**0.623**	**133.23**	600.00
	5	72.02	440,586,000	**441,180,000**	0.770	**0.637**	**76.62**	600.00
	7	76.87	440,737,000	**441,138,000**	0.736	**0.646**	**38.40**	600.00
	10	85.72	440,072,000	**440,726,000**	0.886	**0.739**	**3.24**	86.00
G3	3	68.28	475,829,000	**476,235,000**	0.876	**0.791**	**220.37**	600.00
	5	75.98	475,819,000	**476,479,000**	0.878	**0.741**	**81.19**	600.00
	7	79.39	474,915,000	**476,223,000**	1.066	**0.794**	**60.20**	600.00
	10	85.75	474,842,000	**476,154,000**	1.082	**0.808**	**20.73**	600.00
G4	3	71.44	**483,017,000**	468,449,000	**1.026**	4.011	**474.77**	600.00
	5	78.79	482,942,000	**483,462,000**	1.042	**0.935**	**172.43**	600.00
	7	83.40	481,973,000	**483,112,000**	1.240	**1.007**	**95.57**	600.00
	10	87.95	480,782,000	**481,794,000**	1.484	**1.277**	**16.32**	600.00
G5	3	76.52	**461,840,000**	348,393,000	**1.262**	25.516	**981.08**	1200.00
	5	83.48	**461,614,000**	432,246,000	**1.310**	7.589	**185.70**	600.00
	7	87.89	460,563,000	**461,197,000**	1.535	**1.399**	**73.99**	600.00
	10	91.87	460,223,000	**461,313,000**	1.607	**1.374**	**29.65**	600.00

Table 10
Comparing the proposed algorithm to PSTWH.

			%Gap					Time (minutes)				
	N	T	PSTWH	PA-G3	PA-G5	PA-G7	PA-G10	PSTWH	PA-G3	PA-G5	PA-G7	PA-G10
C1	4273	3	0.008	0.010	0.014	0.026	0.237	0.56	0.85	0.72	0.78	0.77
C2	7141	4	0.015	0.026	0.027	0.074	0.334	7.85	3.29	3.37	3.00	2.42
C3	12,627	7	0.077	0.116	0.145	0.160	0.408	370.82	18.68	17.36	15.38	12.81
C4	20,626	10	–	0.467	0.609	0.780	0.920	–	81.04	57.97	46.94	31.57
C5	26,021	13	–	0.296	0.327	0.431	0.598	–	128.26	92.93	70.99	52.82
G1	18,821	5	–	0.528	0.539	0.509	0.565	–	81.78	116.29	134.16	103.63
G2	23,901	7	–	0.777	0.770	0.736	0.886	–	272.91	266.66	248.96	185.61
G3	30,013	8	–	0.876	0.878	1.066	1.082	–	453.73	420.64	431.91	402.89
G4	34,981	9	–	1.026	1.042	1.240	1.484	–	778.97	585.87	641.89	499.14
G5	40,762	11	–	1.262	1.310	1.535	1.607	–	1266.92	565.15	481.16	442.69

is first solved by replacing the random parameters (the metal content) by their means to get a first-stage solution (a mining sequence, defined by the x_i^t). Then, the first-stage solution is fixed at that value, and the problem is solved for each scenario s to get the value of the objective function under each scenario ($Z_s = \sum_{t\in T}\sum_{i\in N} v_{is}^t x_i^t - \sum_{t\in T} c^t d_s^t$). Finally, the expected value of the so-obtained objective values Z_s is computed ($E[Z] = \frac{1}{S}\sum_{s\in\Omega} Z_s$). The *Value of the Stochastic Solution* is defined as the difference between the value of the two-stage stochastic problem solution, denoted by Z^* in the previous sections, and this expected value, $E[Z]$ ($VSS = Z^* - E[Z]$). It is well-known that $VSS \geq 0$ (see Birge and Louveaux (2011), for example). The objective of the numerical tests presented below is to confirm, as shown in all related studies mentioned in the introduction, that it is worthwhile to account for metal uncertainty when scheduling production of open-pit mines despite the additional complexity this approach might entail.

Table 11 shows results obtained for the 10 instances considered in this paper when the two-stage stochastic model is solved using PA-G3, PA-G5, PA-G7, and PA-G10; that is, the proposed algorithm combining PH and STWH with group sizes 3, 5, 7, and 10, respectively. In addition to the VSS values, Table 11 also provides the values of *%Gain*, indicating the relative percentage

Table 11
Value of the stochastic solution.

			VSS ($)				%Gain			
	N	*T*	PA-G3	PA-G5	PA-G7	PA-G10	PA-G3	PA-G5	PA-G7	PA-G10
C1	4273	3	1,641,000	1,634,000	1,614,000	1,265,000	1.00	1.00	0.98	0.77
C2	7141	4	2,416,000	2,414,000	2,319,000	1,802,000	1.23	1.23	1.18	0.92
C3	12,627	7	5,404,000	5,335,000	5,302,000	4,733,000	2.42	2.39	2.37	2.12
C4	20,626	10	6,162,000	5,801,000	5,369,000	5,015,000	2.51	2.36	2.18	2.04
C5	26,021	13	4,106,000	4,030,000	3,774,000	3,363,000	1.71	1.67	1.57	1.40
G1	18,821	5	87,417,000	87,373,000	87,496,000	87,264,000	27.18	27.16	27.20	27.13
G2	23,901	7	93,923,000	93,953,000	94,104,000	93,439,000	27.10	27.10	27.15	26.96
G3	30,013	8	97,272,000	97,262,000	96,358,000	96,285,000	25.70	25.69	25.45	25.43
G4	34,981	9	92,834,000	92,759,000	91,790,000	90,599,000	23.79	23.77	23.52	23.22
G5	40,762	11	83,102,000	82,876,000	81,825,000	81,485,000	21.94	21.88	21.60	21.51

increase in the objective function value resulting from solving the stochastic model rather than its deterministic counterpart. *%Gain* is calculated as follows:

$$\%Gain = \frac{Z^* - E[Z]}{E[Z]} \times 100.$$

Clearly, it pays off to use the stochastic solution rather than the mean value solution. The objective function value is 1–27 percent higher, and the average *%Gain*, over all instances, is around 13 percent no matter which group size is used when solving the stochastic model. The increase of 1 percent may not be seen as very substantial, but it should be noted that it represents significant benefits in the order of millions of dollars (c.f. columns *VSS*). The largest increases are found for the gold instances, G1-G5, and this is partly explained by the fact that these instances have a higher variability compared to the copper instances, C1-C5.

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. The potential benefits of solving the stochastic model over solving the deterministic model in which the random parameters are replaced by their expected values have also been highlighted. The results of the tests indicate that it would pay off by millions of dollars (1–27 percent increase in the value of the objective function) to use the stochastic solution rather the mean value solution.

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 that include more operational constraints, such as variable cut-off grade, grade blending, and stockpiling, 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. Finally, another important research direction is the development of other efficient solution approaches. Since it has been observed empirically that the problem formulation often achieves small integrality gaps, one approach could be to solve the linear relaxation of the problem using an efficient algorithm and then to use an LP-rounding procedure to get an integer solution.

Acknowledgments

The work in this paper was funded by NSERC CRDPJ 411270-10, NSERC Discovery Grant 239019-06, and the industry members of the COSMO Stochastic Mine Planning Laboratory: AngloGold Ashanti, Barrick Gold, BHP Billiton, De Beers, Newmont Mining, and Vale. This support is gratefully acknowledged. The authors would like to thank three anonymous referees for their valuable comments and suggestions that helped improve the paper.

References

Albor, F., & Dimitrakopoulos, R. (2009). Stochastic mine design optimization based on simulated annealing: pit limits, production schedules, multiple orebody scenarios and sensitivity analysis. *IMM Transactions, Mining Technology, 118*(2), 80–91.

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.

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.

Behrang, K., Hooman, A., & Clayton, D. (2014). A linear programming model for long-term mine planning in the presence of grade uncertainty and a stockpile. *International Journal of Mining Science and Technology, 24*, 451–459.

Bertsekas, D. (1982). *Constrained optimization and Lagrange multipliers methods*. New York: Academic Press.

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. (2011). *Introduction to stochastic programming, Second Edition*. 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 Online*. [Last accessed: 31.01.12].
Boucher, A., & Dimitrakopoulos, R. (2009). Block simulation of multiple correlated variables. *Mathematical Geosciences, 41*(2), 215–237.
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.
Chatterjee, S. (2014). Stochastic production scheduling - solution through lagrangian relaxation and the branch-and-cut algorithm. In *Proceedings of orebody modelling and strategic mine planning symposium 2014, the australasian institute of mining and metallurgy* (pp. 323–328).
Chiles, J., & Delfiner, P. (2012). *Geostatistics: modeling spatial uncertainty, Second ed*. New Jersey: John Wiley & Sons.
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. (2011). Stochastic optimization for mine planning: a decade of developments. *Journal of Mining Science, 47*, 138–150.
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.
Fricke, C., Velletri, P., & Wood, R. (2014). Enhancing risk management in strategic mine planning through uncertainty analysis. In *Proceedings of orebody modelling and strategic mine planning symposium 2014, the australasian institute of mining and metallurgy* (pp. 275–279).
Godoy, M. (2002). *The effective management of geological risk*. Qld, Australia: University of Queensland Ph.D. thesis.
Goovaerts, P. (1997). *Geostatistics for Natural Resources Evaluation*. New York: Oxford University Press.
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.
Horta, A., & Soares, A. (2010). Direct sequential co-simulation with joint probability distributions. *Mathematical Geosciences, 42*(3), 269–292.
Lagos, G., Espinoza, D., Moreno, E., & Amaya, J. (2011). Robust planning for an open-pit mining problem under ore-grade uncertainty. *Electronic Notes in Discrete Mathematics, 37*, 15–20.
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*.
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.
Maleki, M., & Emery, X. (2015). Joint simulation of grade and rock type in a stratabound copper deposit. *Mathematical Geosciences, 47*(4), 471–495.
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.
Newman, A. M., Rubio, E., Caro, R., Weintraub, A., & Eurek, E. (2010). A review of operations research in mine planning. *Interfaces, 40*, 222–245.
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.
Rossi, M., & Deutsch, C. (2014). *Mineral resource estimation*. New York: Springer.
Whittle, J. (2015). Money mining - value extraction on a vast scale. http://www.whittleconsulting.com.au. [Accessed: 15.04.15].