Les Cahiers du GERAD ISSN: 0711–2440

An Efficient Algorithm for the LP Relaxation of the Maximal Closure Problem with a Capacity Constraint

J. Byun
R. Dimitrakopoulos

G–2013–60

September 2013

An Efficient Algorithm for the LP Relaxation of the Maximal Closure Problem with a Capacity Constraint

Jongik Byun
Roussos Dimitrakopoulos*

COSMO – Stochastic Mine Planning Laboratory
Department of Mining and Materials Engineering
McGill University
Montreal (Quebec) Canada, H3A 2A7

jongik.byun@mcgill.ca
roussos.dimitrakopoulos@mcgill.ca

* *and GERAD*

September 2013

Les Cahiers du GERAD
G–2013–60

Abstract: The classical maximum closure problem is of particular importance to the mining industry because it is the underlying formulation related to mine design and production scheduling; this problem can be easily solved by a polynomial time max-flow/min-cut algorithm. However, if a single capacity constraint is added to the maximum closure formulation, the classical structure is destroyed and is classified in a category of NP-hard problems. Using classical integer programming techniques to solve these formulations with capacity constraints, it can take several days to solve linear programming (LP) relaxation of the real instances. This phenomenon has hindered the development of exact optimization approaches for open pit mine design and production scheduling of realistic sized problems.

In this paper, we develop an algorithm to solve the LP relaxation of the maximal closure problem with a single constraint, which is often referred to as the precedence constrained knapsack problem. The proposed algorithm expands on an existing parametric maximum flow model, and it is shown that the LP relaxation of the precedence constrained knapsack problem can be solved at the cost of only a constant factor of the worst-case time bound of the max flow algorithm. Computational results show that it is possible to solve the LP relaxation less than one minute for open pit mines with hundreds of thousands of mining blocks to be scheduled.

Key Words: Open pit mining problem, mining industry, precedence constrained knapsack problem, parametric max flow algorithm, LP relaxation.

1 Introduction

The goal of long-term open pit mine design consists primarily of three phases: ultimate pit design, which yields the largest extents that can be economically extracted; pushback design, which generates a general strategic sequence of extraction to get to the ultimate pit; and finally production scheduling, which details the annual sequence of extraction. All three phases have the common goal in that one wishes to maximize the economic value of the design while obeying slope (precedence) constraints. Ultimately, these three phases are interrelated; however, traditionally ultimate pit design and pushback design have been dealt with separately from long-term production scheduling because they are similar in formulation.

Lerchs and Grossmann (1965) propose the earliest algorithm (L-G algorithm) for the ultimate pit limit and pushback design heuristic for open pit mine design optimization, which is formulated as a discrete optimization problem referred to as the maximal closure problem. The authors note that it is possible to generate a sequence of extraction (pushbacks or nested pits) by parameterizing the value of the ore blocks and re-applying the ultimate pit algorithm. Using a simple discrete argument in the comparison of the quadratic formulations, Picard (1976) shows that the maximal closure problem can also be solved using the efficient max-flow/min-cut algorithm. More recently, Hochbaum (2008) and Hochbaum and Chen (2000) report new developments in the theory of max-flow/min-cut algorithms based on the ideas of Lerchs, Grossman and Picard. The L-G algorithm has been the industry standard for pit design optimization for the last three decades.

The shortcoming of max-flow algorithms when applied to open pit mine design is that the formulation and solution methods do not easily accommodate capacity constraints, which are used to define the maximum amount of material that can lie within a single nested pit. It has been shown from a theoretical perspective that the maximal closure problem with a single cardinality constraint is NP-hard (Hochbaum, 2000); given that the cardinality constraint is the simplest form of the capacity constraint, one can hardly expect that there is polynomial time algorithm to solve maximal closure problem with capacity constraints.

Furthermore, it is well known that even the linear programming (LP) relaxation of the maximal closure problem with a capacity constraint is in practice too time consuming (Moreno et al., 2010; Bienstock and Zuckerberg, 2009). In an effort to reduce the computational burden, many authors rely on aggregating blocks to reduce the size of the formulation (Ramazan, 2005; Boland et al., 2009), however these methods tend to reduce the resolution from which decisions can be made, thereby leading to sub-optimal solutions for the problem at hand.

Recently, Moreno et al. (2010) provide an algorithm to solve the LP relaxation of the open pit scheduling with multi-period capacity constraints. By applying heuristics to obtain an integer solution from the LP solution, the authors solve an open pit scheduling instance with 15 periods, 4 million blocks and 81 million precedence constraints. Bienstock and Zuckerberg (2009) present an algorithm that solves the LP relaxation of the maximal closure problem with multiple constraints for each period and demonstrate empirically that their formulation generates small integrality gaps. The authors analyze the relationship between optimal solution of the maximal closure problem produced through Lagranginan relaxation and the optimal solution of the LP relaxation.

A vast amount of theoretical research related to open pit mine design is in the area of cutting planes for the *precedence constrained knapsack problem*, which consists of a set of capacity constraints and multiple precedence constraints. This research is of interest to open pit mine design optimization because an efficient algorithm for solving the LP relaxation using cutting planes would lead to substantial reduction in computing time for the integer formulation of the problem. Given that the knapsack problem is the simplest form of the integer programming problems with only a single constraint, there has been a substantial amount of research into algorithms and cutting planes for the knapsack problem (Nemhauser and Wolsey, 1988; Byun et al., 2011). Additionally, researchers have been investigating aspects of polyhedral structures and cutting planes for the precedence constrained knapsack problem (Boyd, 1993; Park and Park, 1997; Ven de Leensel et al., 1999; Boland et al., 2011). Recently, several authors have proposed cutting plane algorithms in the context of open pit mine scheduling (Fricke, 2006; Bley et al., 2010; Boland et al., 2011). Meagher (2010) investigates the cutting planes for the max cut problem for directed graph, and applies the cutting planes to an instance of max cut problem with a knapsack constraint on the arcs, which is transformed from the precedence constrained knapsack problem.

This paper describes an efficient algorithm that provides exact solutions for the LP relaxation of the maximal closure problem with a single capacity constraint. While Moreno et al. (2010) have proposed an alternative algorithm for a related problem, we propose more efficient and flexible algorithm using *(complete) parametric max flow algorithm* (Gallo et al., 1989; Hochbaum, 2008). The proposed approach differs from the parametric max flow algorithm that is used in the nested pit heuristic, whereby we use the complete description of the max flow function, which is piecewise-linear function of the given parameter, to exactly solve the LP relaxation.

The proposed algorithm, although deals with open pit mine scheduling with a capacity constraint for each period, may be applied successively for each constraint in a set of constraints to approximate the LP relaxation of the maximal closure problem with multiple constraints. In the following section, the fundamentals of the maximal closure problem are revisited, including an alternative derivation of the maximal closure problem through duality theory. Then, the formulation is extended to incorporate a single capacity constraint. Subsequently, it is shown that the LP relaxation of the precedence constrained knapsack problem can be solved using the parametric max flow algorithm. The proposed algorithm is then tested on large, "real-world" data sets and computational results discussed. Conclusions and avenues for future research follow.

2 Prerequisites

In this section, we revisit the formulation of the maximal closure problem and show the relationship between maximal closure problem and max-flow problem in the context of primal-dual relationship.

2.1 Maximal closure problem

Consider a directed graph $G = (N, A)$, where N represents a set of blocks and A represents a set of precedence relationships among the blocks. A closure of N is defined as a subset of blocks $Y \subset N$ which does not violate any precedence relationship in A. For example, a precedence relationship can be described by an arc $ij \in A$, whereby block $i \in N$ must be mined after another block $j \in N$. A real number $p_j \in R$ is associated with each block called the profit of $j \in N$. It is assumed that the profit is permitted both to be positive or negative. The maximal closure problem consists of finding a closure $Y \subset N$ with the maximum sum of profits in Y. The maximal closure problem can be formulated as the following binary programming problem.

$$\begin{aligned} \max \sum_{j \in N} & p_j x_j \\ \text{s.t. } x_i - x_j &\leq 0 \qquad \forall_{ij} \in A \\ x_j &\in \{0, 1\} \qquad \forall_j \in N \end{aligned} \tag{1}$$

where variable $x_j \in \{0, 1\}$ represents the choice whether to include block $j \in N$ in the maximal closure or not. The only constraints in the maximal closure problem are precedence constraints. Note that this formulation does not contain a capacity constraint, and is used solely to determine the ultimate pit limit. The inclusion of a capacity constraint will be considered in a subsequent section.

2.2 Maximum flow problem

Picard (1976) shows the maximal closure problem can be transformed to a min-cut problem. Using the max-flow/min-cut theorem, the author demonstrates that a max-flow algorithm can be used to solve the maximal closure problem. Consider a directed graph graph $\overline{G} = (\overline{N}, \overline{A})$ where N represents set of nodes and A represents the set of arcs. For any arc $in \in A$, there can be the arc flow from $i \in N$ to $j \in N$. Each arc ij is associated with an arc capacity bound b_{ij}. Additionally, consider two specialized nodes in the graph: a sink node $s \in \overline{N}$ and a terminal node $t \in \overline{N}$. In the maximum $s - t$ flow problem, one tries to find the maximum flow from s to t while observing all the arc capacity constraints. It is easily shown that if an arc ts is added to the graph $\overline{G}$ with a unit flow cost of -1 and unlimited arc capacity, the maximum $s-t$ flow problem can be formulated as follows.

$$\begin{aligned} & \min -x_{ts} \\ \text{s.t.} \sum_{k\in N^+(j)} x_{jk} - \sum_{k\in N^-(j)} x_{kj} &= 0 \qquad \forall j \in N \\ 0 \leq x_j &\leq b_{ij} \qquad \forall ij \in A \end{aligned} \tag{2}$$

where variable x_{ij} is the flow in arc ij and $N^+(j)$ is the set of outgoing arcs from j. The maximum $s-t$ flow problem consists of flow balance constraints and arc capacity constraints. For more details, see Wolsey (1998).

2.3 Dual of the maximal closure problem

The precedence constraints in (1) build a node-edge incidence matrix and that the balance constraints in (2) build an edge-node incidence matrix. A node-edge incidence matrix can be built by transposing the edge-node incidence matrix and vice-versa. There is a similar relationship, connected by transposed matrices, in the duality theory in the linear programming theory. Here, we show the relationship between the maximal closure problem and min-cut problem, shown by Picard (1976), can also be derived by analyzing primal-dual relationship between (1) and (2).

To begin, the linear programming relaxation for the maximal closure problem (1) is presented by relaxing the integral constraints for x_j into linear bound constraints as follows:

$$\begin{aligned} & \max \sum_{j\in N} p_j x_j \\ \text{s.t. } & x_i - x_j \leq 0 \qquad \forall ij \in A \\ & 0 \leq x_j \leq 1 \qquad \forall j \in N \end{aligned} \tag{3}$$

Note that the objective function value for the linear relaxation (3) is greater than or equal to the objective value of the original (integer) problem. The dual of problem (3) can be written as follows:

$$\begin{aligned} & \min \sum_{j\in N} z_j \\ \text{s.t.} \sum_{k\in N^+(j)} y_{jk} - \sum_{k\in N^-(j)} y_{kj} + z_j &\geq c_j \qquad \forall j \in N \\ 0 \leq j_{ij}, 0 &\leq z_j \qquad \forall ij \in A, \quad \forall j \in N \end{aligned} \tag{4}$$

where variables y_{ij} are the dual variables for each precedence constraint in (3) and z_j is the dual variable for each bound constraint in (3). Using the dual formulation from problem (4), it is possible to apply three transformations to yield a problem similar to the maximal flow problem (2).

First by introducing slack variable s_j, the inequalities in formulation (4) transform into equalities constraints, and the problem is now modelled as follows:

$$\begin{aligned} & \min \sum_{j\in N} z_j \\ \text{s.t.} \sum_{k\in N^+(j)} y_{jk} - \sum_{k\in N^-(j)} y_{kj} + z_j - s_j &= c_j \qquad \forall j \in N \\ 0 \leq y_{ij}, 0 \leq z_j, 0 &\leq s_j \qquad \forall ij \in A, \quad \forall j \in N \end{aligned} \tag{5}$$

right-hand side coefficients c_j are then moved to the left-hand side. For this purpose, it is desirable to split c_j into positive and negative values, such as $c_j = c_j^+ - c_j^-$, where both c_j^+ and c_j^- are nonnegative.

$$\begin{aligned} & \min \sum_{j \in N} z_j \\ \text{s.t.} \quad & \sum_{k \in N^+(j)} y_{jk} - \sum_{k \in N^-(j)} y_{kj} - (c_j^+ - z_j) + (c_j^- - s_j) = 0 \qquad \forall j \in N \\ & 0 \le y_{ij}, 0 \le z_j, 0 \le s_j \qquad \forall ij \in A, \quad \forall j \in N \end{aligned} \tag{6}$$

Finally, by letting $\overline{z}_j = c_j^+ - z_j$ and $\overline{s}_j = c_j^- - s_j$, we obtain the final transformation which is equivalent to (4), the dual of the linear programming relaxation of maximal closure problem, as follows:

$$\begin{aligned} & \sum_{j \in N} c_j^+ - \max \sum_{j \in N} \overline{z}_j \\ \text{s.t.} \quad & \left(\sum_{k \in N^+(j)} y_{jk} + \overline{s}_j \right) - \left(\sum_{k \in N^-(j)} y_{kj} + \overline{z}_j \right) = 0 \qquad \forall j \in N \\ & 0 \le y_{ij}, \overline{z}_j \le c_j^+, \overline{s}_j \le c_j^- \qquad \forall ij \in A, \quad \forall j \in N \end{aligned} \tag{7}$$

Consider a modified graph $\overline{G} = (\overline{N}, \overline{A})$ by adding two nodes and one arc for each of the nodes in $G : \overline{N} = N \cup \{\text{s,t}\}$ and $\overline{A} = A \cup \{sj | j \text{ such that } c_j^+ > 0\} \cup \{jt | j \text{ such that } c_j^+ \le 0\}$.

Then it is clear that we can solve (7) by using maximum $s-t$ flow problem in $\overline{G}$ and the optimal solution of (7) has the following property:

Theorem 1 *The linear programming relaxation of the maximal closure problem is solved by the maximum $s-t$ flow problem using $\overline{G}$, and satisfies the following relationship (Picard, 1976):*

$$(\textit{LP relaxation of Max Closure}) = \sum_{j \in N} c_j^+ - \left(\textit{Max } s-t \textit{ flow in } \overline{G} \right).$$

Lemma 1 *The maximal closure problem is solved by the maximum $s-t$ flow problem for $\overline{G}$, and satisfies the following relationship (Picard, 1976):*

$$(\textit{Max Closure}) = \sum_{j \in N} c_j^+ - \left(\textit{Max } s-t \textit{ flow in } \overline{G} \right).$$

Theorem 1 and Lemma 1 state that the objective function value of the maximal closure problem is equal to the objective value of the LP relaxation, which is due to the total unimodularity of matrix in (3).

3 Precedence Constrained Knapsack Problem

In this section, we present the relationship between the precedence constrained knapsack problem and the complete parametric max flow algorithm (Gallo et al., 1989; Hochbaum, 2008). We show, by extending results in the previous section, that the LP relaxation of the precedence constrained knapsack problem can be solved by a parametric network flow algorithm.

3.1 LP relaxation and its dual

Consider the capacity constraint in the precedence constrained knapsack problem:

$$\sum_{j \in N} q_j x_j \le \kappa, \tag{8}$$

where κ is a maximum capacity and q_j is the contribution to the maximum capacity associated with node (block) $j \in N$. The linear relaxation of the precedence constrained knapsack problem can be derived by adding capacity constraints (8) into (3). The linear programming dual of the precedence constrained knapsack problem follows.

$$\begin{aligned} & \min \sum_{j \in N} z_j + \kappa\lambda \\ \text{s.t.} \quad & \sum_{k \in N^+(j)} y_{jk} - \sum_{k \in N^-(j)} y_{kj} + z_j \geq (c_j - q_j\lambda) \qquad \forall j \in N \\ & 0 \leq y_{ij}, 0 \leq z_j, 0 \leq \lambda \qquad \forall ij \in A, \quad \forall j \in N \end{aligned} \tag{9}$$

where variable λ is the dual variables for the capacity constraint (8). In a similar manner as the previous section, it is possible to transform (9) into a similar formulation as (7). By letting $\overline{z}_j = (c_j = q_j\lambda)^+ - z_j$ and $\overline{s}_j = (c_j - q_j\lambda)^- - s_j$, we get the transformation which is equivalent to (9):

$$\begin{aligned} & \min \left(\sum_{j \in N} (c_j = q_j\lambda)^+ - \sum_{j \in N} \overline{z}_j + \kappa\lambda \right) \\ \text{s.t.} \quad & \left(\sum_{k \in N^+(j)} y_{jk} + \overline{s}_j \right) - \left(\sum_{k \in N^-(j)} y_{kj} + \overline{z}_j \right) = 0 \qquad \forall j \in N \\ & \overline{z}_j \leq (c_j - q_j\lambda)^+ \qquad \forall j \in N \\ & \overline{s}_j \leq (c_j - q_j\lambda)^- \qquad \forall j \in N \\ & 0 \leq y_{ij}, 0 \leq \overline{z}_j, 0 \leq \overline{s}_j, 0 \leq \lambda \qquad \forall ij \in A, \quad \forall j \in N \end{aligned} \tag{10}$$

For a fixed λ, (10) has the same structure to (7) and can be solved it using the max-flow algorithm as in the previous section. However, note that λ is a positive real variable, which cannot be solved directly with a max-flow algorithm.

3.2 Parametric max flow algorithm

In this section, the basic properties of (10) are presented and show that (10) can be solved by using a parametric max flow algorithm. Gallo et al. (1989) propose the *parametric max flow algorithm* be used, where the arc capacities are not fixed and are a function of a single parameter. The author's proposed algorithms take advantage of the similarity of the successive max flow algorithms that should be solved, where the worst-case time bound is a constant factor of the traditional preflow-push max flow algorithm bound.

In Gallo et al. (1989), arc capacities are a function of a real-valued parameter, λ, and are denoted by $c_\lambda(ij)$ for $ij \in \overline{A}$. There are three assumptions:

1. $c_\lambda(sj) = a_j^s - b_j^s\lambda$, where $b_j^s \geq 0$
2. $c_\lambda(jt) = a_j^t + b_j^t\lambda$, where $b_j^t \geq 0$
3. $c_\lambda(ij)$, is constant, otherwise.

Note that, for the ease of the presentation, the assumptions have been modified from Gallo et al. (1989) by reversing the direction of the flow without loss of generality.

Let us define the max flow function m_λ to be the capacity of a max flow problem as a function of the parameter λ. It is known that m_λ is a piecewise-linear concave function with at most $|N|$ breakpoints. First, the parametric algorithm is used to solve the max flow algorithm for each member of an increasing sequence of parameter values $\lambda_1 < \lambda_2 < \cdots < \lambda_l$, which is referred to as the simple parametric max flow algorithm (Hochbaum, 2008). Then, Gallo et al. (1989) provide an algorithm that provides all breakpoints for m_λ. Given that the max flow function m_λ is a piecewise-linear concave function, it is possible to describe the whole function, m_λ, using all of the breakpoints of m_λ. This algorithm is called the complete parametric max flow algorithm (Hochbaum, 2008). All the algorithms run in $O\left(nm \log\left(\frac{n^2}{m}\right)\right)$ time.

Now, consider (10), which is the linear programming dual of the precedence constrained knapsack problem. It is clear that the constraints of (10) satisfies all three assumptions for the parametric max flow algorithm of Gallo et al. (1989). Among the three terms in the objective function of (10), $\sum_{j\in N}(c_j - q_j\lambda)^+$ and $\kappa\lambda$ are simple functions of λ, so it is unnecessary to minimize these terms. It is possible to solely consider the term $(\max \sum_{j\in N} \overline{z}_j)$, which can be interpreted as the objective function m_λ of the parametric max flow problem of Gallo et al., (1989).

As $\sum_{j\in N}(c_j - q_j\lambda)^+$, $\kappa\lambda$ and $(-\max \sum_{j\in N} \overline{z}_j)$ are piecewise-linear concave function of λ, the sum of the three terms in the objective value of (10) is also piecewise-linear concave function of λ. Finally, the following theorem is proposed:

Theorem 2 *The objective value of the linear programming relaxation of the precedence constrained max flow problem is*

$$\sum_{j\in N}(c_j - q_j\lambda^*)^+ - (\textit{Max } s-t \textit{ flow at } \lambda^*) + \kappa\lambda^*,$$

where λ^* is the optimal value of λ^* in (10).

4 Computational Tests

In this section, the algorithm to solve the LP relaxation of the precedence constrained knapsack problem is presented and computational results shown.

Algorithm

1. Get all the break points of the max flow function, m_λ, by only considering the objective term $(\max \sum_{j\in N} \overline{z}_j)$ in (10). In this stage, the parametric max flow algorithm gives all of the breakpoints of m_λ with complexity $O\left(nm \log\left(\frac{n^2}{m}\right)\right)$.
2. Get the optimal value λ^* in (10). In this stage, we scan three piecewise-concave functions $\sum_{j\in N}(c_j - q_j\lambda)^+$, $\kappa\lambda$ and $-m_\lambda$ with complexity $O(n)$.
3. Get the optimal solution of (10). In this stage, we use the max flow algorithm by fixing $\lambda = \lambda^*$ in (10) with complexity $O\left(nm \log\left(\frac{n^2}{m}\right)\right)$.

The code for the above was written in C++ and compiled under cygwin g++ compiler environment. The test was carried out on the notebook computer with a 2.13GHz Intel Core i3 330M CPU and 3GB RAM available. Table 1 summarizes the computational results. Note that Para Max Flow and Max Flow in Table 1 represent the run time of steps 1 and 3 in the algorithm shown above. The computational results show that parametric max flow algorithm provides the complete description of m_λ within a short time.

Table 1: The performance of proposed algorithm

Instance	Mine1	Mine2	Mine3
Blocks	4,275	40,762	219,434
Precedence	21,673	188,270	4,033,028
Para Max Flow	0.12s	5.2s	24.7s
Max Flow	0.02s	0.3s	12.5s

Moreno et al. (2010) deal with the same problem and in the absence of a direct comparison it is of interest to somehow consider the performance of both algorithms. Table 2 shows the computational results of Moreno et al. (2010). CMA presents the running time of Critical Multiplier Algorithm and CPX represents the running time of CPLEX linear programming solver. Reporting in Table 1 and Table 2 indicate that the proposed algorithm may be more comparable to the CMA algorithm in terms of order of time needed. Further comparisons will be required.

Table 2: The performance of CMA algorithm

Instance	American Mine	Marvin	Asia Mine	Andina
Blocks	19,320	53,668	772,800	4,320,480
Precedence	88,618	606,403	49,507,796	81,973,942
CMA	4s	12s	2m 36s	1h 44m
CPX	19m 26s	1h 3m	10d+	N/A

5 Conclusions

This paper presented an efficient algorithm to solve the LP relaxation of the precedence constrained knapsack problem. The algorithm provides the optimal solution of the LP relaxation using the parametric max flow algorithm. Additionally, it is possible to directly use the algorithm iteratively to approximate the LP relaxation with multiple knapsack constraints. Combined with efficient heuristics, which generate an integer solution from the fractional solution, we expect that we can solve the real instance of open pit mine scheduling in the reasonable time. We also expect that the algorithm in this paper makes it easier to use cutting planes for the precedence constrained knapsack problem.

The parametric max flow algorithm appears promising for exactly solving the LP relaxation of the problem with precedence constraints and multiple knapsack constraints. In is noted that in the proposed method there are no negative coefficients in the capacity constraints; another research topic could be to modify the algorithm to deal with the negative coefficients, which are commonly seen in blending constraints. Finally, it is desirable to extend the proposed algorithm to the stochastic integer programming formulation of the open pit mine scheduling (Meagher et al., 2009; Ramazan and Dimitrakopoulos, 2012; Asad and Dimitrakopoulos, 2012).

References

Asad, MWA, and Dimitrakopoulos, R (2012) Implementing a parametric maximum flow algorithm for optimum open pit mine design under uncertain supply and demand. Journal of the Operational Research Society, DOI: 10.1057/jors.2012.26.

Bienstock, D and Zuckerberg, M (2009) A new LP algorithm for precedence constrained production scheduling. Posted on Optimization Online.

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

Boland, N, Bley, A, Fricke, C, Froyland, G and Sotirov, R (2011) Clique-based facets for the precedence constrained knapsack problem. Mathematical Programming, DOI: 10.1007/s10107-010-0438-7.

Boland, N, Dumitrescu, I, Froyland, G and Gleixner, AM (2009) LP-based disaggregation approaches to solving the open pit mining productioin scheduling problem with block processing selectivity. Computers & Operations Research, 36, 1064–1089.

Boyd, EA (1993) Polyhedral results for the precedence-constrained knapsack problem. Discrete Applied Mathematics, 41, 185–201.

Byun, J, Park, K, Lee, K, Lee, C and Park, S (2011) Characterized separation of the rank 1 Chvátal-Gomory inequalities for integer knapsack problem. Draft paper.

Fricke, C (2006) Applications of integer programming in open pit mine planning, PhD thesis, Department of Mathematics and Statistics, The University of Melbourne, Melbourne, Australia.

Gallo, G, Grigoriadis, MD and Tarjan RE (1989) A fast parametric maximum flow algorithm and applications. SIAM Journal on Computing, 18, 30–55.

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

Hochbaum, D (2008) The pseudoflow algorithm: A new algorithm for the maximum flow algorithm. Operations Research, 58, 992–1009.

Lerchs, H and Grossman, IF (1965) Optimum design of open-pit mines. Transactions of CIM, 68, 17–24.

Meagher, C (2010) On the directed cut polyhedra and open pit mining. PhD thesis, School of Computer Science, McGill University, Montreal, Canada.

Meagher, C, Abdel Sabour, SA and Dimitrakopoulos, R (2009) Pushback design of open pit mines under geological and market uncertainties. Proceedings of the International Symposium on Orebody Modeling and Strategic Mine Planning: Old and new dimentions in changing world, 297–304.

Moreno, E, Espinoza, D and Goycoolea, M (2010) Large-scale multi-period precedence constrained knapsack problem: A mining application. Electronic Notes in Discrete Mathematics, 36, 407–414.

Nemhauser, GL and Wolsey, LA (1988) Integer and Combinatorial Optimization. Jones Wiley and Sons, New-York.

Park, K and Park, S (1997) Lifting cover inequalities for the precedence-constrained knapsack problem. Discrete Applied Mathematics, 72, 219–241.

Picard, JC (1976) Maximal closure of a graph and applications to combinatorial problems. Management Science, 22, 1268–1272.

Ramazan, S (2005) The new fundamental tree algorithm for production scheduling of open pit mines. European Journal of Operational Research, 177, 1153–1166.

Ramazan, S, Dimitrakopoulos, R (2012) Production scheduling with uncertain supply: A new solution to the open pit mining problem. Optimization and Engineering, DOI 10.1007/s11081-012-9186-2.

Van de Leensel, RLMJ, van Hoesel, CPM and van de Klundert, JJ (1999) Lifting valid inequalities for the precedence constrained knapsack problem. Mathematical Programming, 86, 161–185.

Wolsey, LA (1998) Integer Programming. Jones Wiley and Sons, New York.