Lagrangian relaxation of the MILP open pit production scheduling formulation

Mohammad Mahdi Badiozamani and Hooman Askari-Nasab
Mining Optimization Laboratory (MOL)
University of Alberta, Edmonton, Canada

Abstract

The optimization models for open pit mine planning have been developed in the literature under various assumptions. In a typical mathematical model, the number of variables directly relates to the number of blocks in the optimized final pit and the number of planning time periods. These variables are either integer, continuous, or a mixture of both types. However, most of the open pit mine block models consist of millions of blocks. In addition, a multi-period long-term planning time horizon over the mine-life should be considered for the optimal solution. As a result in almost all the cases, the real world problems have a large number of variables that makes the problem unsolvable with current in-hand optimization software or solvable with a very long solution time. In order to tackle such problems, two approaches are developed; (1) creating mining cuts, which reduces the number of variables by aggregating the blocks into larger clusters and reducing the problem size, and (2) defining Lagrange multipliers which decreases the number of constraints and provides an upper/lower bound for the problem. In this research, the focus is on the second group of methods, using Lagrange multipliers. The mixed integer linear programming (MILP) formulation that is presented by Askari-Nasab and Awuah-offei (2008) is considered as the basis for the mine production scheduling and is used to find the relaxed forms of the MILP. An algorithm, based on the sub-gradient method is developed to find the proper Lagrange multipliers. The algorithm is applied to a data set and the results are reported.

1. Introduction

Mine planning has been among the most dominant aspects of decision making in the mining industry. It is important, because the extraction of ore, using the expensive equipment and work force, especially in remote areas is very cost sensitive. On the other hand, the mine planning is considered as one of the branches of planning and scheduling literature that has been studied for some decades by the academia. This joint subject of interest has resulted in development of many related issues in the mining and operations research contexts and as a result, the problem is well modeled as an MILP since 1970s. The number of variables in such MILP models is directly related to the number of blocks and the number of time periods considered in planning. Since in real world problems, usually thousands of blocks are considered in the optimum pit and for long-term planning, multiple periods are taken into account, the problem has millions of variables, among them some integer and some continuous. This makes the problem NP-hard and non solvable with current in hand software or solvable, but with a very long solution time. In a brief review, the literature regarding the methodology used in finding the solution, can be classified into two main categories; (1) those based on the exact solutions, mainly relying on linear programming (LP) to find the exact optimal solution that in most cases take a very long solution time (cpu time), and (2) those based on the approximation of optimal solution, applying the heuristic and meta heuristic

algorithms to find the best solution. Using the heuristic approach, some quality solutions can be found in reasonable time, but the solutions are not necessarily optimal.

Most of researchers who have tried to find the MILP optimal solutions using exact methods have relaxed the binary nature of integer variables. For example, Tan and Romani (1992) consider the equipment capacity constraints and find the optimal extraction schedule over multiple periods, using both linear programming (LP) and dynamic programming (DP). Since they relax the integer nature of decision variables, the block sequencing constraints are not satisfied. Fytas *et al.* (1993) use different approaches for long-term and short-term decisions. They employ simulation to find those blocks that are extracted in long-term. At this stage, they consider the precedence constraints, as well as the minimum and maximum production and processing capacity constraints and the bounds on the grade of entering material to the processing plant. Then, they use LP for the short-term planning, subject to the same constraints of long-term, but they assume that the partial extraction of blocks is permitted. Finally, they propose an iterative approach to deliver a practical mining sequence.

As an example of meta heuristics application, Denby and Schofield (1994) use genetic algorithms to solve the large integer programs efficiently. They integrate two decision making problems together; determining the final open pit and extraction schedule. They consider schedules as a combination of final open pit and extraction schedule. They use a genetic algorithm, applying the concepts of mutations and crossover, to find the best schedule among mentioned combinations. Although they produce good solutions for small size problems, their algorithm does not work in reasonable time for large size problems.

Apart from meta heuristics, in order to reduce the problem size and make the large problems solvable using exact methods of operations research, some researchers have employed the idea of aggregating the blocks. The idea is to merge the blocks to create "*mining-cuts*" and hence, reduce the number of MILP variables. Askari-Nasab *et al.* (2010) propose a two stage algorithm to create mining-cuts. Their aim is to reduce the number of dependencies among the blocks/cuts in consecutive benches. They introduce a similarity index that is the basis for aggregation of neighboring blocks/cuts and continue the aggregation to reach to a predefined number of cuts in each bench (hierarchical clustering stage). Then they apply a Tabu Search algorithm to reduce the number of arcs, i.e. precedence relations, as much as possible by changing the borders of mining-cuts that are produced in previous stage (Tabu Search stage).

In 1980s, some researchers started to reduce the problem size, not by reducing the number of variables, but by relaxing some of the constraints. They used Lagrangian relaxation as an exact approach to find the optimal solution. The main idea in the Lagrangian relaxation approach is to relax some of the constraints and instead, add required penalty terms into the objective function. The constraints of the MILP can be classified into two categories: (1) hard constraints, those that define the precedence of blocks extraction and (2) soft constraints, including those constraints that are defined to satisfy the limited production and processing capacities and grade bounds for the material entering the processing plant. In most of the papers, the soft constraints are relaxed and corresponding terms are added into the objective function with a penalty multiplier. This relaxation arises another question: how much the objective function should be penalized if the corresponding constraint was not satisfied? In other words, what are the proper values for Lagrangian multipliers? Some algorithms are developed to find the answer to such question. Among them, one of non-heuristic algorithms is the sub-gradient method.

Dagdelen and Johnson (1986) use the Lagrangian relaxation to relax some of the constraints, e.g. constraints on the maximum production in each period, and placing such constraints with associated multipliers in the objective function. They apply the sub-gradient method to update the multipliers in some small examples. Their work is one of the first papers in the field of Lagrangian relaxation, using the sub-gradient method. Akaike and Dagdelen (1999) extend this work by

changing the value of the Lagrangian multipliers in an iterative procedure. The procedure continues until the relaxed problem reaches the optimal solutions that are feasible for the original problem.

Kawahata (2006) expands (Dagdelen and Johnson, 1986) work by defining a variable cut off grade that specifies the destination of extracted material, i.e. whether to go to the processing plant, to be stockpiled or to be dumped as waste. Then he uses two Lagrangian relaxation sub problems to find the bounds of the original problem; one for the most conservative case of mine sequencing and the other one for the most aggressive case. Since the solution for the relaxed problem is not necessarily feasible for the original problem, he adjusts some bounds on capacities to guarantee the feasibility of Lagrangian solution for the original problem.

Gaupp (2008) presents the MILP formulation for the open pit mine production planning problem. He proposes three approaches to make the MILP model tractable for large scale problems as (1) applying deterministic variable reduction techniques to eliminate some of the blocks, (2) producing cuts to strengthen the model formulation and (3) employing Lagrangian relaxation method using sub-gradient algorithm to reduce the problem size by eliminating some constraints. In Gaupp's (2008) proposed algorithm, to find the multiplier values corresponding to relaxed constraints, he develops a feasing routine to make the relaxed problem solution feasible for the original problem. However, Gaupp (2008) disregards the multi-element case in his model and solves the MILP model for a single element.

Newman *et al.* (2010) review those papers that have used operations research in mine planning. They specify that dynamic programming is not tractable for large size problems, while the Lagrangian approaches remain as a reliable method for solving large size problems. The Lagrangian methods also theoretically provide an optimal solution.
The structure of the paper is as follows; section 2 covers the problem definition and formulation. The theoretical framework that is the basis for the Lagrangian relaxation, in addition to the proposed algorithm is presented in section 3. A case study is described in section 4 and the results from implementation of the algorithm on the case are reported in section 5. Conclusions and future works are covered in section 6.

2. Problem definition

It is assumed that the optimal open pit has been determined previously and the block model is considered as the input to our algorithm. In addition, we use the 4th MILP model and the corresponding notation, presented by Askari-Nasab and Awuah-offei (2008) which is a long-term planning problem. In this model, it is assumed that the mining and processing are both at mining-cut level (not at block level), in order to reduce the problem size.
The notation of sets, indices, parameters and decision variables are as follows:

2.1. Sets

$\mathcal{K} = \{1,...,K\}$	set of all mining-cuts in the model.
$H(S)$	for each mining-cut c_k, there is a set $H(S) \subset \mathcal{K}$ defining the immediate predecessor cuts that must be extracted prior to extracting mining-cut k, where S is the total number of cuts in set $H(S)$.

2.2. Indices

A general parameter *f* can take three indices in the format of $f_k^{e,t}$, where:

$t \in \{1,\ldots,T\}$	index for scheduling periods.
$k \in \{1,\ldots,K\}$	index for mining-cuts.

$e \in \{1,\ldots,E\}$ index for elements of interest in each block/mining-cut.

2.3. Parameters

v_k^t the discounted revenue generated by selling the final product within mining-cut k in period t minus the extra discounted cost of mining all the material in block n as ore and processing it.

q_k^t the discounted cost of mining all the material in mining-cut k as waste.

g_k^e average grade of element e in ore portion of mining-cut k.

$gu^{e,t}$ upper bound on acceptable average head grade of element e in period t.

$gl^{e,t}$ lower bound on acceptable average head grade of element e in period t.

o_k ore tonnage in mining-cut k.

w_k waste tonnage in mining-cut k.

pu^t upper bound on processing capacity of ore in period t (tonnes).

pl^t lower bound on processing capacity of ore in period t (tonnes).

mu^t upper bound on mining capacity in period t (tonnes).

ml^t lower bound on mining capacity in period t (tonnes).

2.4. Decision Variables

$y_k^t \in [0,1]$ continuous variable, representing the portion of mining-cut c_k to be mined in period t, fraction of y characterizes both ore and waste included in the mining-cut.

$b_k^t \in \{0,1\}$ binary integer variable controlling the precedence of extraction of mining-cuts. b_k^t is equal to one if extraction of mining-cut c_k has started by or in period t, otherwise it is zero.

$s_k^t \in [0,1]$ continuous variable, representing the portion of mining-cut c_k to be extracted as ore and processed in period t.

2.5. Problem formulation

The original MILP problem, referred as the primal problem (P), is as follows:

(P):

$$\max \sum_{t=1}^{T}\sum_{k=1}^{K}(v_k^t \times s_k^t - q_k^t \times y_k^t) \quad (1)$$

Subject to:

$$gl^{t,e} \le \sum_{k=1}^{K} g_k^e \times o_k \times s_k^t \Big/ \sum_{k=1}^{K} o_k \times s_k^t \le gu^{t,e} \qquad \forall t \in \{1,...,T\}, \quad e \in \{1,...,E\} \tag{2}$$

$$pl^t \le \sum_{k=1}^{K} o_k \times s_k^t \le pu^t \qquad \forall t \in \{1,...,T\}, \quad e \in \{1,...,E\} \tag{3}$$

$$ml^t \le \sum_{k=1}^{K} (o_k + w_k) \times y_k^t \le mu^t \qquad \forall t \in \{1,...,T\} \tag{4}$$

$$s_k^t \le y_k^t \qquad \forall k \in \{1,...,K\}, \quad t \in \{1,...,T\} \tag{5}$$

$$b_k^t - \sum_{i=1}^{t} y_s^i \le 0 \qquad \forall k \in \{1,...,K\}, \quad t \in \{1,...,T\}, \quad s \in H(S) \tag{6}$$

$$\sum_{i=1}^{t} y_k^i - b_k^t \le 0 \qquad \forall k \in \{1,...,K\}, \quad t \in \{1,...,T\} \tag{7}$$

$$b_k^t - b_k^{t+1} \le 0 \qquad \forall k \in \{1,...,K\}, \quad t \in \{1,...,T-1\} \tag{8}$$

Eq. (1) presents the objective function, which is the maximization of the profit, i.e. the difference between discounted revenue and cost. Eqs. (2) to (4) control the grade blending, processing capacity, and mining capacity. Eq. (5) represents inequalities that ensure the amount of ore that is sent to the processing plant is less than or equal to the total amount of mined material. For each mining-cut k, Eqs. (6) to (8) check the set of immediate predecessor cuts that must be extracted prior to extracting mining-cut k. Fig. 1 illustrates the set $H(S) = \{B, C\}$, the predecessor mining-cuts that must be extracted prior to extraction of mining-cut A.

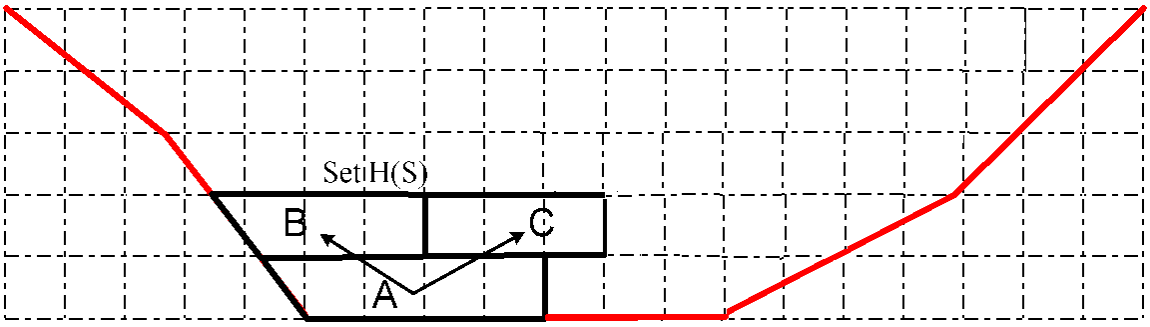

Fig. 1. Precedence of extraction in proposed MILP model.

According to Lagrangian relaxation approach, some of the constraints should be relaxed and placed in the objective function, with proper Lagrangian multipliers as the penalty for violation of relaxed constraints. The multipliers in our notation are called *lambda*, taking three indices in the format of $\lambda_n^{e,t}$ where:

$t \in \{1,....,T\}$ index for scheduling periods.

$n \in \{1,...,6\}$ index for the relaxed constraint's type.

$e \in \{1,...,E\}$ index for elements of interest in each mining-cut.

Here, we just study the relaxation of some or all of the soft constraints. As mentioned previously, soft constraints are the maximum and minimum of production capacity, processing capacity and acceptable average head grade of material entering into the processing plant in each period. This combination results in six constraint types, so $n \in \{1,...,6\}$. We have left the hard constraints, which control the precedence relationship between mining-cuts, to guarantee that the optimal solution of the relaxed problem does not violate the precedence constraints.

The six soft constraints generate 63 different relaxation combinations for the relaxed problem, in each one, just one or some of constraints out of six are relaxed. Only seven important combinations are considered as follows:

(D1) Dual problem; all soft constraints (grade, mining and processing) are relaxed.

(D2) Dual problem; grade and mining constraints are relaxed.

(D3) Dual problem; processing and mining constraints are relaxed.

(D4) Dual problem; processing and grade constraints are relaxed.

(D5) Dual problem; mining constraints are relaxed.

(D6) Dual problem; processing constraints are relaxed.

(D7) Dual problem; grade constraints are relaxed.

The first Dual MILP problem, named as (D1) is as follows:

(D1):

$$\begin{aligned}
&\max \sum_{t=1}^{T}\sum_{k=1}^{K}\left(\sum_{n\in c_k} v_k^t \times s_n^t - q_k^t \times y_k^t\right) \\
&+\sum_{t=1}^{T}\lambda_1^{e,t}(gu^{t,e}\sum_{k=1}^{K}o_k\times s_k^t - \sum_{k=1}^{K}g_k^e\times o_k\times s_k^t) + \sum_{t=1}^{T}\lambda_2^{e,t}(\sum_{k=1}^{K}g_k^e\times o_k\times s_k^t - gl^{t,e}\sum_{k=1}^{K}o_k\times s_k^t) \\
&+\sum_{t=1}^{T}\lambda_3^{t}(pu^t - \sum_{k=1}^{K}o_k\times s_k^t) + \sum_{t=1}^{T}\lambda_4^{t}(\sum_{k=1}^{K}o_k\times s_k^t - pl^t) \\
&+\sum_{t=1}^{T}\lambda_5^{t}(mu^t - \sum_{k=1}^{K}(o_k+w_k)\times y_k^t) + \sum_{t=1}^{T}\lambda_6^{t}(\sum_{k=1}^{K}(o_k+w_k)\times y_k^t - ml^t)
\end{aligned} \tag{9}$$

Subject to:

$$s_k^t \le y_k^t \qquad \forall k \in \{1,...,K\},\quad t \in \{1,...,T\} \tag{10}$$

$$b_k^t - \sum_{i=1}^{t} y_s^i \le 0 \qquad \forall k \in \{1,...,K\},\quad t \in \{1,...,T\},\quad s \in H(S) \tag{11}$$

$$\sum_{i=1}^{t} y_k^i - b_k^t \le 0 \qquad \forall k \in \{1,...,K\},\quad t \in \{1,...,T\} \tag{12}$$

$$b_k^t - b_k^{t+1} \le 0 \qquad \forall k \in \{1,...,K\},\quad t \in \{1,...,T-1\} \tag{13}$$

Since the (P) is a maximization problem, any feasible solution for the (P) is considered as a lower bound (LB) for the optimal objective function of (P). On the other hand, since we have relaxed some constraint in the formulation of (D), the optimal solution for the (D) should be considered as an upper bound (UB) for the optimal objective function of (P).

The main question is how to find the Lagrangian multipliers, $\lambda_n^{e,t}$, in a way that the corresponding UB and LB lay in a predefined desired gap interval, where gap is defined as the difference between UB and LB as (UB-LB)/UB

3. Theoretical framework and methodology

Let us assume a maximization problem with some constraints. We name this original problem as the *primal* problem. In addition, suppose that the constraints have made the primal problem so complicated that it takes a long run time to find the optimal solution. One practical way to tackle the complexity of the problem is to relax some of the constraints and consider the relaxed version of the primal problem as the *dual* problem. Since the feasible space corresponding to the dual problem is larger than the feasible space of the primal problem, the optimal solution to the dual problem always equals, or is greater than the optimal solution to the primal problem. In other words, the dual optimal solution is always considered as an upper bound to the primal optimal solution.

On the other hand, suppose that we manage to find just a feasible solution to the primal problem. Since by definition, the optimal solution to any maximization problem is greater than, or equal to any point in the feasible space of the problem, any feasible solution is considered as a lower bound to the optimal solution of the primal problem. The idea of upper bound and lower bound works for any minimization problem as well; the only difference is that any feasible solution to the primal problem is considered as an upper bound, while any optimal solution to the relaxed problem is considered as a lower bound for the optimal solution of the primal problem. This concept is illustrated in Fig. 2 for a maximization problem.

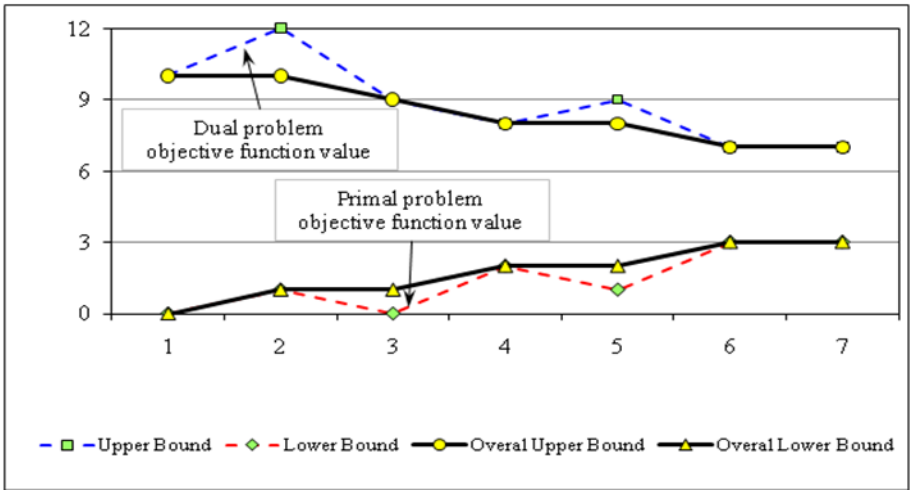

Fig. 2. Bounds for a typical maximization problem.

In Lagrangian relaxation, one or a number of constraints are relaxed to reduce the complexity of the primal problem. In return, in order to avoid the violations from relaxed constraints, proper penalty terms are added up to the objective function to penalize any violation from the corresponding constraints. For any penalty term, a penalty multiplier also called the Lagrangian multiplier, is considered and multiplied to the penalty term. One of the most well known and typical approaches used to find the multiplier values is the sub-gradient method. In the next section, a generic sub-gradient algorithm is briefly discussed.

3.1. The sub-gradient method

Fisher (1985) provides an application oriented guide to Lagrangian relaxation. He presents the following notation and formulation for typical primal and dual problems and the sub-gradient method.

The integer programming problem, called the primal is formulated as:

$$Z = \max cx \tag{14}$$

Subject to:

$$\begin{aligned} &Ax \le b \\ &Dx \le e \\ &x \ge 0, \text{int} \end{aligned} \tag{15}$$

Where x is $n \times 1$, b is $m \times 1$, e is $k \times 1$ and all other matrices have conformable dimensions. In this formulation, the constraints are partitioned into two sets, $Ax \le b$ and $Dx \le e$. It is assumed that it is easy to solve the primal problem, if the set of constraints $Ax \le b$ are relaxed. This relaxation produces the dual problem $LR_u^{\,k}$, using an *m* vector of non-negative multipliers *u,* as:

$$Z_D(u) = \max cx + u(b - Ax) \tag{16}$$

Subject to:

$$\begin{aligned} &Dx \le e \\ &x \ge 0, \text{int} \end{aligned} \tag{17}$$

Since the dual problem provides an upper bound for the primal maximization problem, ideally the vector *u* should be found in a way that $Z_D(u)$ be minimized;

$$Z_D = \min Z_D(u) \tag{18}$$

Eq. (18) is considered as the basis of the sub-gradient method. The goal is to find the proper set of Lagrangian multipliers that minimizes $Z_D(u)$. Multiplier values are updated, considering the initial value of u^0 and according to Eq. (19).

$$u^{k+1} = \max\{0, u^k - t_k(b - Ax^k)\} \tag{19}$$

Where x^k is the optimal solution to $LR_u^{\,k}$, the Lagrangian problem with dual variables set to u^k, and t_k is a scalar step size value. According to (Fisher, 1985), a formula for t_k that has been proven to be effective in practice is given by Eq. (20).

$$t_k = \frac{\lambda_k (Z_D(u^k) - Z^*)}{\sum_{i=1}^{m} (b_i - \sum_{j=1}^{n} a_{ij} x_j^{\,k})^2} \tag{20}$$

In this formula, Z^* is the objective value of the best known feasible solution to (P) and λ_k is a scalar chosen between 0 and 2. Frequently, the sequence λ_k is determined by starting with $\lambda_k = 2$

and reducing it by a factor of two whenever $Z_D(u^k)$ has failed to decrease in a specific number of iterations.

3.2. Proposed algorithm

The following algorithm is developed for implementation of the sub-gradient method to find Lagrangian multipliers for a minimization case (the objective function id multiplied by -1):

Step 1: Assign initial values for multipliers (u^0).

Step 2: Assign a (P) feasible solution as UB and a (D) optimal solution as LB.

Step 3: Update multipliers and solve (D).

Step 4: If (D) solution results in a better LB, update LB.

Step 5: If (D) solution is feasible for (P) and results in a better UB, update UB. Otherwise, do the feasing sub-routine and repeat step 5 if possible.

Step 6: If the stopping criteria are true, stop and report u^k and UB-LB gap, otherwise, go to step 3.

The flowchart of the proposed algorithm is illustrated in Fig. 3.

Since the solutions of (D), corresponding to different sets of multipliers are not necessarily feasible for (P), in most cases the UB remains unchanged, meaning that no feasible solution is found for the (P) with a better UB. This may cause the algorithm to gain no benefit from a large number of iterations. In some cases, making few changes to the (D) solution makes it feasible for (P). To do so, whenever the dual's solution is not feasible for the primal problem, the feasing sub-routine is called. However, it may happen that even after several attempts to make a solution feasible for the primal problem, the constraints still are violated and so, that specific iteration ends without any feasible solution.

Feasing sub-routine is as follows:

Step 1: sort the objective function coefficients and corresponding variable values in descending order and identify the relaxed constraint.

Step 2: select the m first ($F_i, i=1,...,m$) and n last ($L_j, j=1,...,n$) variables from the sorted list that have relaxed constraint multiplier.

Step 3: find the maximum possible change increments for each of $m+n$ variables as:

$h(F_i) = F_i / h$ and $h(L_j) = (1-L_j) / h$, where h is the change increment.

Step 4: Update $m+n$ variable values by one increment, i.e. $F_{i(revised)} = F_i - h$ and $L_{j(revised)} = L_j + h$

Step 5: Check the feasibility of revised solution. If it is feasible, send it to the main algorithm, otherwise change variable values by one more increment (if possible) and go to step 4.

The flowchart for the feasing sub-routine is illustrated in Fig. 4.

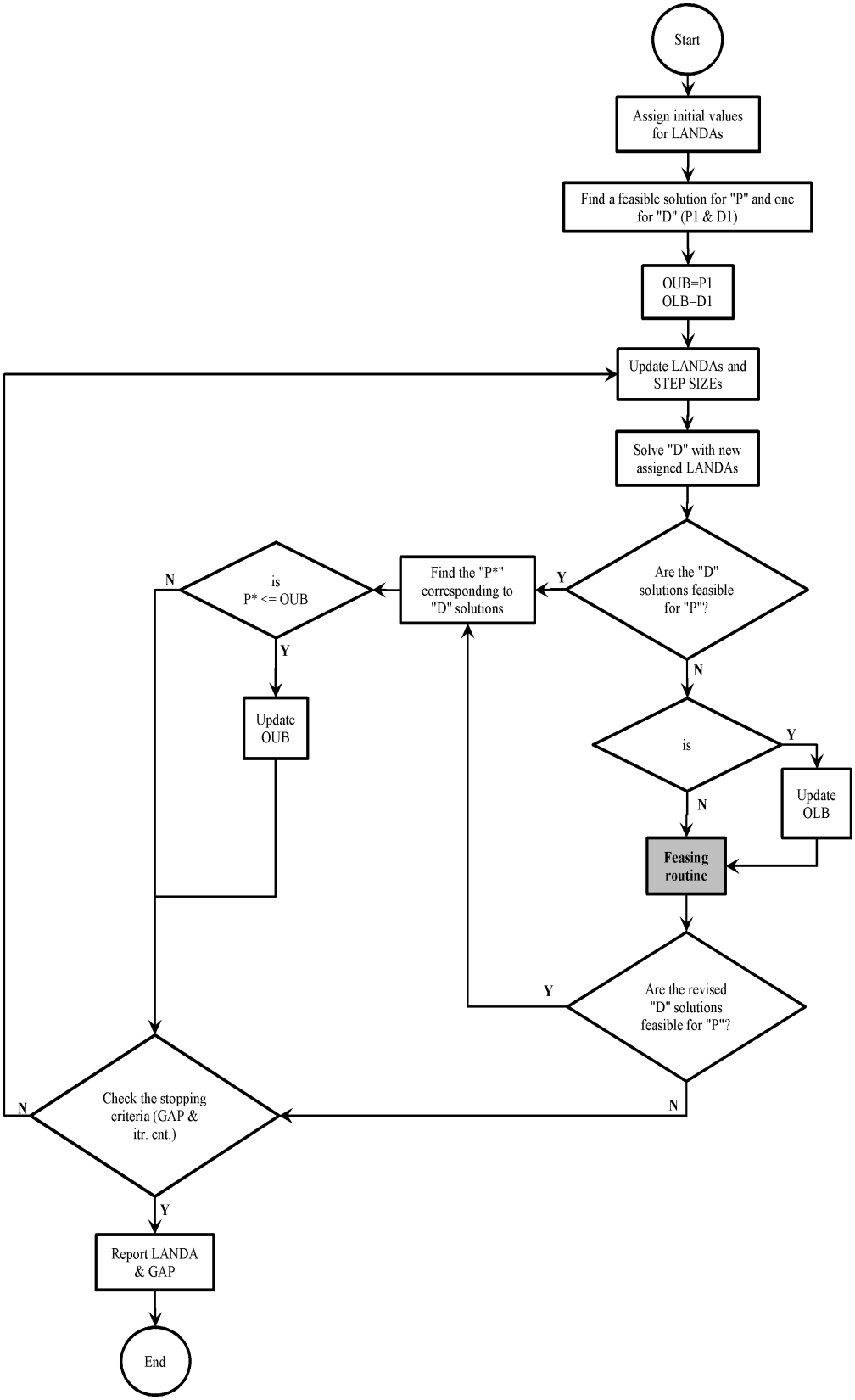

Fig. 3. Sub-gradient algorithm flowchart.

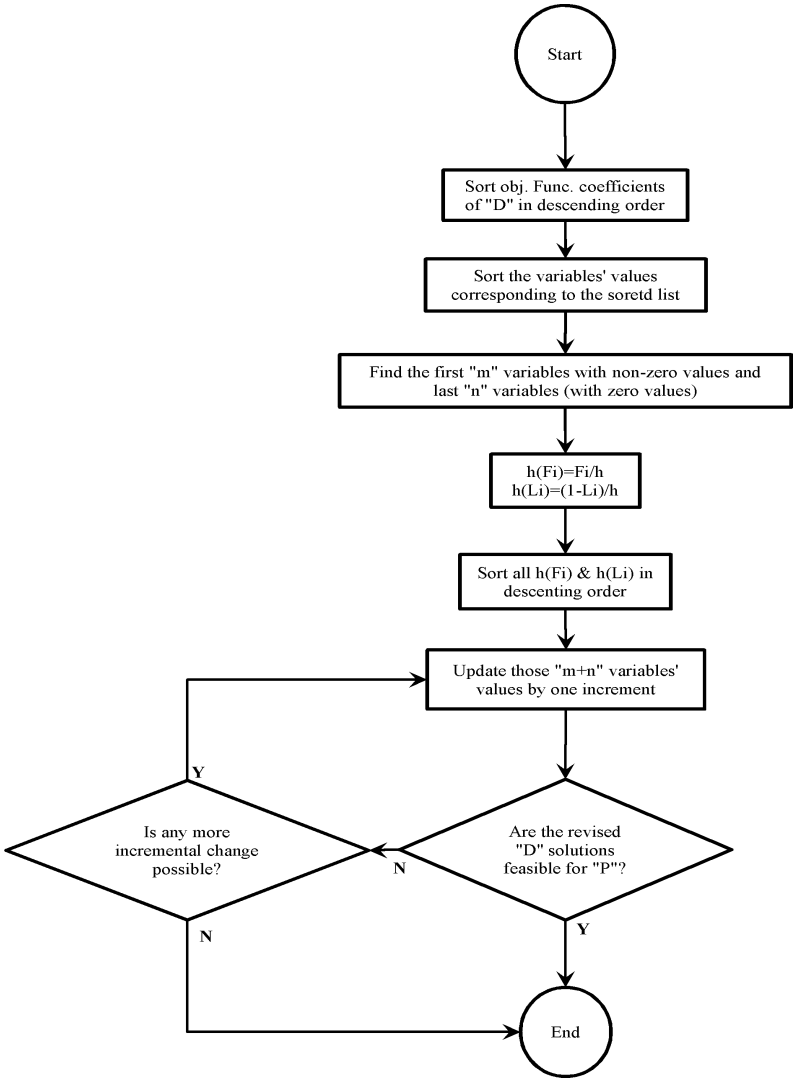

Fig. 4. Proposed feasing sub-routine flowchart.

4. Case study

To implement the proposed algorithm, a block model data set, containing 2598 blocks (aggregated into 148 miming-cuts) is considered. The mining-cuts are defined in seven mining benches. Some important specifications of the input data are presented in Table 1.

Table 1. Case study specifications.

number of blocks:	2598	max & min of mining capacity (10^6 tons):	(0.0,13.2)
number of mining-cuts:	148	max & min of processing capacity (10^6 tons):	(0.0,7.15)
interest rate:	0.1	max & min of acceptable grade for Iron ore (%):	(0.65,0.80)
number of elements:	3	max & min of acceptable grade for Sulfur (%):	(0.0,0.018)
Planning for:	12 periods	max & min of acceptable grade for Phosphor (%):	(0.0,0.014)

The algorithm is executed for 50 iterations. The detailed input parameters to the sub-gradient algorithm and the feasing sub-routine are presented in Table 2.

Table 2. Input parameters to the sub-gradient algorithm and feasing sub-routine.

Stopping criteria	UB/LB convergence: % 0.01
	maximum iterations: 50
Dual type	D5 (mining capacity constraints are relaxed)
initial multiplier values	1
initial scalar value	2 (used in step size formula)
m / n / h	500 / 500 / 0.05 (used in feasing)

5. Results and discussion

To execute the algorithm for the presented case study, a MATLAB program (MathworkInc., 2009) is developed. The code calls the TOMLAB/CPLEX (ILOGInc., 2007) in each iteration to solve the relaxed MILP problem. Fig. 5 to Fig. 8 Show two cross sections and two plan views of the resulting extraction sequence, corresponding to the relaxed problem in the last iteration. The units in figures are in meter and the size of each block is 15 by 25 by 50 cubed meters.

Fig. 5. Sample cross section, looking East. Each block 15m.

Fig. 6. Sample cross section, looking North, Each block 15m.

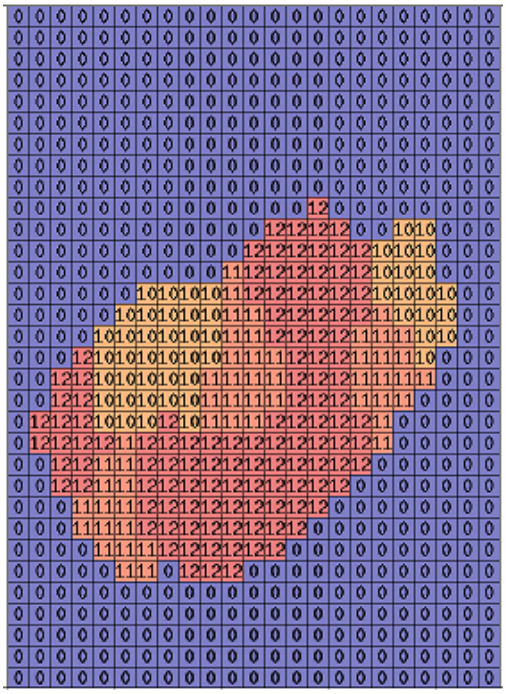

Fig. 7. Sample plan view, 1st bench.

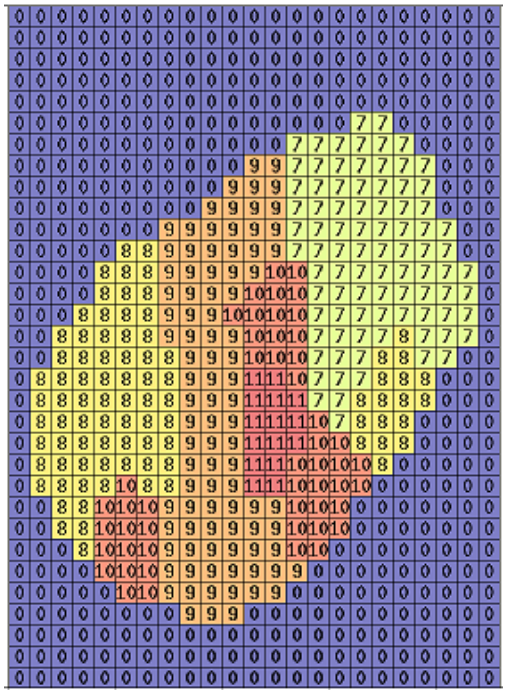

Fig. 8. Sample plan view, 3rd bench.

The extraction sequences in each iteration result in a specific amount of rock to be mined in each period and a specific amount of ore to be entered into the processing plant. Fig. 9 to Fig. 12 show the results, corresponding to four iterations. In these figures, the yellow and green lines represent the maximum processing and mining capacities, respectively. The run time for 50 iterations of algorithm on a machine with 3.00 GB of RAM and 2.40 GHz CPU is 675 seconds.

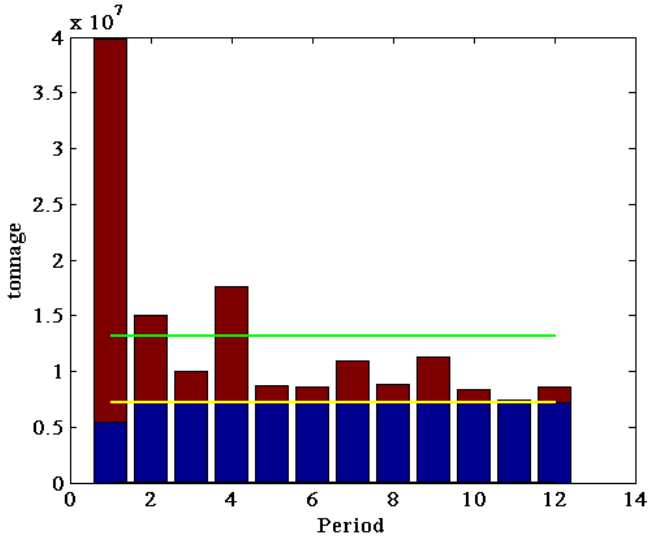

Fig. 9. Total mined and processed material, iteration 1.

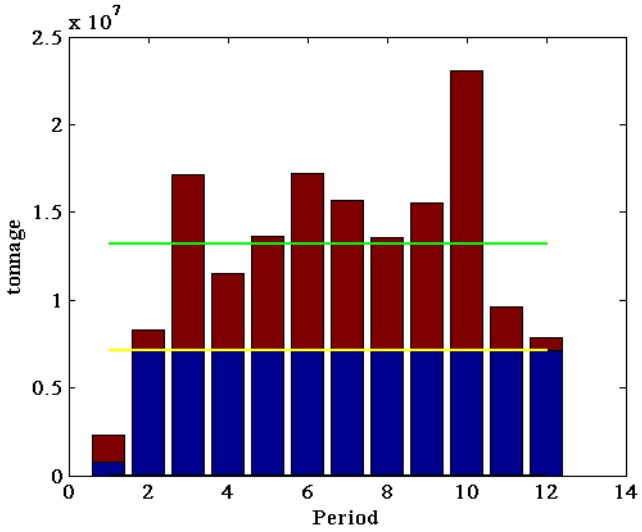

Fig. 10. Total mined and processed material, iteration 10.

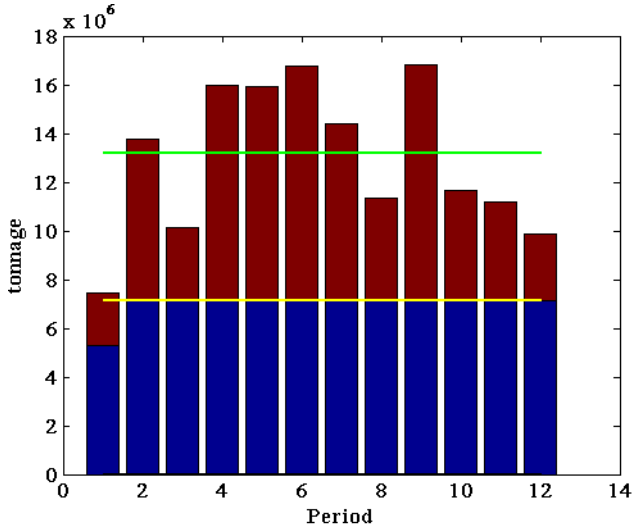

Fig. 11. Total mined and processed material, iteration 30.

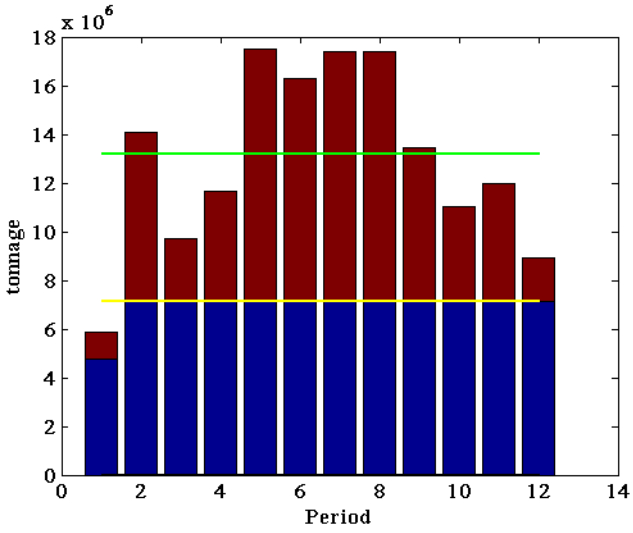

Fig. 12. Total mined and processed material, iteration 50.

Since only mining capacity constraints are relaxed in the studied case, these constraints are not satisfied, but the processing capacity constraints in all periods are satisfied and the solution has produced a uniform ore feed to the processing plant. The lower and upper bounds in each iteration, in addition to the overall lower and upper bound trends of the objective function value are illustrated in Fig. 13. Since the MILP is switched to a minimization problem by multiplying the objective function by -1, the objective function values are negative. The blue and green lines show the lower bound and upper bounds corresponding to each iteration, respectively. The red line represents the overall lower bound trend over all 50 iterations which is an increasing function.

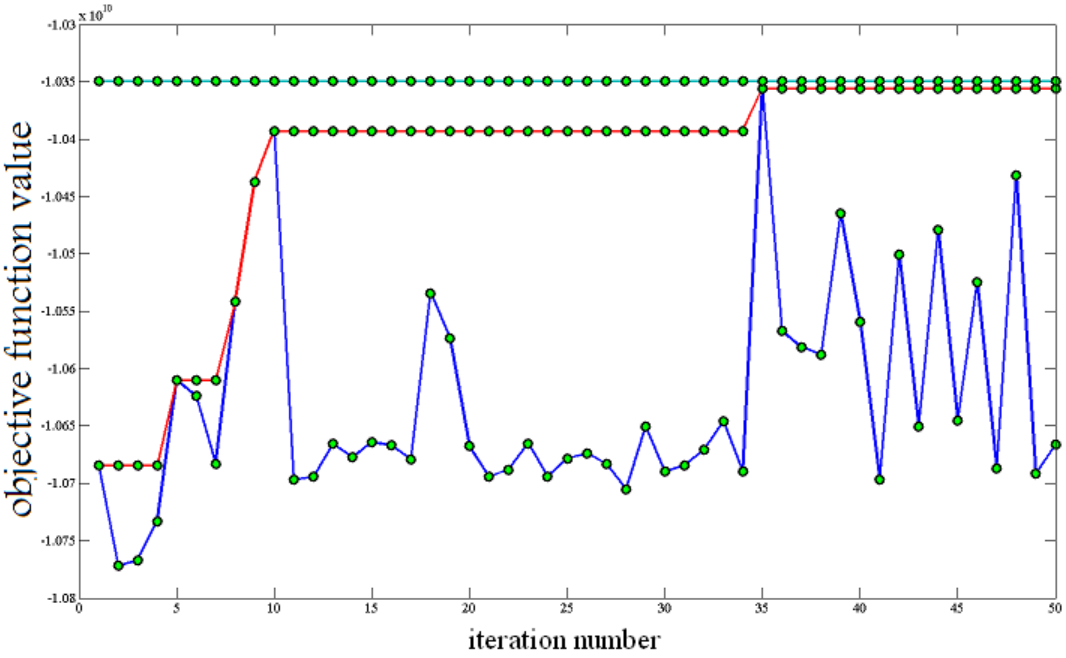

Fig. 13. Upper bound vs. lower bound.

After running the algorithm for 50 iterations, the gap between the lower and upper bounds is reached to 6.4×10^{-4}, showing a good convergence of the algorithm.
In this case, the upper bound of the problem (represented by the green line) has remained unchanged. It means either none of the dual solutions have been feasible for the primal problem, or none of those feasible solutions have been better than the first feasible solution to the primal problem and thus, could not update the upper bound.

6. Conclusions and future work

Lagrangian relaxation method is considered as a successful approach in dealing with huge and complicated LP and MILP problems. One of the well known algorithms which over the years have been used to find the Lagrangian multipliers is the sub-gradient algorithm. However in most cases, optimal solution to the relaxed (dual) problem is not feasible for the original (primal) problem. This infeasibility forces the algorithm developers to consider some additional steps (feasing) to make the solution feasible for the primal problem. In this research, we developed a heuristic feasing routine to overcome the infeasibility of solutions. Although in our studied cases, the resulting gap between the upper and lower bounds became relatively too small, but our proposed feasing routine did not overcome the infeasibility of the dual solution. Therefore, the upper bound has remained unchanged. As a future direction, it is recommended to focus on revised and more intelligent feasing heuristics.

7. References

[1] Akaike, A. and Dagdelen, K. (1999). *A strategic production scheduling method for an open pit mine*. Paper presented at 28rd Application of Computers and Operations Research in the Mineral Industries (APCOM) symposium, Littleton, Co. pp. 729-738.

[2] Askari-Nasab, H. and Awuah-offei, K. (2008). Mixed integer linear programming formulations for open-pit production scheduling. vol. 1. Edmonton, Alberta, Canada: Mining Optimization Laboratory (MOL), pp. 6-36.

[3] Askari-Nasab, H., Tabesh, M., and Badiozamani, M. M. (2010). *Creating mining cuts using hierarchical clustering and tabu search algorithms*. Paper presented at International Conference on Mining Innovation (MININ), Santiago, Chile. pp. 159-171.

[4] Dagdelen, K. and Johnson, T. (1986). *Optimum open pit mine production scheduling by Lagrangian parameterization*. Paper presented at 19th Application of Computers and Operations Research in the Mineral Industries (APCOM) symposium, Littleton, Co. pp. 127-141.

[5] Denby, B. and Schofield, D. (1994). Open pit design and scheduling by use of genetic algorithms. *Mining Technology: IMM Trans.*, Section A103, A21-A26.

[6] Fisher, M. (1985). An application oriented guide to Lagrangian relaxation. *Interfaces*, *15*,(2), 10-21.

[7] Fytas, K., Hadjigeorgiou, J., and Collins, J. L. (1993). Production scheduling optimization in open pit mines. *International journal of surface mining, reclemation and environment*, *7*,(1), 1-9.

[8] Gaupp, M. P. (2008). Methods for improving the tractability of the block sequencing problem for open pit mining.Thesis, Colorado School of Mines, Golden, Pages 150.

[9] ILOGInc. (2007). ILOG CPLEX. Ver. 11.0.

[10] Kawahata, K. (2006). A new algorithm to solve large scale mine production scheduling problems by using the Lagrangian relaxation method.Thesis, Colorado School of Mine, Golden.

[11] MathworkInc. (2009). MATLAB Software. Ver. 7.9 (R2009b).

[12] Newman, A. M., Rubio, E., Caro, R., Weintraub, A., and Eurek, K. (2010). A review of operations research in mine planning. *Interfaces*, *40*,(3), 222-245.

[13] Tan, S. and Romani, R. (1992). *Optimization models for scheduling ore and waste production in open pit mines*. Paper presented at 23rd Application of Computers and Operations Research in the Mineral Industries (APCOM) symposium, Littleton, CO. pp. 781-791.

8. Appendix

MATLAB and TOMLAB/CPLEX codes and documentation for the proposed algorithm.

The code runs the algorithm and produces the required graphs.