IMPROVING THE PERFORMANCE OF A MIXED-INTEGER PRODUCTION SCHEDULING MODEL FOR LKAB'S IRON ORE MINE, KIRUNA, SWEDEN

by

Michael A. Martinez

ProQuest LLC.
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106 – 1346

A thesis submitted to the Faculty and the Board of Trustees of the Colorado School of Mines in partial fulfillment of the requirements for the degree of Doctor of Philosophy (Mineral Economics).

Golden, Colorado
Date 07 April 2006

Signed: Michael A. Martinez
Michael A. Martinez

Approved: Alexandra M. Newman
Dr. Alexandra M. Newman
Associate Professor
Thesis Advisor

Golden, Colorado
Date 4/7/06

Dr. Roderick Eggert
Professor and Head,
Division of Economics and Business

ABSTRACT

LKAB operates the Kiruna underground iron ore mine, which utilizes a mining method known as large-scale sublevel caving. To optimize production scheduling at Kiruna, we present a combined (short- and long-term) resolution model using mixed-integer programming. The model, which incorporates various operational requirements unique to sublevel caving, minimizes deviations from demand to produce a schedule containing monthly time periods. However, the resulting model is large and solution times for schedules of requisite length are excessive. To expedite solution time, we develop a decomposition-based heuristic consisting of two phases: (i) solving five subproblems, and (ii) solving a modified version of the original model based upon information gained from the subproblem solutions. We compare the performance of the heuristic to solving the original model directly on 15 datasets. On average, we find that our heuristic obtains better solutions faster than solving the original problem directly. We present various limitations to our approach and suggest possible extensions by modifying the heuristic when solving for schedules of greater length.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

First and foremost, I would like to thank my Lord and Savior Jesus Christ, who has given me the opportunity and ability to work on this project. I am also deeply indebted to my thesis advisor, Dr. Alexandra Newman. She patiently guided me through the research process, spent numerous hours reading my drafts, and constantly offered constructive criticism to make this work of the highest quality possible.

I would also like to extend thanks to the professors on my dissertation committee for their support and encouragement. I thank the Kiruna mine for the opportunity to work on this challenging problem and for access to its scheduling data. I acknowledge the U.S. Air Force Institute of Technology for sponsoring my degree over these three years. I am also grateful to the Modeling Simulation and Research Center at the U.S. Air Force Academy for allowing me access to its computing power for an aspect of this project. The views expressed in this article are those of the author and do not reflect the official policy or position of the United States Air Force, Department of Defense, or the U.S. Government.

For my precious wife Catalina and our two beautiful daughters.

Chapter 1

INTRODUCTION AND BACKGROUND

The Loussavaara-Kiirunavaara Aktiebolag (LKAB) company operates the Kiruna mine, located above the Arctic circle in northern Sweden. Currently the second largest underground mine in the world, Kiruna has been operational for more than a century. The mine employs about 600 workers and produces approximately 24 million tons of iron ore per year in the form of three raw iron ore products: $B1$, $B2$, and $D3$. These raw products are used to supply planned production quantities at four ore post-processing plants, or mills. The ore products, classified according to their phosphorous content, are processed into fines and pellets, both of which are used as raw materials in the manufacture of steel end-products.

The Kiruna ore body is a high-grade magnetite deposit which is, on average, 4 kilometers long and 80 meters (m) wide. Phosphorus (P) is the primary contaminant and its presence directly determines the ore type. The $B1$ and $B2$ ore types, which are low-phosphorus, high-iron (Fe) magnetites, contain approximately 0.06% and 0.20% phosphorus, respectively, and together constitute about 80% of the deposit. The mills use $B1$ to make high-quality fines, while $B2$ is transformed into medium-grade pellets. The remaining 20% of the ore body is *D3* ore, which is a high-phosphorus (average 2%), apatite-rich magnetite processed by the mills into low-quality pellets. Potassium (K_2O) is another contaminant with typical levels less than 0.15%.

Production scheduling at the Kiruna mine intends to satisfy, with minimum deviation, the target production quantities set by the mills; both over- and under-

production are undesirable. The production schedule, typically spanning multiple years with monthly resolution, should provide the ore extraction sequence and timing over the planning horizon. In the past, manual schedulers, even with computer spreadsheet tools, were extremely limited in considering possible schedules, at times expending considerable effort to simply achieve a *feasible* production schedule. More recently, Kiruna has turned to mathematical programming, and a long-term (strategic planning) optimization model is currently being used to plan production.

Kiruna began mining operations in 1898, extracting iron ore exclusively via surface methods. By about 1952, the pit deepened to such an extent that it became more cost-effective to mine underground. The underground mining method Kiruna currently employs is known as large-scale *sublevel caving*. This method is well-suited for extracting ore from vertically positioned, fairly pure, large, vein-like deposits. The mine is divided into ten main production areas, about 400 to 500 m in length. Each production area has its own group of ore passes; such a group is also known as a *shaft group*. A shaft group is located at the center of each production area, and extends down to the main level. Each production area consists of about 10 sublevels, and entry to these sublevels is gained via access ramps. Currently, the mine is producing down to the 1,045 m sublevel.

One or two 25-ton-capacity electric Load Haul Dump units (LHDs) operate on a sublevel within each production area and transport the ore from the crosscuts (from which the ore is extracted) to the ore passes, where loaded trains haul the ore to a crusher. At the crusher, the ore is broken into pieces small enough to be hoisted to the surface via vertical shafts. Up to 25 LHDs can operate daily throughout the mine; however, the allowable number of LHDs within each shaft group is restricted to about two or three to prevent LHD operators from driving over and damaging other

LHD cables.

Figure 1.1 illustrates the relationship between production areas, ore passes, sublevels and ore haulage routes. See Topal (2003) for a more detailed description of the Kiruna mine and its characteristics.

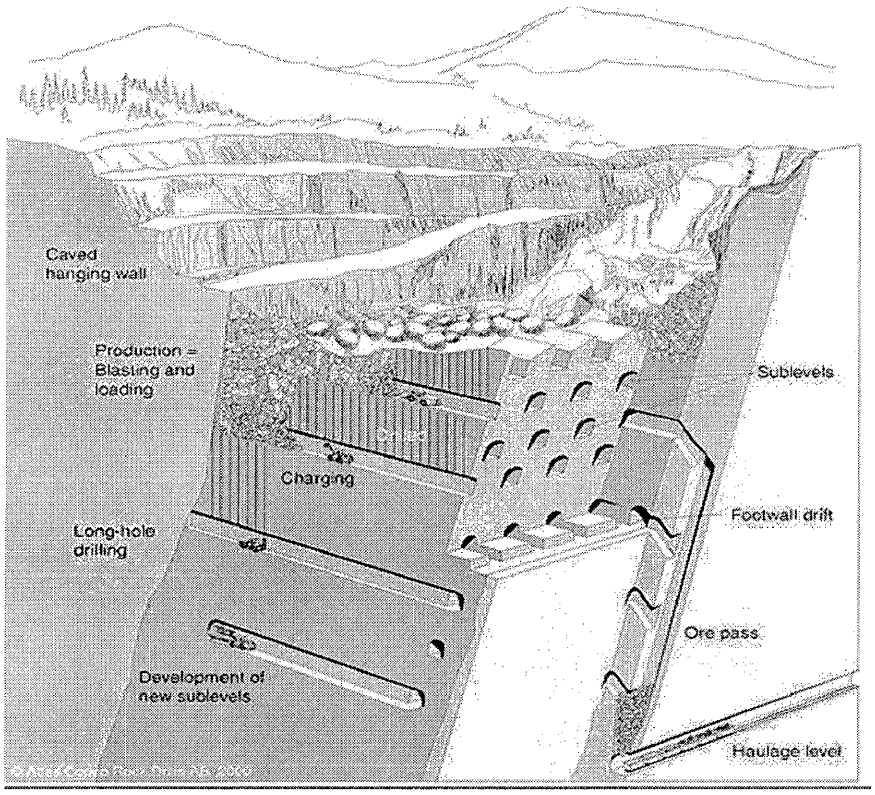

Figure 1.1. An example of sublevel caving (Altas Copco, 2005).

The site on which each LHD operates is also referred to as a *machine placement.* The number of machine placements that can be started in a given time period is restricted due to the availability of the crew that prepares the machine placement to start to be mined. The number of active machine placements, i.e., machine placements currently being mined, is also restricted due to LHD availability. Each machine placement belongs to a unique shaft group. A machine placement averages 200 to

500 m in length and contains from one to three million tons of ore and waste rock. A machine placement possesses the same height as the mining sublevel and extends from the hangingwall to the footwall. Between five and fifteen smaller (100 m) entities known as *production blocks* constitute a machine placement. About one month is required to mine each production block. If a machine placement is left partially mined, old explosives (which only have a life of about 30 days) must be replaced to blast the solidified cave rock. This requirement, coupled with the aggravation of tracking partially-mined machine placements, results in operational restrictions that require continuous production within a machine placement until all available ore has been removed.

Whether a machine placement can (or must) be mined depends on the relative position of machine placements that have started to be mined. Specifically, certain machine placements beneath a given machine placement cannot start to be mined until some portion of the given machine placement has been mined, and machine placements to the right and left of a given machine placement must start to be mined after a specified portion (typically, 50%) of the given machine placement has been mined (to prevent blast damage on adjacent machine placements). These operational constraints are referred to as vertical and horizontal *sequencing constraints*, respectively.

Each machine placement possesses a series of notional *drawdown lines* consisting of several production blocks each. Within a machine placement, the order in which production blocks must be mined is regulated by this series of drawdown lines, which also helps to enforce continuous mining of a machine placement. These drawdown lines cut horizontally or at a 45 degree angle though several blocks within the machine placement and preclude production blocks in a drawdown line underneath a

given drawdown line from being extracted until all ore in the given drawdown line is extracted. This mining pattern is necessary to correctly execute the sublevel caving method so that the mined out areas do not collapse on top of ore that is yet to be retrieved. We illustrate the sequencing relationships, along with drawdown lines, in Figure 1.2. Minimum and maximum production levels per month govern the rate at which the blocks within a machine placement are mined. These rates ensure continuous mining of machine placements, as discussed above, as well as adherence to production capacity restrictions. Because of vertical and horizontal sequencing constraints and the relative positions of machine placements and the production blocks within them, there are only certain time periods in which these can be mined.

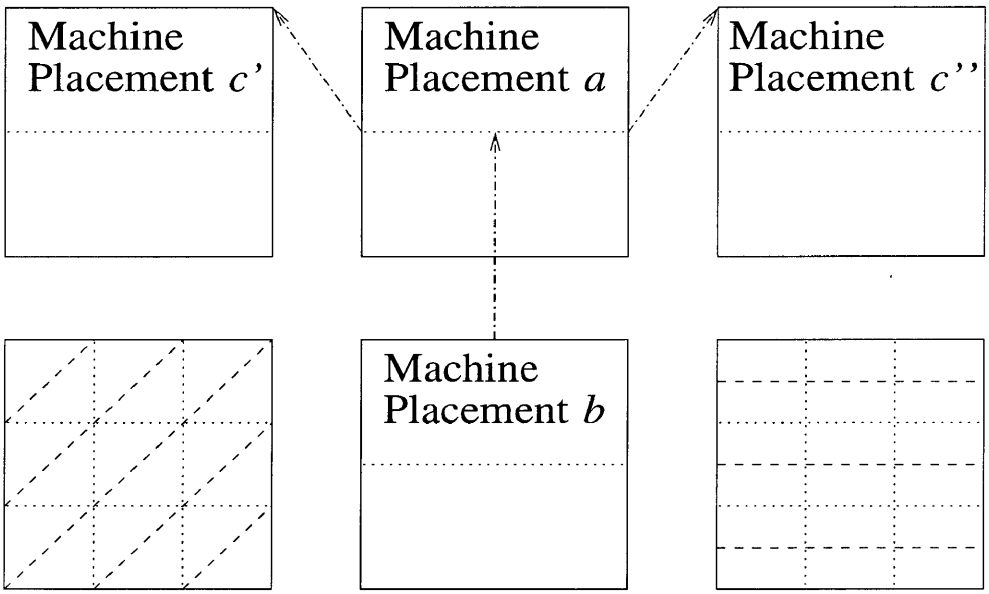

Figure 1.2. A depiction of six machine placements. The vertical sequencing requires that at least 50% of machine placement a be mined before beginning to mine machine placement b. The horizontal sequencing requires that machine placements c' and c'' begin to be mined once 50% of machine placement a is mined. Dotted lines divide the machine placements in the lower corners into 9 production blocks each, with dashed drawdown lines at either a 45-degree angle or parallel to the surface.

Chapter 2

LITERATURE REVIEW

We divide the relevant production scheduling literature by mining method: either surface or underground mining. We emphasize the underground mining papers, and cite specific examples of previous models the Kiruna mine has employed for its production scheduling.

2.1 Surface Mining

Early applications of mathematical programming in the mining industry primarily focus on open pit (surface) mining, a fundamentally different operation from underground mining. In an open pit mine, a single pit is dug downward from the surface; ore grade restrictions must be observed so that waste material is not allowed to dilute the ore quality. Typical optimization problems focus on determining the economic "break-even" depth for the pit, called the *ultimate pit limits*, below which operations should either terminate or switch to underground mining methods. The Kiruna mine itself was a surface mine until 1952 when underground mining commenced.

In a seminal work, Lerchs & Grossman (1965) develop a graph-theoretic algorithm for optimally calculating the ultimate pit limits. To improve the efficiency of the solution method, others (e.g., Underwood & Tolwinski, 1998; Hochbaum & Chen, 2000) utilize a dual simplex algorithm and maximum flow algorithms, respectively. Winkler & Griffin (1998) use a linear programming (LP) formulation with

multiple objectives to maximize profit while meeting ore blending and mine workload goals. Wilke & Reimer (1977) also formulate a linear program to produce a short-term schedule for an open pit iron ore mine. Their model maximizes mining ore blocks assigned highest priority with respect to ore quality. Though the constraints appropriately model various features of surface mining, such as ore blending and equipment capacity requirements, achieving practical schedules requires iterative manual interaction.

In order to model discrete decisions for mine production scheduling, variables constrained to integer values are necessary. Mixed-integer programming (MIP) formulations prove much less tractable than comparable linear programming models. *Tractability* is the ease with which an optimization model is solved and analyzed. Barbaro & Ramani (1986), along with Smith (1998), provide examples in which MIP is applied to open pit mines. Barbaro and Ramani maximize total profit and include facility location decisions, which necessitate integer variables. The small problem instance that they solve contains only 10 binary variables, so tractability is not an issue. Smith models production scheduling decisions with variables representing whether to mine a block or not. Because each production decision requires its own binary variable, the resulting problem is very large, even for a short mining schedule. He restricts the model to a single time period in a silver and gold surface mining operation. The objective function minimizes deviations from production goals with respect to silver, gold, and waste rock.

2.2 Underground Mining

Within the underground mining literature, we find applications of various mining methods. We first explore models of other mining methods, then turn specifically to

Kiruna models, which are based on its sublevel caving method.

2.2.1 General Underground Mining Models

Scheduling of underground mining operations is primarily characterized by discrete decisions to mine blocks of ore, along with complex sequencing relationships between blocks. Due to these complex relationships, formulation as a strict network flow problem is typically not appropriate. Since LP models cannot capture the discrete decisions required for scheduling, MIPs are generally the more appropriate mathematical programming model. Integer decision variables represent whether to mine a block of ore in a particular time period, which is the essence of production scheduling in an underground context.

Because MIPs are computationally more complex to solve than LPs, solution techniques involving relaxing integer variables to assume continuous values have been devised. For example, Williams *et al.* (1972) plan sublevel stoping operations for an underground copper mine over one year. They use a linear programming approximation model to determine the amount of ore to be mined per month from each stope. The objective function minimizes deviations between successive months, seeking to produce a well-balanced schedule. Because their goal is merely to generate higher quality schedules than the then-current manual trial-and-error method, continuous variables suffice. More recently, Jawed (1993) formulates a linear goal program for production planning in an underground room-and-pillar coal mine. The decision variables determine the amount of ore to be extracted and the objective function minimizes production deviations from target levels, though only for a single time period. Note that these two models compromise schedule quality and the length of the time horizon, respectively. Tang *et al.* (1993) integrate linear programming with sim-

ulation to address scheduling decisions, as does Winkler (1998). The linear program handles the continuous variables, which determine the amount of ore to extract; the simulation model evaluates discrete scheduling decisions. In these two examples, the applications consist of only a single time period and/or cannot guarantee optimal solutions because the technique iteratively fixes variable values and optimizes only a portion of the scheduling problem.

Trout (1995) was perhaps the first to try integer programming for optimizing underground mine production schedules. By maximizing net present value, his model schedules an underground stoping mine for base metals (e.g., copper sulphide). The constraint set incorporates block sequencing, equipment capacity, and backfill indicators. However, his attempt does not achieve optimality since, after 200 hours, the algorithm terminates early when the computer reaches memory capacity. Winkler (1996) models production scheduling in an underground coal mine to minimize fixed and variable extraction costs, but limits his model to a single time period.

Several years later, researchers formulate more tractable MIP models. Carlyle & Eaves (2001) present a model that maximizes revenue from Stillwater's platinum and palladium mine which uses the sublevel stoping mining method. The problem focuses on strategic mine expansion planning, so the integer decision variables schedule the timing of various mining activities: development and drilling, and stope preparation. Although the formulation relaxes integrality constraints for variables tracking the amount of ore extracted, the model returns near-optimal solutions for a 10-quarter time horizon. Smith *et al.* (2003) incorporate a variety of features into their lead and zinc underground mine model, including sequencing relationships, capacities, and minimum production requirements. However, they significantly reduce the resolution of the model by aggregating stopes into larger blocks. The resulting model, with

time periods of one-year length, maximizes net present value over the life of the mine (here, 13 years). The model generates near-optimal results in less than an hour. The authors note that further research should refine the level of detail to account for ore grade fluctuations and block sequencing considerations. Sarin & West-Hansen (2005) schedule a coal mining operation to maximize net present value less penalties for irregular schedules. They expedite the solution time for their model with a Benders' decomposition-based methodology. We note that none of the previous examples applies optimization to the sublevel caving mining method used at Kiruna.

2.2.2 Kiruna-Specific Models

Over the last decade, successive efforts at production scheduling for the Kiruna mine sought a schedule of requisite length in a reasonable amount of solution time. Using the machine placement as the basic mining unit, initial attempts significantly shorten the time horizon, sacrificing schedule quality. Almgren (1994) considers a one-month time frame; hence, in order to generate a five-year schedule, he runs the model 60 times. In a similar vein, Topal (1998) and Dagdelen *et al.* (2002) iteratively solve one-year subproblems (with monthly resolution) in order to achieve production plans for five-year and seven-year time horizons, respectively. These three models provide suboptimal solutions because they myopically disregard the effects of subsequent time periods outside the horizon length. Additionally, Kuchta (2002) develops a computer-assisted manual heuristic scheduling program. However, he admits that it is common to abandon partial schedules and restart the procedure due to the difficulty of satisfying target demands and the inability to assess the future impact of current scheduling decisions.

Kuchta *et al.* (2004) consider a five-year time horizon and three ore types, and significantly improve model tractability by redefining the decision variables and by assigning earliest and latest possible start dates for mining machine placements. These early and late start pre-processing assignments substantially reduce the number of binary variables in the model, resulting in a near-optimal schedule in a matter of minutes. In an implementation consisting of two ore types, Newman & Kuchta (in press) aggregate binary variables, effectively collapsing the time periods into multi-period "windows;" the solution to this aggregation problem allows elimination of all but a "reasonably good" set of starting times for each machine placement, restricting the model to a subset of start date choices beyond the restrictions determined with the early and late start algorithms. The restricted dates are then applied to the original problem, resulting in optimal three-year production schedules with monthly resolution in less than an hour.

Given the most recent advances with efficient MIP formulations and faster solution times, model progression calls for finer resolution in terms of mineable block size. For the Kiruna mine, this means modeling at the *production block* level. Additionally, extra features must be incorporated to allow more realistic schedules that accurately mimic actual progression of mining operations (e.g., *drawdown lines*). Newman *et al.* (to appear) present such a model with combined resolution incorporating three ore types. Short-term decisions concern mining of production blocks and drawdown lines, while in the long-term, decisions model mining of entire machine placements only.

Chapter 3

MODEL

The objective of production scheduling at Kiruna is to minimize deviations from target quantities (demand) for each ore type. Due to company policy, LKAB does not stockpile iron ore. Specifically, there is physically no space in which to store more than about 50 ktons of extracted iron ore. Because of this guideline, we do not use inventory constraints. Furthermore, because LKAB's goal is to meet demand as closely as possible in each time period so as to regulate the amount of ore processed at the mills, a shortage in one time period cannot be compensated by a surplus in, say, the following time period.

3.1 Combined Resolution Model Features

Currently, Kiruna uses a long-term production scheduling model (Kuchta *et al.*, 2004) to strategically plan its ore extraction sequence. This long-term model is a mixed-integer program that minimizes deviations from planned production targets using hundreds of binary variables representing whether or not to mine a specific machine placement in each month of the planning horizon. A restriction forces continuous mining of a machine placement once the machine placement has started to be mined, which, in turn, forces production blocks to be mined in a fixed sequence and at a fixed time. The model also considers the sequencing restrictions (both vertical and horizontal) between machine placements and physical limitations of the mine, particularly with regard to shaft group and equipment capacities. Specifically, the

currently existing features in the Kiruna mathematical programming model are:

- *accounting constraints* that monitor deviations from target production levels in terms of ore type per time period;
- *vertical sequencing constraints* that prevent mining a given machine placement until 50% of a vertically constraining machine placement is mined;
- *horizontal sequencing constraints* that force mining a given machine placement when 50% of an adjacent machine placement is mined;
- *shaft group constraints* that restrict the number of operational LHDs within a shaft group during a single time period;
- *operational constraints* that limit the number of machine placements that can start to be mined during a single time period.

We modify the long-term model to comprise several levels of detail. At the coarser (original) level of detail, binary decision variables indicate which machine placements to start mining each month. For machine placements that are currently active, we model decisions at a finer level of detail, with continuous variables representing how much to mine from each production block in each month. Correspondingly, another set of binary variables tracks which drawdown lines have finished being mined, or equivalently, which production blocks constituting a drawdown line have all finished being mined. This extra level of detail allows the model to more closely control the amount and types of ore that are extracted from each machine placement in the short-term planning horizon. Accordingly, when integrating the short-term detail, constraints must be added that maintain the sequencing requirements and operational limitations of the mine. Note also that constraints are required to relate the short- and

long-term resolutions. Specifically, we must consider the sequencing between machine placements modeled as production blocks and drawdown lines (short-term resolution), and those machine placements modeled as single entities (long-term resolution). We categorize these additional constraints that incorporate the short-term resolution as follows:

- *production block constraints* that limit the amount extracted from a production block at the amount of reserves available within that production block;
- *drawdown line completion constraints* that indicate when a drawdown line has been completely mined out;
- *drawdown line sequencing constraints* that prevent underlying drawdown lines from being mined until the overlying drawdown line has been completely extracted;
- *vertical sequencing constraints* both between production blocks and drawdown lines (in machine placements modeled with short-term resolution), and also between drawdown lines and machine placements modeled with long-term resolution;
- *horizontal sequencing constraints* that force mining a machine placement (long-term resolution) when, for an adjacent machine placement (short-term resolution), the drawdown line indicating at least 50% of total ore in the machine placement has been extracted;
- *production-rate constraints* that regulate both minimum and maximum monthly production rates;

- *short-term demand* constraints that enforce mining exactly the amount demanded for each of the first six time periods, regardless of ore type.

3.2 Mathematical Formulation

A complete mathematical formulation of the combined short- and long-term model follows:

SETS:

- K = set of ore types
- V = set of shaft groups
- A = set of machine placements
- A_v = set of machine placements in shaft group v
- IA = set of inactive machine placements
- $A_a^{\mathcal{V}}$ = set of machine placements whose start date is restricted vertically by machine placement a
- $A_a^{\mathcal{H}}$ = set of machine placements whose start date is forced by adjacency to machine placement a
- A_t = set of machine placements that can start to be mined in time period t
- B = set of production blocks
- B_a = set of production blocks in machine placement a
- B_l = set of production blocks in drawdown line l

- B_t = set of production blocks that can be mined in time period t
- L = set of drawdown lines
- L_a = last (i.e., most deeply positioned) drawdown line in machine placement a
- LC = set of drawdown lines constrained by another drawdown line
- L_l = set of drawdown lines that vertically constrain drawdown line l
- $L_a^{\mathcal{V}}$ = drawdown line whose finish date vertically restricts starting to mine machine placement a
- $L_a^{\mathcal{H}}$ = drawdown line whose finish date forces the start date of machine placement a by adjacency
- L_t = set of drawdown lines that can be mined in time period t
- $\mathcal{T}$ = set of time periods composing the long-term time horizon
- T = set of time periods composing the short-term time horizon ($\subset \mathcal{T}$)
- T_a = set of time periods in which machine placement a can start to be mined (restricted by machine placement location and the start dates of other relevant machine placements)
- T_b = set of time periods in which production block b can be mined (restricted by production block location and the start dates of other relevant production blocks)
- T_l = set of time periods in which drawdown line l can finish being mined (restricted by drawdown line location and the finish times of other relevant drawdown lines)

- $\hat{T}_l$ = time period by which all production blocks in drawdown line l must finish being mined

PARAMETERS:

- p_t = penalty associated with deviations in time period t $(= |\mathcal{T}| + 1 - t)$
- LHD_t = maximum number of machine placements that can start to be mined in time period t
- LHD_v = maximum number of active machine placements in shaft group v
- d_{kt} = target demand for ore type k in time period t (ktons)
- $r_{at'tk}$ = reserves of ore type k available at time t in machine placement a given that the machine placement started to be mined at time t' (ktons)
- $\rho_{at't} = \begin{cases} 1 & \text{if machine placement } a \text{ is being mined at time } t \text{ given that} \\ & \text{it started to be mined at time } t' \\ 0 & \text{otherwise} \end{cases}$
- R_{bk} = reserves of ore type k contained in production block b (ktons)
- $\underline{C}_{at}$ = minimum production rate of machine placement a in time period t (ktons per time period)
- $\overline{C}_{at}$ = maximum production rate of machine placement a in time period t (ktons per time period)

DECISION VARIABLES:

- $\overline{z}_{kt}$ = deviation above the target demand for ore type k in time period t (ktons)

- $\underline{z}_{kt}$ = deviation below the target demand for ore type k in time period t (ktons)
- x_{bt} = amount of ore mined from production block b in time period t (ktons)
- $w_{lt} = \begin{cases} 1 & \text{if we finish mining all blocks contained in drawdown line } l \\ & \text{by time period } t \\ 0 & \text{otherwise} \end{cases}$
- $y_{at} = \begin{cases} 1 & \text{if we start mining machine placement } a \text{ at time period } t \\ 0 & \text{otherwise} \end{cases}$

Formulation:

(P):

$$\min \quad \sum_{k,t} (p_t)(\underline{z}_{kt} + \overline{z}_{kt})$$

subject to:

$$\sum_{a \in A_t} \sum_{t' \in T_a, \leq t} r_{at'tk} y_{at'} + \sum_{b \in B_t} \frac{R_{bk}}{\sum_{\hat{k} \in K} R_{b\hat{k}}} x_{bt} + \underline{z}_{kt} - \overline{z}_{kt} = d_{kt} \quad \forall\, k \in K, t \in \mathcal{T} \quad (3.1)$$

$$\sum_{a \in A_t} \sum_{k \in K} \sum_{t' \in T_a, \leq t} r_{at'tk} y_{at'} + \sum_{b \in B_t} x_{bt} = \sum_{k \in K} d_{kt} \quad \forall\, t \in \mathsf{T} \quad (3.2)$$

$$\sum_{a \in A_v \cap A_{t'}} \sum_{t' \in T_a, \leq t} \rho_{at't} y_{at'} + \sum_{a \in A_v} \sum_{l \in L_a \cap L_t} (1 - w_{l\hat{t}}) \leq LHD_v \quad \forall\, v \in V, t \in T_l, \hat{t} \in \hat{T}_l \quad (3.3)$$

$$\sum_{a \in IA \cap A_t} y_{at} \leq LHD_t \quad \forall\, t \in \mathcal{T} \quad (3.4)$$

$$\sum_{t \in T_b} x_{bt} \leq \sum_{k \in K} R_{bk} \quad \forall\, b \in B \quad (3.5)$$

$$\sum_{b \in B_l} \sum_{u \leq t} x_{bu} \geq \sum_{b \in B_l} \sum_{k \in K} R_{bk} w_{lt} \quad \forall\, l \in L, t \in T_l \quad (3.6)$$

$$\sum_{u \leq t} x_{bu} \leq \sum_{k \in K} R_{bk} w_{\hat{l}t} \quad \forall\, l \in LC, b \in B_l, \hat{l} \in L_l, t \in T_{\hat{l}} \quad (3.7)$$

$$\sum_{b \in B_a \cap B_t} x_{bt} \leq \overline{C}_{at} \quad \forall\, a \in A, l \in L_a, t \in T_l \tag{3.8}$$

$$\sum_{b \in B_a \cap B_t} x_{bt} \geq \underline{C}_{at}(1 - w_{lt}) \quad \forall\, a \in A, l \in L_a, t \in T_l \tag{3.9}$$

$$w_{lt} \geq y_{\tilde{a}t} \quad \forall\, a \in A, \tilde{a} \in A_a^{\mathcal{V}}, l \in L_a^{\mathcal{V}}, t \in T_{\tilde{a}} \tag{3.10}$$

$$\sum_{t \in T_{\tilde{a}}, \leq \hat{t}} y_{\tilde{a}t} \geq w_{l\hat{t}} \quad \forall\, a \in A, \tilde{a} \in A_a^{\mathcal{H}}, l \in L_a^{\mathcal{H}}, \hat{t} \in T_l \tag{3.11}$$

$$\sum_{t \in T_a} y_{at} \geq y_{a't'} \quad \forall\, a \in A, a' \in A_a^{\mathcal{V}}, t' \in T_{a'}, a' \neq a \tag{3.12}$$

$$\sum_{t' \in T_{a'}} y_{a't'} \geq y_{at} \quad \forall\, a \in A, a' \in A_a^{\mathcal{H}}, t \in T_a, a' \neq a \tag{3.13}$$

$$\overline{z}_{kt}, \underline{z}_{kt} \geq 0 \ \forall\, k, t; \quad x_{bt} \geq 0 \ \forall\, b, t; \quad w_{lt} \text{ binary } \forall\, l, t; \quad y_{at} \text{ binary } \forall\, a, t \tag{3.14}$$

The objective function measures the total weighted tons of deviation, placing more emphasis on exactly meeting demand in the short-term time horizon. Not only does the weighting scheme place a greater penalty on more important shortfalls, but it also breaks symmetry which helps to guide the search algorithm. Constraints (3.1) record for each ore type and time period the amount in excess or short of the target demand of ore production. Constraints (3.2) require that for each time period in the short term planning horizon (typically, six months), the exact total target amount of ore required, regardless of ore type, is mined. This requirement prevents the post-processing mills from sitting idle. Constraints (3.3) limit the maximum number of active machine placements in each shaft group and time period. Constraints (3.4) restrict the number of long-term machine placements that can be started in a time period; short-term machine placements are assumed to be currently active. Constraints (3.5) preclude mining more than the available reserves within a production block. Constraints (3.6) relate finishing mining a drawdown line to mining the production

blocks within that drawdown line. Constraints (3.7) preclude a production block in a drawdown line from starting to be mined until all blocks in constraining drawdown lines have been mined. Constraints (3.8) and (3.9) enforce monthly maximum and minimum production rates, respectively. Constraints (3.10) and (3.11) enforce vertical and horizontal sequencing, respectively, between machine placements modeled with short-term and long-term resolution. Note that the drawdown line in a machine placement modeled with short-term resolution both (i) controls access (vertically) to a constrained machine placement modeled with long-term resolution, and (ii) forces mining (horizontally) a machine placement modeled with long-term resolution. Constraints (3.12) and (3.13) enforce vertical and horizontal sequencing, respectively, between machine placements modeled with long-term resolution. Finally, constraints (3.14) enforce non-negativity and integrality, as appropriate. Typical model scenarios (65 machine placements, 102 production blocks, 3 ore types) that we consider span three years and contain more than 1000 binary variables and over 2500 constraints.

This Kiruna model differs from typical underground mining formulations (e.g., Trout, 1995; Winkler, 1996; Carlyle & Eaves, 2001) in two respects: (i) it does not account for the difference in costs from mining various machine placements due, for example, to their location in the mine, and (ii) the objective does not consider the net present value of ore. With respect to the first issue, we assume that all ore will be mined eventually, and hence, total mining costs are sunk. Therefore, we need not consider discrepancies in costs between mining various machine placements. The second aspect is explained by the difference between the markets for iron ore and precious metals. Precious metals such as gold and silver are traded on, e.g., the Commodity Exchange of New York. These metals are bought and sold worldwide, and the strategy of mines extracting these metals is to maximize profits by producing

as much as is economically viable given current market prices. By contrast, markets associated with base metals such as iron ore are regionalized, as transportation costs are high relative to the value of the commodity. Within these markets, steel companies enter into a contract with an iron ore producer, settling on a price commensurate with the chemical and physical characteristics of the iron ore. Large buyers tend to influence prices in contracts between other buyers and iron ore producers. The negotiated prices generally hold for about a year, and iron ore producers are obligated to supply a certain amount of iron ore to each buyer with whom they hold a contract. Therefore, iron ore mines like Kiruna are concerned with meeting contractual demands as closely as possible.

Additionally, except for the most recent formulation (Newman *et al.*, to appear), this model also differs from its predecessors (i.e., Topal, 1998; Dagdelen *et al.*, 2002; Kuchta *et al.*, 2004) in that it consists of both short- and long-term resolution, increasing problem size and creating a much more difficult problem. In a restatement of this formulation, Newman *et al.* (to appear) use a two-year time horizon and highlight that the combined resolution model reduces total absolute deviations by approximately 70% over those obtained from a model with only long-term resolution. We note, however, that this improved solution comes with a significant decrease in tractability.

Chapter 4

SOLUTION METHODOLOGIES

In general, MIP solvers apply the *branch-and-bound* algorithm, which is based on solving an enumeration tree of LP relaxation problems with some subset of the variables fixed to integer values. The relaxation problem at each node allows those integer variables that are not fixed to assume continuous values. At the root node of the tree, no variables' values are fixed, and an LP relaxation of the original problem is solved; this solution provides an initial "best bound" on the objective function value. During the *branching* procedure, constraints that fix variables assuming fractional values in a parent-node solution are systematically added and two new LP relaxation problems are solved at the child nodes. For example, if an LP relaxation solution sets a binary decision variable equal to 0.79, then the two child branches from this node would include constraints forcing the variable to assume a value of 0 or 1, respectively. If more than one variable restricted to be an integer in the original problem assumes a fractional value, then we select one branching variable arbitrarily. A branch of the tree is fathomed when: (i) the branch results in an integer-feasible solution, (ii) the branch cannot result in a better objective function value than the incumbent, best integer-feasible solution, or (iii) when an LP relaxation problem is infeasible. Accordingly, when a branch cannot be fathomed, the branching procedure is again applied. The *bounding* procedure consists of updating, as necessary, a "best bound" on the optimal objective function value based on the LP relaxation solutions. The practical runtime of the algorithm relies both on the ease with which the LP relaxations are solved

and on the number of relaxations that must be solved, which is a function of the mathematical structure of the problem. Typically, bounding requires enumeration of only a fraction of the potential nodes, but in a worst case, full enumeration is required (see Rardin, 1998).

Although the branch-and-bound procedure eventually results in an optimal solution, run times of realistically-sized problems can become unwieldy. Some techniques used to expedite MIP solution times include: (i) eliminating extraneous integer variables, (ii) decomposing the problem into smaller subproblems and solving these, then reconstructing a solution to the original problem, and (iii) applying cutting planes, also known as valid inequalities, which excise integer-infeasible and/or suboptimal portions from the feasible region via additional constraints. We utilize a combination of (i) and (ii) to solve our Kiruna scheduling models, but we address, in Chapter 6, how we are unsuccessful in incorporating (iii) into our solution methodology.

4.1 Variable Elimination

Elimination of extraneous integer variables reduces the potential size of the enumeration tree for the branch-and-bound algorithm. This methodology dramatically improves solution times for previous Kiruna models (e.g., Kuchta *et al.*, 2004; Newman & Kuchta, in press) that only consist of the long-term resolution (machine placements). We apply similar rules to both sets of binary variables in our model (P), the y_{at} and w_{lt} variables. Based upon the principle of a critical path model (CPM), we may determine the earliest and latest possible start dates for mining machine placements (see Newman *et al.*, to appear). If we consider a directed, acyclic network, then every node, except for the source and sink nodes, may represent the activity of mining a machine placement. We connect nodes i and j with an arc if mining machine

placement i must directly precede mining machine placement j. The cost associated with arc (i, j) is the time required to mine machine placement i before we may begin mining machine placement j. Vertical sequencing constraints (3.12) dictate this duration of time, and require that arcs connect nodes based on the rule that once 50% of machine placement i is mined out, machine placement j can start to be mined. A variation of this method, incorporating both horizontal sequencing constraints (3.13) and LHD availability, allows for the elimination of y_{at} binary variables corresponding to time periods before a machine placement's earliest start date and after its latest start date. We label this restricted set of start dates (for machine placement a) T_a. Likewise, we apply a similar CPM procedure to the w_{lt} binary variables, which indicate completion of a drawdown line, to establish earliest and latest finish dates for each drawdown line. From these dates, we construct the set T_l. Although we model decisions to *finish* a drawdown line, as opposed to *starting* to mine a machine placement, we may apply similar precedence rules to overlying drawdown lines. We can then eliminate variables that denote finishing to mine a drawdown line either before its earliest finish date or after its latest finish date.

Although variables tracking the amount of ore mined from each production block per month (x_{bt}) are continuous, the same CPM principle can determine early start and late finish dates for the production blocks by constructing a set T_b. This reduction in the number of continuous-valued variables is not as effective as binary variable elimination because it does not directly impact the branch-and-bound procedure. Rather, elimination of continuous variables allows for faster LP relaxation solutions, which is not the principal factor affecting branch-and-bound solution times. Additionally, we may use the early start dates to eliminate irrelevant terms in constraints (3.3).

For a typical three-year scheduling model, we find that specifying: (i) early and late starts for machine placements, and (ii) early start, early finish, and late finish times for both drawdown lines and production blocks reduces the number of binary variables from over 6000 to under 1200, and the number of continuous variables from approximately 3900 to slightly less than 700. In addition to the variable elimination based on exact early and late start and finish dates, in previous research addressing only the long-term model, Newman & Kuchta (in press) use an optimization-based heuristic, which they label the *aggregation procedure*, to further eliminate y_{at} variables, as mentioned in Section 2.2.2.

4.2 Decomposition-based Heuristic ($\mathcal{H}$)

If we attempt to solve the baseline, three-year instance of (P) directly, we encounter extremely long solution times, even with the variable elimination techniques described in Section 4.1. The use of the early and late start and finish procedures, along with AMPL and CPLEX presolve algorithms, results in a model containing 1103 binary variables, 687 continuous variables, and 2675 constraints. If we attempt to solve this model instance to within 5% of optimality, the time required is in excess of five million seconds, or approximately 60 days. To further expedite solution times, we turn to heuristic methodologies.

We describe a decomposition-based heuristic that we apply to (P), the monolith integer program. The goal of the heuristic, which we label $(\mathcal{H})$, is to achieve near-optimal solutions more quickly than by solving (P) directly. The heuristic works by first solving decomposition-based subproblems that are similar to (P), but easier to solve. Then, we use information from the subproblem solutions to further constrain the solution space when solving (P). We expect this constrained revision of (P),

which we label (P'), to solve much faster than attempting to solve (P) directly. Two potential drawbacks in using a decomposition-based heuristic such as $(\mathcal{H})$ are that: (i) the optimal solution may be cut off by the procedure, and (ii) subproblems may still not solve to optimality very quickly (e.g., within 500 seconds).

We rule out decomposing the problem by time and/or space (via mine geometry) due to undesirable interaction effects between subproblems. As examples, previous Kiruna formulations (e.g., Topal, 1998; Dagdelen *et al.*, 2002) result in significantly suboptimal solutions when applying temporal decomposition. Similarly, spatially decomposing the model into two or more regions could generate conflicting guidelines when reconstructing the solution for (P') due to sequencing relationships between adjacent machine placements from different regions. Another difficulty arises when apportioning the ore-type demands among the different regions, since the composition of the ore reserves is not uniform.

The intuition behind our heuristic lies in subproblems, which, instead of utilizing the typical decomposition techniques mentioned above, emphasize decomposition with respect to ore type and the nature of the deviation. Recall that the objective function of (P) seeks minimum deviations from target production quantities; in essence, the purpose is to balance, for each ore type, both under- and over-deviations. We formulate subproblems whose objective functions capture aspects of the original objective function in (P). In our decomposition of the objective function of (P), we consider subproblem solutions that are "extreme" cases of what may be optimal values for the decision variables in the solution to (P). We formulate a total of five subproblems, primarily decomposing the objective function either by ore type or by the nature of the deviation (i.e., under- vs. over- deviations). This novel decomposition approach represents the key contribution of our work. We explain in Chapter 7

how this approach may be applicable to a variety of optimization models.

Each of the first three subproblems penalizes deviations only for a particular ore type $(B1, B2, D3)$. For subproblem (P_k), where $k \in \{B1, B2, D3\}$, then, the objective function reduces to:

$$\min \sum_{t} (p_t)(\underline{z}_{kt} + \overline{z}_{kt}) \quad \forall\, k \tag{4.1}$$

Additionally, we alter the constraint set as follows: when calculating deviation from target production quantities, we allow all three ore types to fulfill demand for the particular ore type of that subproblem. In this way, we consider that the entire mine is composed of only one ore type. Specifically, for subproblem (P_k), constraints (3.1) become:

$$\sum_{a \in A_t} \sum_{t' \in T_a, \leq t} \sum_{k \in K} r_{at'tk} y_{at'} + \sum_{b \in B_t} x_{bt} + \underline{z}_{kt} - \overline{z}_{kt} = d_{kt} \quad \forall\, t \in \mathcal{T} \tag{4.2}$$

which, assuming three ore types, reduces the number of constraints generated by $2k$. We now define subproblem (P_k) with (4.1) as the objective function, subject to constraints (4.2), and (3.2) through (3.14).

For the remaining two subproblems, we formulate each with a constraint set identical to that of (P), but with different objective functions. For subproblem (P_O), we penalize only over-deviations, while we penalize only under-deviations for the fifth subproblem, (P_U). We define (P_O) as:

$$\min \sum_{k,t} p_t \overline{z}_{kt} \tag{4.3}$$

subject to all constraints (3.1) through (3.14). Similarly, we define (P_U) as:

$$\min \sum_{k,t} p_t \underline{z}_{kt} \tag{4.4}$$

subject to all constraints (3.1) through (3.14).

4.2.1 Overview of $(\mathcal{H})$

We now outline the decomposition-based procedure as follows:

Heuristic $(\mathcal{H})$:

Step 1 Solve $k+2$ subproblems, (P_s), $s \in \{B1, B2, D3, O, U\}$.

Step 2 Utilizing information from the subproblem solutions, formulate and append constraints to (P), forming (P').

2a Append constraints applying to drawdown lines, the w_{lt} variables.

2b Append constraints applying to machine placements, the y_{at} variables.

2c Append constraints applying to both machine placements and drawdown lines simultaneously.

Step 3 Solve (P').

We now describe the three steps in detail below, with particular emphasis on **Steps 2a** through **2c**.

4.2.2 Detailed Description of $(\mathcal{H})$

We expect the subproblems of **Step 1** to solve quickly because fewer tradeoffs are necessary when optimizing only a component of the objective function of (P).

Note that all subproblem solutions are feasible for (P), and that the problem size (in terms of the number of variables and constraints) of each subproblem is similar to that of (P). When solving (P_s), we impose a time limit to prevent excessive run times for this phase of $(\mathcal{H})$. For the (P_{D3}) and (P_U) subproblems, we place an optimality gap as an additional stopping criterion. We assume that the five (P_s) subproblems run in parallel, so the required time to execute this portion of $(\mathcal{H})$ is the maximum among the five solution times.

After solving the five subproblems, we utilize the subproblem solutions to formulate additional constraints that we append to (P). We begin by examining the subproblem solutions with respect to each individual drawdown line l in **Step 2a**. We note the time period in which each drawdown line is finished being mined. Note that due to the way we define the w_{lt} variables, if a drawdown line is finished being mined in a particular time period t, then $w_{lt'}$, $t' > t$ (variables representing being finished with mining a drawdown line in subsequent time periods) may also assume the value of 1. We begin by defining the following parameter:

- e_{l,P_s} = first time period for which $w_{lt} = 1$ for drawdown line l in the solution to (P_s), $s \in \{B1,\ B2,\ D3,\ O,\ U\}$

The e_{l,P_s} value is the time period in which drawdown line l finishes being mined. Since we seek the earliest and latest finish dates for each drawdown line, we use these e_{l,P_s} values to define:

- $\underline{f}_l$ = earliest time period in which drawdown line l finishes being mined among all five subproblems $(= \min_s\{e_{l,P_s}\})$

- $\overline{f}_l$ = latest time period in which drawdown line l finishes being mined among all five subproblems $(= \max_s\{e_{l,P_s}\})$

We consider parameters $\underline{f}_l$ and $\overline{f}_l$ to provide a heuristic *window* of time periods in which a particular drawdown line would finished being mined in an optimal solution to (P). To enforce this restricted subset of time periods when solving (P'), we formulate two additional sets of constraints as follows:

$$\sum_{t<\underline{f}_l} w_{lt} = 0 \quad \forall\, l \in L \tag{4.5}$$

$$\sum_{\underline{f}_l \le t \le \overline{f}_l} w_{lt} \ge 1 \quad \forall\, l \in L \tag{4.6}$$

Constraints (4.5) prevent drawdown line l from being completely mined before time period $\underline{f}_l$, while constraints (4.6) require that we finish mining drawdown line l by the end of the heuristic window, time period $\overline{f}_l$. For instances where $\underline{f}_l = \overline{f}_l$, we fix $w_{lt} = 1$ in the solution in that time period. Together, these constraints effectively reduce the set of potential w_{lt} variables when we append them to (P).

In **Step 2b**, we apply a related procedure to the machine placements. We first note that when seeking an optimal schedule, decisions about machine placements differ fundamentally from those of drawdown lines. Assuming a schedule of two years or longer ($|\mathcal{T}| \ge 24$), all drawdown lines must be completely mined (finished) due to minimum production rates. This requirement facilitates the creation of windows of time periods during the time horizon for which a drawdown line must finish being mined. However, most machine placements have late start dates beyond the end of the time horizon, and therefore might not be mined at all during the schedule. Enforcing a window similar to that given by (4.5) and (4.6) for a machine placement start date may be infeasible if its late start date is beyond the end of the time horizon.

When examining each subproblem solution, we note whether a machine placement is mined, and if so, the time period in which mining begins. We check for the

following conditions, which are not collectively exhaustive, and formulate corresponding constraints:

- If none of the five subproblem solutions prescribe mining machine placement a at any point during the time horizon, then formulate constraints restricting (P') from mining machine placement a during the time horizon;
- If three or more subproblem solutions prescribe mining machine placement a during the time horizon, then formulate constraints forcing (P') to begin mining machine placement a at some point during the time horizon;
- If all five subproblem solutions prescribe beginning to mine machine placement a during the same time period, then formulate constraints forcing (P') to begin to mine machine placement a during that time period.

Note that the third condition is a subset of the second. To implement the appropriate constraints, we first define the following parameters and sets:

- y_{at,P_s} = value of y_{at} in the solution to subproblem (P_s)
- A^0 = set of machine placements a for which $y_{at,P_s} = 0 \ \forall \ (P_s), t$
- A^3 = set of machine placements a for which $\sum_{P_s,t} y_{at,P_s} \geq 3$
- A^5 = set of machine placements a for which $\sum_{P_s} y_{at,P_s} = 5$ for time period t
- $\hat{t}_a$ = time period for which $y_{at} = 1$ for machine placement $a \in A^5$ in each solution to (P_s), $s \in \{B1,\ B2,\ D3,\ O,\ U\}$.

The set A^0 denotes machine placements for which mining is not started in any of the subproblem solutions. We may restrict these (unmined) machine placements

from being mined in (P') with the following constraints:

$$\sum_{t \in T_a} y_{at} = 0 \quad \forall\, a \in A^0 \tag{4.7}$$

To ensure that the solution to (P') denotes beginning to mine all machine placements in the set A^3, we use the following set of constraints:

$$\sum_{t \in T_a} y_{at} = 1 \quad \forall\, a \in A^3 \tag{4.8}$$

Finally, we force the solution to (P') to specify beginning to mine machine placements that belong to the set A^5 during the appropriate time period:

$$y_{a,\hat{t}_a} = 1 \quad \forall\, a \in A^5 \tag{4.9}$$

In $(\mathcal{H})$, **Steps 2a** and **2b** address limiting or setting values for the w_{lt} and y_{at} binary variables independently. In **Step 2c**, we simultaneously restrict both w_{lt} and y_{at} variables with additional heuristic constraints which limit the number of active machine placements in the mine in each time period. Recall that (i) machine placements modeled in the short-term (drawdown lines and production blocks) are already active, and (ii) (P) already restricts the number of active machine placements within each shaft group with constraints (3.3).

Similar to the previous steps, we use the five subproblem solutions to determine a heuristic minimum and maximum number of active machine placements (MPs) allowed in each time period, in **Step 2c**. If we consider the number of active MPs as a proxy measure for the amount of "mining activity" occurring during a particular time period, then these constraints restrict the amount of mining activity through-

out the time horizon. Because we allow all ore types to satisfy the demand for the ore-type subproblems (see constraints (4.2)), the solutions to these three subproblems ($(P_{B1}), (P_{B2})$, and (P_{D3})) generally consist of less mining activity than that in the optimal solution to (P). Since (P_O) minimizes over-deviations, its solution also represents less mining activity than would be optimal for (P). By contrast, the (P_U) subproblem minimizes under-deviations, so we may consider its solution to represent more mining activity than would be optimal for (P). Therefore, we use subproblems $(P_{B1}), (P_{B2}), (P_{D3})$, and (P_O) to determine the heuristic *minimum* number of active MPs and use (P_U) exclusively to determine the heuristic *maximum* number of active MPs.

We implement these constraints by first defining the following parameters:

- $B1_t$ = number of active MPs in the solution to (P_{B1}) during time period t
- $B2_t$ = number of active MPs in the solution to (P_{B2}) during time period t
- $D3_t$ = number of active MPs in the solution to (P_{D3}) during time period t
- O_t = number of active MPs in the solution to (P_O) during time period t
- U_t = number of active MPs in the solution to (P_U) during time period t

We then define A^{min} and A^{max} for time period t as

- $A_t^{min} = \max\{B1_t,\ B2_t,\ D3_t,\ O_t\} \quad \forall\, t$
- $A_t^{max} = U_t \quad \forall\, t$

We modify constraints (3.3) to enforce these heuristic bounds with the following:

$$A_t^{min} \leq \sum_{a \in A_{t'}} \sum_{t' \in T_a, \leq t} \rho_{at't} y_{at'} + \sum_{a \in A} \sum_{l \in L_a \cap L_t} (1 - w_{l\hat{t}}) \leq A_t^{max} \quad \forall\, t \in T_l, \hat{t} \in \hat{T}_l \quad (4.10)$$

Note that the middle expression counts the number of active machine placements in time period t, which we then bound by A_t^{min} and A_t^{max}.

Considering all the additional constraints we formulate during **Step 2**, we may define (P') formally as:

(P'):

$$\min \sum_{k,t} (p_t)(\underline{z}_{kt} + \overline{z}_{kt})$$

subject to: Constraints (3.1) through (3.14), and constraints (4.5), (4.6), (4.7), (4.8), (4.9), and (4.10).

Note that (P') is (P) with additional constraints generated from $(\mathcal{H})$, **Steps 2a** through **2c**. In **Step 3**, we solve (P') to obtain a solution. As when solving the (P_s) subproblems, we impose a time limit when solving (P').

4.2.3 Implementation Guidelines for $(\mathcal{H})$

Since all subproblems (P_s) and (P') remain large MIP problems with sizes comparable to (P), we impose time limits on each problem type within $(\mathcal{H})$. Within the (P_s) subproblems, we discriminate subproblems (P_{D3}) and (P_U) from the other three because the deviations they minimize (under-deviations for the $D3$ ore type) typically constitute the majority of the deviations in optimal schedules for (P). Among other reasons, this characteristic seems to contribute to making them the most difficult to solve of the subproblems, since the lower bounds seem particularly weak in these cases. In addition to the time limit, we also impose an optimality gap as a stopping criterion. The time limits are 250 seconds for the subproblems (P_s); for (P_{D3}) and (P_U) specifically, the criteria are min{250 seconds, 1% optimality gap}. The rationale for the additional stopping criterion for these two subproblems is twofold: (i) experi-

mental runs exhibit long solution times (e.g., over 1000 seconds), and (ii) we consider solutions within 1% of optimality sufficient due to the heuristic nature of the procedure. If subproblems (P_{D3}) and (P_U) run to optimality, their solutions may slightly alter specific constraints we generate in ($\mathcal{H}$), **Step 2**. When using optimal solutions to (P_{D3}) and (P_U) in practice, we realize minimal improvements in the solution of (P').

From empirical analysis of twelve lengthy (P') runs (e.g., 3000 seconds or more), we notice that almost all of the improvement (typically, about 95%) in the objective function is made in the first 750 seconds. Additionally, as we describe in Section 5.1, we set the solver parameters to aggressively seek an optimal solution, so we expect improvement in the objective function value to diminish over time. Therefore, we set a time limit of 750 seconds when solving (P'). Typically, once the solver encounters a near-optimal solution, it spends a high percentage of the remaining solution time improving the lower bound. Using the example of our baseline 3-year Kiruna model, which requires over 60 days to achieve a 5% gap, we note that the solver obtains the ultimate solution approximately 60% into the run—so the final 40% of the run time merely improves the lower bound.

Chapter 5

COMPUTATIONAL RESULTS

We present computational results from implementing Heuristic ($\mathcal{H}$) described in Chapter 4 and compare it to solving (P) directly. When executing ($\mathcal{H}$), we incorporate into the model the previous variable elimination methods based on the early and late start and finish dates described in Section 4.1. We conduct all numerical experiments with the AMPL programming language (Fourer *et al.*, 2003; Bell Laboratories, 2001) and the CPLEX solver, Version 9.1 (ILOG Corporation, 2005). We execute the timed runs on a Sunblade 1000 computer with 1 GB RAM, while also conducting additional runs calculating lower bounds on a Beowulf Parallel Cluster with 96 processors and 280 GB of total RAM.

5.1 CPLEX Parameter Settings

We tailor the CPLEX parameter settings to obtain efficient performance for each of the three problem types: the (P_s) subproblems, (P'), and (P). In general, we rely on CPLEX's relaxation induced neighborhood search (RINS) heuristic and its MIP emphasis (MIPEMPHASIS) feature.

Since we wish the five subproblems (P_s) to solve to optimality as quickly possible, we utilize the RINS heuristic heavily (i.e., every 20 nodes) to search for the optimal solution. The MIP emphasis setting we employ directs the branch-and-bound algorithm to focus on improving the lower bound, rather than finding additional improving solutions. The two settings, while seemingly contradictory, work well together to find

an optimal or a near-optimal solution very quickly; apparently, this combination is effective because the subproblems are relatively simple to solve in comparison to (P).

When solving (P'), we assume that we do not easily attain a provably optimal solution. Instead, we focus on achieving near-optimal solutions as quickly as possible. We apply the RINS heuristic every 20 nodes and set the MIP emphasis to focus on finding new feasible solutions. The two settings complement each other, each with a goal of exploring many feasible (and perhaps improving) solutions without regard to improving the lower bound. We find this combination best to achieve the goal of near-optimal solutions in a short time.

As with (P'), we also assume that we do not obtain a provably optimal solution when solving (P). We again utilize the RINS heuristic to search for improving solutions quickly, without regard to the strength of the lower bound. We apply RINS every 100 nodes and leave the MIP emphasis at its default setting, which balances finding new solutions with improving the lower bound. Performing the RINS heuristic at 100-node intervals seems to yield better results than employing the heuristic more frequently (e.g., every 40 or 60 nodes). Less frequent application of the heuristic apparently allows more time for the solver to explore various regions of the branch-and-bound tree, yielding better solutions.

5.2 Original Kiruna Scenario and Perturbed Datasets

Our baseline scenario possesses current data from LKAB's Kiruna mine. The dataset contains three ore types, and spans three years or 36 (monthly) time periods. From this scenario, we perturb the dataset in three ways: (i) changing labor availability, (ii) changing equipment availability, and (iii) changing ore reserves. In (i) and (ii), we simply rearrange existing mine capacity. We use a proxy in (iii) instead of

perturbing the ore reserves directly.

At Kiruna, manual labor is required to prepare a machine placement for blasting and subsequent mining. The current Kiruna dataset allows up to two machine placements to start to be mined each month ($LHD_t = 2 \ \forall \ t$) in constraints (3.4). Assuming a time horizon of T months, a maximum of $2T$ machine placements may start to be mined; we rearrange this capacity among the time periods. We first modify the dataset by randomly assigning LHD_t values between 1 and 3 for each time period t. We then randomly select time periods from which to add or subtract LHD_t values until $\sum_t LHD_t = 2T$, while ensuring $1 \ \leq LHD_t \leq 3 \ \forall \ t$.

Although individual monthly LHD_t values of 1 or 3 represent a decrease or an increase of 50%, we do not allow four or more subsequent months to have identical perturbed LHD_t values (either 1 or 3). We consider a consistent deviation over that length of time or greater to be excessive. We correct such excessive deviation by randomly augmenting or removing capacity (LHD_t) within the pattern. In this way, we ensure that over any four month period, $5 \leq \sum_t LHD_t \leq 11$. Note that in the baseline dataset, any four-month period would set $\sum_t LHD_t = 8$; here, we are simply restricting that sum from equaling the extremes of 4 or 12.

Per constraints (3.3), each shaft group v operates either two or three LHDs, totaling 25 LHDs within the mine. We modify equipment capacity by rearranging the 25 LHDs among the ten shaft groups. We begin by randomly assigning each shaft group v either two or three LHDs: $2 \ \leq LHD_v \leq \ 3 \ \forall \ v$. Then, in order to maintain exactly 25 LHDs in the perturbed dataset, we add or subtract LHDs from random shaft groups, while maintaining $2 \ \leq LHD_v \leq \ 3$ in each shaft group. We consider assigning either 1 or 4 LHDs to one shaft group to be an excessive deviation from the original scenario.

We do not have sufficient information to perturb the ore reserves and/or target demand quantities in a realistic manner. As a proxy for perturbing the ore reserves, we shorten or lengthen the time horizon (from the baseline of three years) by six months. This proxy is valid due to the nature of minimizing deviations; for different time horizons, an optimal solution must minimize deviations from different sets of ore reserves and demands. The perturbations result in time horizons of 2.5 years and 3.5 years ($|\mathcal{T}| = 30$ and 42, respectively).

We consider each of the three methods of data perturbation to result in realistic Kiruna scenarios. Note that we simultaneously perturb LHD_t and LHD_v when generating the modified datasets, as well as shorten or lengthen the time horizon, as necessary. Using the baseline Kiruna data and only altering the length of the time horizon, we obtain three datasets: 2.5-year, 3-year, and 3.5-year. Then, for each time horizon we generate four additional datasets, yielding a total of 12 perturbed datasets. We generate each perturbed dataset independently. The three baseline cases and the 12 perturbed datasets yield 15 datasets against which we compare the performance of $(\mathcal{H})$ to that of solving (P) directly.

5.3 Performance Metrics

To compare the performance of $(\mathcal{H})$ to solving (P) directly, we propose two approaches. The two methods are based on (i) time elapsed until a particular solution is obtained, and (ii) objective function value (i.e., solution quality after applying a particular time limit). For each approach, we maintain consistent CPLEX parameter settings (e.g., application of the RINS at particular node intervals), as described in Section 5.1.

We label the first approach the Time-based Performance Metric (TPM). In this methodology, we perform the following steps:

1. Run Heuristic ($\mathcal{H}$) until an optimal (with respect to (P')) solution is obtained and/or appropriate time limits are reached (250-second time limit for subproblems (P_s) and 750-second time limit for (P'))

2. Record the objective function value, $V_{\mathcal{H}}$, and the time required, $T_{\mathcal{H}}$, where $T_{\mathcal{H}} \leq 1000$

3. Solve (P) until $V_{\mathcal{H}}$ is equaled or just surpassed

4. Record the time required, (T_P)

5. Calculate the metric:

$$TPM = \frac{T_P - T_{\mathcal{H}}}{T_P} \times 100\ (\%) \tag{5.1}$$

The TPM metric is a percentage that may be either positive (indicating that ($\mathcal{H}$) outperforms solving (P) directly) or negative (indicating the opposite). When calculating $T_{\mathcal{H}}$, we assume that the five subproblems (P_s) run in parallel, so we assign only the maximum time ($\max_s\{(P_s)\}$ run times) to ($\mathcal{H}$). To the maximum (P_s) run time, we add the run time for (P') to obtain the total solution time required for ($\mathcal{H}$), $T_{\mathcal{H}}$.

We call the second approach the Objective Function Value-based Metric (OFM). Recall that our model is a minimization problem; lower objective function values are better. In this methodology, we perform the following steps:

1. Run Heuristic ($\mathcal{H}$) until an optimal (with respect to (P')) solution is obtained

and/or appropriate time limits are reached (250-second time limit for subproblems (P_s) and 750-second time limit for (P'))

2. Record the objective function value, $V_{\mathcal{H}}$, and the time required, $T_{\mathcal{H}}$, where $T_{\mathcal{H}} \leq 1000$

3. Solve (P) for exactly $T_{\mathcal{H}}$ seconds

4. Record the corresponding objective function value for (P), (V_P)

5. Calculate the metric:

$$OFM = \frac{V_P - V_{\mathcal{H}}}{V_P} \times 100\ (\%) \tag{5.2}$$

Like the TPM metric, the OFM metric is a percentage that may either be positive (indicating that ($\mathcal{H}$) outperforms solving (P) directly) or negative (indicating the opposite). We use these two metrics because, in practice, the Kiruna mine seeks near-optimal solutions quickly. If ($\mathcal{H}$) outperforms the default approach of solving (P) directly, based upon these two metrics, then we may reasonably conclude that it will, on average, outperform solving (P) directly on realistic datasets. A robust heuristic would allow Kiruna to experiment with different mining configurations, make minor corrections to reserve/demand data as necessary, and answer "what-if" questions based upon realistic scenarios.

5.4 Results

The size of (P) is directly related to the length of the time horizon. For the 2.5-year models, the problems contain approximately 870 binary variables, 650 continuous variables, and 2160 constraints. In general, the 3-year models contain 1100 binary

variables, 690 continuous variables, and 2675 constraints, while the 3.5-year models contain about 1350 binary variables, 730 continuous variables, and 3260 constraints. On average, we find that (P') consists of 27% fewer binary variables, 14% fewer continuous variables, and 21% fewer constraints, when compared to its respective (P) instance. These reductions from $(\mathcal{H})$ are in addition to those we realize after applying the variable elimination techniques of Section 4.1. We also note that, because of the heuristic nature of $(\mathcal{H})$, the reductions come with a risk of chopping off optimal and/or near-optimal solutions.

5.4.1 Comparison of Solution Times

We compare solution times of $(\mathcal{H})$ and solving (P) directly and present summary results in Table 5.1. Each row represents the performance of one of the 15 datasets and each column represents a specific problem type. Solution times are given, or, if necessary, in the case of subproblems (P_{D3}), the gap remaining after 250-seconds of run time. The next two columns show the solution time required for $(\mathcal{H})$ and the corresponding solution time for (P) to obtain an equal or better objective function value than $V_{\mathcal{H}}$, respectively. The final column displays the calculated TPM metric, referring to the reduction or increase in solution time when comparing $(\mathcal{H})$ to (P). We see that subproblems (P_{B1}), (P_{B2}), and (P_O) solve to optimality quickly, while subproblem (P_{D3}) sometimes requires the full 250-second time limit, particularly for the 3.5-year models. Solving (P_U) to within 1% of optimality never requires the full 250 seconds allotted. In all 15 instances, solving (P') requires 750 seconds and does not reach optimality (with respect to (P')). Recall that since we assume that the (P_s) subproblems run in parallel, the run time for $(\mathcal{H})$, given in column 7, is the sum of $\max_s\{(P_s)\}$ run times and the run time for (P'). We note that the time required

Dataset	Time (sec) or Gap (%) (P_{B1})	(P_{B2})	(P_{D3})	(P_O)	(P_U)	$(\mathcal{H})$	(P)	TPM
2.5-year								
(1)	1	1	14	41	14	791	†	99%
(2)	2	3	8	139	32	889	1799	51%
(3)	3	2	4	31	13	781	9274	74%
(4)	2	3	7	55	18	805	753	-7%
(5)	2	2	14	35	11	785	604	-30%
3-year								
(6)	3	5	21	37	12	787	1482	47%
(7)	4	5	140	36	35	890	62,270	99%
(8)	9	10	1.52%	52	57	1000	2157	54%
(9)	3	3	137	55	48	887	1611	45%
(10)	4	4	1.24%	41	64	1000	581	-72%
3.5-year								
(11)	2	3	1.73%	50	51	1000	2622	62%
(12)	5	5	1.91%	154	136	1000	†	99%
(13)	5	5	1.63%	149	132	1000	12,171	92%
(14)	13	14	2.84%	101	143	1000	†	99%
(15)	6	9	4.30%	40	115	1000	1192	16%

Table 5.1. Comparison of Solution Times. Times are given in seconds for (P_s), $(\mathcal{H})$, and (P) for 15 datasets. In cases where (P_s) reaches the 250-second time limit, we show the optimality gap (%). † signifies a run we stop at 100,000 seconds. The last column reports the TPM metric.

for $(\mathcal{H})$ is less than 1000 seconds for the 2.5-year models, while each of the 3.5-year models require the full 1000 seconds.

(P) outperforms $(\mathcal{H})$ in three instances (datasets (4), (5), and (10)). We attribute the strong performance of solving (P) directly on these datasets to CPLEX's RINS setting. Occasionally, the solver obtains a remarkably strong solution quickly. However, we note that this occurs more frequently on smaller problems; as the difficulty of the problems increases (i.e., as the time horizon lengthens to 3.5 years) $(\mathcal{H})$ consistently outperforms solving (P) directly. Additionally, we emphasize the

inconsistency of the solution times required when solving (P) directly—in three other instances, $(\mathcal{H})$ approaches a 100% reduction in solution time with respect to (P). The † symbolizes a run length of 100,000 seconds in which (P) does not attain an objective function value equal to or better than $V_{\mathcal{H}}$. We report 99% as the value for the TPM metric in datasets (1), (12), and (14); though we do not know the exact time required for (P), we know that $99\% < TPM \leq 100\%$. We find the mean TPM is +48.5%, meaning that, on average, $(\mathcal{H})$ requires about half the solution time of (P) to attain a particular solution quality.

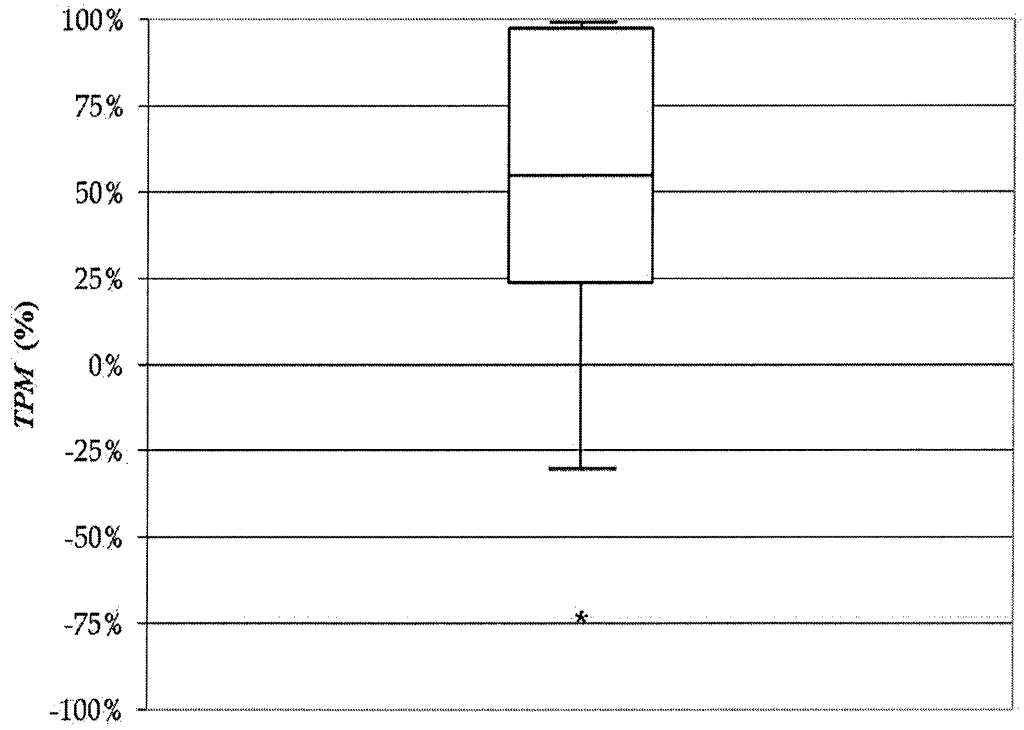

Figure 5.1. A box plot for the TPM metric. A horizontal line denotes the median of the dataset. The edges of the box represent the first and third quartiles of the data. The whiskers report the minimum and maximum values, while an asterisk denotes an outlier.

We present additional summary statistics in graphical form; specifically, the box plot shown in Figure 5.1. The plot displays quantile information, which refers to the percentage of data below a given value. The median TPM, depicted by the horizontal line, is 54.0%. The box edges depict the first ($Q1$) and third ($Q3$) quartiles (25^{th} and 75^{th} percentiles), which are 23.3% and 97.3%, respectively. We note that $Q1$ lies well above 0.0%. The whiskers, set at -30.0% and 99.0%, display the minimum and maximum of the dataset, while the asterisk denotes an outlier datapoint (-72%). We define an outlier, based on the interquartile range ($IQR = Q3 - Q1$), as a datapoint outside the range: $\tilde{x} \pm 1.5 \times IQR$, where $\tilde{x}$ denotes the median.

5.4.2 Comparison of Solution Quality

We compare the resulting objective function values and present summary results in Table 5.2. Each row represents the performance of one of the 15 datasets. Column 2 repeats the total solution time required for ($\mathcal{H}$), given in Table 5.1, for comparison purposes. Columns 3 and 4 display the resulting objective function values for ($\mathcal{H}$) and (P), respectively. The final column displays the calculated OFM metric, referring to the improvement or degradation in the objective function value when comparing ($\mathcal{H}$) to (P). We see that in the three instances where (P) outperforms ($\mathcal{H}$), the maximum degradation in the objective function value is less than 1.5% (dataset (10)). By contrast, in one instance (dataset (12)), ($\mathcal{H}$) attains an objective function value almost 16% less than that of (P). We calculate the mean OFM to be +3.19%, meaning that, on average, ($\mathcal{H}$) attains objective function values 3.19% less than those of (P) when both models are solved for identical time lengths.

We again present additional summary statistics in a box plot, shown in Figure 5.2. The median OFM is 2.79%. Here, $Q1$ is 0.27% and $Q3$ is 4.62%. Again, we find

Dataset	Sol'n Time ($\mathcal{H}$) (sec)	Weighted Deviation ($\mathcal{H}$)	(P)	OFM
2.5-year				
(1)	791	41,241	42,423	2.79%
(2)	889	42,986	43,173	0.43%
(3)	781	41,424	43,859	5.55%
(4)	805	43,320	43,183	-0.32%
(5)	785	42,961	42,945	-0.04%
3-year				
(6)	787	60,666	61,948	2.07%
(7)	890	61,677	65,681	6.10%
(8)	1000	68,938	69,632	1.00%
(9)	887	64,538	64,682	0.22%
(10)	1000	64,995	64,089	-1.41%
3.5-year				
(11)	1000	87,875	91,464	3.92%
(12)	1000	82,808	98,291	15.75%
(13)	1000	83,484	87,710	4.82%
(14)	1000	98,793	102,906	4.00%
(15)	1000	95,612	98,509	2.94%

Table 5.2. Comparison of Solution Quality. For 15 datasets, column 2 repeats solution times for ($\mathcal{H}$). Columns 3 and 4 display Objective Function Values for ($\mathcal{H}$) and (P), respectively. The last column reports the OFM metric.

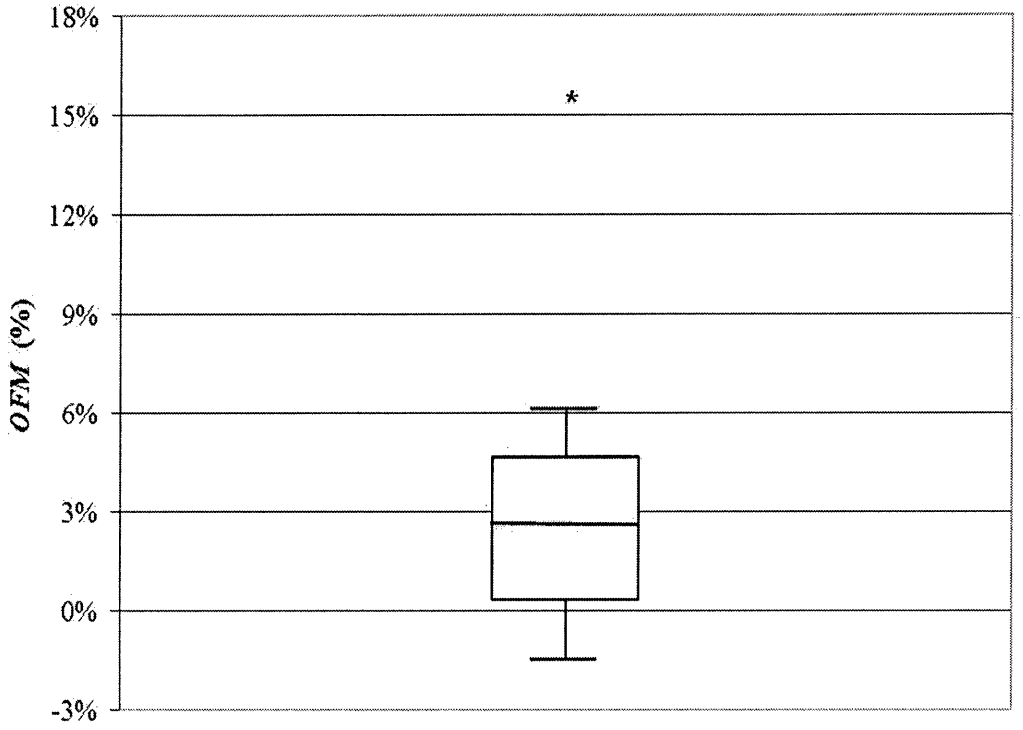

Figure 5.2. A box plot for the OFM metric. A horizontal line denotes the median of the dataset. The edges of the box represent the first and third quartiles of the data. The whiskers report the minimum and maximum values, while an asterisk denotes an outlier.

$Q1$ to lie in the positive region of the graph. The whiskers depict a minimum and maximum of -1.41% and 6.10%, respectively. We note one outlier (15.75%) denoting the poor performance of (P) in this instance.

We can extend the performance analysis by applying a statistical hypothesis test to see whether the expected value (mean) for each metric, $E(TPM)$ and $E(OFM)$, is greater than zero. Since the sample size is 15 for each test, we use the Student's t distribution as the basis for the hypothesis test. Applied to the TPM metric, our

hypotheses are:

$$H_0: \quad E(TPM) \leq 0 \tag{5.3}$$

$$H_1: \quad E(TPM) > 0 \tag{5.4}$$

We use a one-tailed test and a 0.05 level of significance. We note that the sample standard deviation is 0.5196 and the corresponding test statistic is $t = 3.495$. Since the critical statistic $t_{.95,14}$ is 1.753, we may reject H_0 and conclude that $E(TPM) > 0$ with 95% confidence. We approximate the p-value associated with this test statistic to be 0.0018, which means that we may place up to 99.75% statistical confidence in our conclusion from the hypothesis test.

Applied to the OFM metric, our hypotheses are:

$$H_0: \quad E(OFM) \leq 0 \tag{5.5}$$

$$H_1: \quad E(OFM) > 0 \tag{5.6}$$

We again use a one-tailed t test with a 0.05 level of significance. We calculate the sample standard deviation to be 0.0416 and the corresponding test statistic is $t = 2.867$. Again, the critical test statistic is $t_{.95,14} = 1.753$, so we may reject H_0 and conclude that $E(OFM) > 0$ with 95% confidence. We approximate the p-value associated with the test statistic to be 0.0062, which means that we would reject the null hypothesis at a confidence level up to 99%.

Applying $(\mathcal{H})$ to (P) effectively reduces model size and increases tractability. On average, if we spend equal time running each method, $(\mathcal{H})$ produces solutions that are 3% better than those of (P); if we desire a particular solution quality, $(\mathcal{H})$ yields a solution in about half the time required to solve (P) directly. We also find that if

solving (P) outperforms $(\mathcal{H})$, the differences are typically small, but when solving (P) loses to $(\mathcal{H})$, the loss can be exceptionally large. Parametric statistical hypothesis tests on these two metrics confirm our conclusion that $(\mathcal{H})$ outperforms solving (P) directly on realistic Kiruna datasets.

5.4.3 $(\mathcal{H})$ Solution Quality

We summarize the quality of solutions that we obtain with $(\mathcal{H})$ in Table 5.3. Due to the difficulty of solving (P) or (P') to optimality, we are not able to state with exact accuracy the quality of the solutions of these problems. As described in the introduction to Chapter 4, the branch-and-bound algorithm tracks a "best bound" (integer-infeasible solution to an LP relaxation problem) for the MIP, which becomes a basis against which to compare any integer-feasible solution. Since the objective function minimizes deviations, the best bound is a *lower bound* on the optimal amount of deviation. We may then use the lower bound to calculate an optimality gap associated with a particular solution. We report two different gaps (each calculated from different bounds), both with respect to solutions that we obtain with $(\mathcal{H})$. The results given in the second column of Table 5.3 utilize the resulting lower bounds associated with solving (P) for $T_{\mathcal{H}}$ seconds (i.e., runs used to calculate the OFM metric). We note that the gaps calculated from these bounds are weak, never less than 10%. By making separate, lengthy runs, we may obtain stronger lower bounds for these problem instances, allowing us to reduce the optimality gap, as shown in third column of the table. We perform these lengthy runs on a Beowulf Parallel Cluster with 96 processors and 280 GB of total RAM; we use the strongest CPLEX settings to improve the lower bound. Typical run times are between one and three weeks, depending on the time horizon of the dataset. Due to these time

Dataset	Gap ($\mathcal{H}$) (P) l.b. (%)	Gap ($\mathcal{H}$) strong l.b. (%)	Abs. dev. ($\mathcal{H}$) (%)	Abs. dev. th. min. (%)
2.5-year				
(1)	15.8	6.00	4.08	2.24
(2)	18.1	7.79	4.60	2.26
(3)	16.3	6.62	4.03	2.25
(4)	14.8	6.07	4.64	2.30
(5)	14.8	4.13	4.91	2.30
3-year				
(6)	27.5	7.20	5.00	1.89
(7)	26.2	9.70	4.70	1.94
(8)	31.6	9.77	5.48	2.08
(9)	29.9	11.9	5.60	1.92
(10)	29.3	10.5	5.51	2.02
3.5-year				
(11)	40.0	20.8	5.22	1.62
(12)	33.9	—	3.96	1.67
(13)	33.2	—	5.17	1.63
(14)	31.9	—	5.59	1.85
(15)	41.5	—	6.33	1.75

Table 5.3. Solution Quality of ($\mathcal{H}$). For 15 datasets, column 2 displays the optimality gap for solutions of ($\mathcal{H}$) using lower bounds obtained by solving (P) for $T_{\mathcal{H}}$ seconds. Column 3 displays the optimality gap for solutions of ($\mathcal{H}$) using stronger lower bounds obtained from separate runs. In column 4, we show the percentage of absolute deviation from the demand using solutions from ($\mathcal{H}$), while the final column reports the theoretical minimum percentage of absolute deviation.

requirements, we are not able to strengthen the lower bounds for all 15 datasets. At the time of this writing, we note that, at worst, ($\mathcal{H}$) produces solutions within 8% of the optimal solution for the 2.5-year time horizon. For the 3-year models, we can only prove optimality to within 12%, while the difficulty obtaining strong bounds for 3.5-year models prevents us from stating specific optimality gaps less than 20%.

We employ another measure to demonstrate the effectiveness of ($\mathcal{H}$) in producing strong schedules in a timely manner. Note that the total demand for the

2.5-year, 3-year, and 3.5-year time horizons are 59,052, 71,334, and 83,616 ktons, respectively. For a particular solution to $(\mathcal{H})$, we use the total absolute deviation (i.e., $\sum_{kt}(\underline{z}_{kt} + \overline{z}_{kt})$) and calculate a corresponding percentage of deviation from total demand. We report these percentages in column 4 of Table 5.3. Further, if we unweight the objective function (i.e., let $p_t = 1 \; \forall \; t$), then we may consider the solution to the root-node LP relaxation a "theoretical minimum" on the amount of deviation for a given problem instance. Likewise, we may calculate a theoretical minimum percentage of deviation, which we show in the final column of Table 5.3. We note that, on average, the percentage of absolute deviation we obtain with $(\mathcal{H})$ is approximately 5%, while the theoretical minimum percentage is about 2%. Currently, the long-term model in use at Kiruna results in schedules with total deviation of about 10%.

Chapter 6

LIMITATIONS AND EXTENSIONS

We have just shown the effectiveness of solving Kiruna scheduling models with Heuristic ($\mathcal{H}$). In this chapter, we discuss the limitations of ($\mathcal{H}$) and potential extensions to the methodology.

6.1 Limitations

When seeking near-optimal solutions to realistic Kiruna scheduling problems, we consider ($\mathcal{H}$) a better alternative to solving (P) directly; however, we also recognize various limitations when applying ($\mathcal{H}$). We categorize these limitations as: (i) a weak lower bound, (ii) degraded performance when applying ($\mathcal{H}$) to datasets that deviate excessively from what we consider realistic, and (iii) inappropriate time limits when solving component problems of ($\mathcal{H}$) in models with different time horizons.

6.1.1 Weak Lower Bound

As already mentioned in Section 5.4.3, the most significant limitation when attempting to solve either (P) or (P') to optimality is the weak lower bound. Weak lower bounds prevent proof of optimality for a particular solution. A typical reason why weak lower bounds occur (in minimization problems) includes an LP relaxation solution at the root node with numerous fractional values for its binary variables. We cannot identify a method to eliminate fractions in the initial LP relaxation solution, as we now discuss.

One method used to improve the lower bound is by adding valid inequalities, also known as cuts, to the formulation. Cuts should chop off otherwise-optimal LP relaxation solutions from the feasible region, while simultaneously not excising any integer-feasible solutions. Typically, cuts require a summation of binary variables to be greater than or equal to a constant, say b_t (referring to time period t). We explored various ideas to generate valid and useful cuts applied to the minimum number of active machine placements in each time period. The cuts would assume the following form:

$$\sum_{a} y_{at} + \sum_{l} w_{lt} \geq b_t \quad \forall\, t \tag{6.1}$$

Note that (6.1) resembles our heuristic implementation of **Step 2c** of $(\mathcal{H})$, as given in constraints (4.10) of Section 4.2. Here, we seek to specify a valid minimum number of active machine placements for each time period. When we compare the LP relaxation solution to an optimal integer solution, we find that in most time periods, a cut would not be useful. For example, in a particular time period, the LP relaxation solution may have 17.2 machine placements active, while the optimal solution has 17 machine placements active. In this case, specifying 17 active machine placements as a minimum would not be useful—the cut does not excise the LP relaxation solution. In instances where a cut would be useful, we have difficulty obtaining the effective right-hand side constants (b_t) easily. Initially, we attempted to simplify (P) by omitting individual constraint sets (e.g., constraints (3.4)), such that the resulting problem solves quickly. We expected the solution of this reduced problem to yield valid b_t values. In our experiments, we found no instances where this methodology produced valid and useful right-hand side constants.

Another possible method for obtaining valid right-hand side constants uses the

minimum production rates for the short-term production blocks. We may use the minimum production rates $\underline{C}_{at}$ (given in constraints (3.9)), to derive the number of drawdown lines that must finish in each time period. Note that this cut would take the form:

$$\sum_{l} w_{lt} \geq b_t \quad \forall\, t \tag{6.2}$$

However, the resulting cuts are not useful because the late finish date (within the set T_l) for each drawdown line adequately satisfies this minimum production requirement.

We also attempted to improve the lower bound with additional constraints using auxiliary variables. This implementation applied only to machine placements that must start to be mined within the time horizon. We created two sets of auxiliary variables, where the first set represented the aggregate amount of ore mined from a machine placement and the second set represented the sum of one or more y_{at} variables. By formulating constraints linking the auxiliary variables to the y_{at} variables, we could give branching priority to the latter set of auxiliary variables, which would drive the y_{at} variables to integral values. The integral y_{at} values in the optimal solution to the root-node LP relaxation problem should result in a much stronger lower bound. This implementation added a substantial number of binary variables to the model, and we found no noticeable improvements to the lower bound.

A crucial difficulty that we encounter when attempting to generate cuts (6.1) and (6.2) is due to the fact that it is not necessary to satisfy demand. Theoretically, our model allows "do nothing" (aside from short-term minimum production rates $\underline{C}_{at}$) as a feasible solution, albeit one with a poor objective function value. If requirements to fulfill demand existed, we could use these requirements to generate lower bounds. As an example of valid inequalities based on inventory bounds, generated in the

context of a lot-sizing problem, we refer the reader to Atamtürk & Küçükyavuz (2005). Since the Kiruna mine seeks minimum deviations from planned production quantities, our model elasticizes the demand accounting constraints (3.1). These elasticized constraints do not support valid inequalities based upon satisfying demand.

6.1.2 Unrealistic Datasets

The primary issue when generating the perturbed datasets of Section 5.2 is ensuring that the datasets remain realistic. Because of the complexity of the relationships between Kiruna mine geometry and the balance of mine production capacity with demand quantities, we find realistic datasets difficult to generate.

In terms of perturbing the number of machine placement starts (LHD_t), recall that we allow a deviation of 1 from the original two per time period ($= 2 \pm 1$). This change represents a 50% deviation for that one time period, but we consider such deviation in successive months unrealistic. Therefore, we enforce the rule (as stated in Section 5.2) that $LHD_t = LHD_{t-1}$ (either 1 or 3) for a maximum of three consecutive months. Since the number of machine placements starts is a measure of labor availability, we consider such deviations (effectively, a 50% increase or decrease) over multiple months to be excessive.

We also consider reducing the number of LHDs within a shaft group (LHD_v) to 1 or increasing it to 4 to be unrealistic. This principle leads to our perturbation rules detailed in Section 5.2 when modifying LHD_v. Additionally, we find that when the number of LHDs in certain shaft groups is reduced to only one, the problem instance becomes infeasible; this infeasibility occurs in not satisfying minimum production rates ($\underline{C}_{at}$) within production blocks (constraints (3.3)). Since we maintain a consistent number of LHDs within the mine (25), we consider assigning 4 LHDs

to certain shaft groups also to be unrealistic. We ensure representative datasets by simply rearranging the 25 LHDs, placing either 2 or 3 LHDs within each shaft group. This arrangement reflects the current LHD assignments of the Kiruna mine.

Initially, as a proxy for altering ore reserves, we randomly perturbed the demand quantities. However, given the absence of information about how these values might change, and the fact that the demand quantities are a result of long-term contracts set by Kiruna and the post-processing mills, i.e., they are fixed and not subject to change, we decided to hold the ore reserves and demands constant. Instead, we altered the time horizon as described in Section 5.2.

In addition to being unrealistic, we observe that ($\mathcal{H}$) behaves differently on such datasets, rarely outperforming solving (P) directly.

6.1.3 Specified Time Limits

We note that we tailor the problem time limits in ($\mathcal{H}$) specifically for the three-year time horizon. We recognize the time limits (250 seconds for subproblem (P_s), 750 seconds for (P')) as limitations if we change the length of the time horizon. Specifically, we consider the time limits excessive when the time horizon is two years or less; we consider the time limits inadequate when the time horizon is four years or longer. When we test ($\mathcal{H}$) against the baseline Kiruna data with a four-year horizon, 250 seconds is insufficient for (P_U) to obtain a feasible integer solution. We explain in Section 6.2 how one might resolve this timing issue.

6.2 Extensions

Based upon the results presented in Section 5.4, we show that ($\mathcal{H}$) is also effective for time horizons of both 2.5 years and 3.5 years. If we desire to solve models with

either the two-year or four-year time horizons, modifications to ($\mathcal{H}$) would likely be necessary.

Because the two-year model is relatively simple when compared to models with longer time horizons, we do not expect significant gains from a procedure similar to ($\mathcal{H}$) applied to a two-year model. Possible modifications to ($\mathcal{H}$) include reducing the time limits on both the subproblems (P_s) and (P') or changing the way in which we use information from the subproblem solutions when generating constraints to form (P').

Initially, one issue we encounter when applying ($\mathcal{H}$) to the four-year model with baseline Kiruna data is insufficient time for subproblem (P_U) to yield a feasible integer solution. A simple extension to the 250-second time limit would most likely solve this issue. Additionally, we note that the CPLEX settings for the subproblems (RINS heuristic every 20 nodes, emphasis on the lower bound) could be altered (i.e., shift emphasis to finding feasible solutions). When we change the MIP emphasis in this manner and execute ($\mathcal{H}$), we obtain a solution to (P_U) within 1% of optimality in less than 150 seconds. Utilizing this feasible (P_U) solution, we formulate and solve (P') for 750 seconds to obtain an objective function value of 117,230. If we run (P) for 1000 seconds (i.e., ($\mathcal{H}$)'s solution time), the resulting objective function is 122,022, which represents a .4% increase. If we continue solving (P) so that the objective function matches that of (P'), (P) requires a total solution time in excess of 5700 seconds.

Since Kiruna is interested in solving scheduling models with longer time horizons with combined resolution, the four-year example is a particularly interesting extension. One may also explore additional methods for generating cuts that would improve the lower bound.

Chapter 7

CONCLUSION

LKAB's Kiruna underground iron ore mine employs large-scale sublevel caving to extract three different ore types. The Kiruna mine is divided, given in order of increasing resolution, into: shaft groups, machine placements, drawdown lines, and production blocks. In its production scheduling, Kiruna seeks to minimize deviations from planned demand quantities. Previous scheduling (done by hand) would result in schedules that were either infeasible (with respect to operational and sequencing constraints) or highly suboptimal (consisting of deviations averaging about 15% from demand quantities). More recently, Kiruna has been using a long-term (machine-placement only) resolution mixed-integer programming model.

To optimize production scheduling at Kiruna, we enhance the existing long-term model and present a combined (short- and long-term resolution) model, also using mixed-integer programming. The model minimizes deviations for a production schedule with monthly time periods and incorporates various operational requirements unique to sublevel caving. However, the resulting model is large and solution times for schedules of requisite length are excessive.

Previous solution techniques utilized for the long-term model include variable elimination based on sequencing constraints. To expedite solution time further, we develop a decomposition-based heuristic consisting of two phases: (i) solving five subproblems, and (ii) solving a modified version of the original model, based on information gained from the subproblem solutions. The subproblems primarily modify

the objective function and capture "extreme" cases the original model must consider when minimizing deviations. Since typical decomposition techniques would decompose the problem by time and/or by space, we consider our approach to be a unique contribution.

We compare performance of the heuristic to solving the original model directly on 15 datasets. On average, we find that our heuristic obtains better solutions faster than solving the original problem directly. We also note the consistency of the heuristic when compared to the default solution method. In instances where the default method outperforms the heuristic, the differences are minimal; however, when the heuristic outperforms the default method, the results may be extreme. In general, the heuristic produces schedules that only deviate from the demand quantities by a total of about 5% in 1000 seconds or less. We consider the heuristic to contribute to solving the production scheduling problem at the Kiruna mine. A robust heuristic such as ours allows Kiruna to experiment with different mining configurations, make minor corrections to reserve/demand data as necessary, and answer "what-if" questions based upon realistic scenarios. Additionally, this decomposition scheme would apply not only to other mines with similar objectives, but also to other scheduling problems where deviation from planned production quantities, based on a resource-constrained scenario, is minimized.

We present various limitations to our approach, primarily discussing the weak lower bound that we encounter and the difficulties associated with resolving this issue. We discuss the importance of applying the heuristic to realistic datasets and how our specified time limits (based on the three-year model) may be inappropriate when the time horizon changes significantly. We also suggest possible extensions by modifying the heuristic when solving for schedules of greater length.

REFERENCES

Almgren, T. 1994. *An approach to long range production and development planning with application to the Kiruna mine, Sweden.* Ph.D. thesis, Luleå University of Technology, Sweden. Doctoral Dissertation 1994:143D.

Altas Copco. 2005. *http://img01.atlascopco.com/lowres/jpeg/202545.jpg.*

Atamtürk, A., & Küçükyavuz, S. 2005. Lot-Sizing with Inventory Bounds and Fixed Costs: Polyhedral Study and Computation. *Operations Research*, **53**(4), 711–730.

Barbaro, R.W., & Ramani, R.V. 1986. Generalized multiperiod MIP model for production scheduling and processing facilities selection and location. *Mining Engineering*, **38**(2), 107–114.

Bell Laboratories. 2001. *AMPL.* Version 10.6.16.

Carlyle, W.M., & Eaves, B.C. 2001. Underground planning at Stillwater Mining Company. *Interfaces*, **31**(4), 50–60.

Dagdelen, K., Kuchta, M., & Topal, E. 2002. Linear programming model applied to scheduling of iron ore production at the Kiruna mine, Kiruna, Sweden. *Transactions of the Society for Mining, Metallurgy, and Exploration*, **312**, 194–198.

Fourer, R., Gay, D., & Kernighan, B.W. 2003. *AMPL: A Modeling Language for Mathematical Programming.* Pacific Grove, CA: Thompson Learning.

Hochbaum, D.S., & Chen, A. 2000. Performance analysis and best implementations of old and new algorithms for the Open-Pit Mining Problem. *Operations Research*, **48**(6), 894–914.

ILOG Corporation. 2005. *CPLEX.* Version 9.1.

Jawed, M. 1993. Optimal production planning in underground coal mines through goal programming—A case study from an Indian mine. *Pages 43–50 of: Proceedings,* 24^{th} *International Symposium.* Application of Computers in the Mineral Industry (APCOM), Montreal, Quebec, Canada.

Kuchta, M. 2002. A database application for long term production scheduling at LKAB's Kiruna Mine. *Pages 797–804 of: Proceedings,* 30^{th} *International Symposium.* Application of Computers in the Mineral Industry (APCOM), Phoenix, AZ.

Kuchta, M., Newman, A., & Topal, E. 2004. Implementing a production schedule at LKAB's Kiruna Mine. *Interfaces*, **34**(2), 124–134.

Lerchs, H., & Grossman, I.F. 1965. Optimum design of open pit mines. *Transactions, Canadian Mining Institute*, **68**, 17–24.

Newman, A., & Kuchta, M. in press, 2006. Using aggregation to optimize long-term production planning at an underground mine. *European Journal of Operational Research.* (currently available online).

Newman, A., Kuchta, M., & Martinez, M. to appear, 2006. Long- and short-term production scheduling at LKAB's Kiruna Mine. *In:* Weintraub, A., & Epstein, R. (eds), *Handbook of Operations Research in Natural Resources.* Springer.

Rardin, R.L. 1998. *Optimization in Operations Research.* Upper Saddle River, NJ: Prentice Hall.

Sarin, S., & West-Hansen, J. 2005. The Long-Term Mine Production Scheduling Problem. *IIE Transactions*, **37**(2), 109–121.

Smith, M.L. 1998. Optimizing short-term production schedules in surface mining: Integrating mine modeling software with AMPL/CPLEX. *International Journal of Surface Mining*, **12**(4), 149–155.

Smith, M.L., Sheppard, I., & Karunatillake, G. 2003. Using MIP for strategic life-of-mine planning of the lead/zinc stream at Mount Isa Mines. *Pages 465–474 of: Proceedings,* 31^{st} *International Symposium.* Application of Computers in the Mineral Industry (APCOM), Capetown, South Africa.

Tang, X., Xiong, G., & Li, X. 1993. An integrated approach to underground gold mine planning and scheduling optimization. *Pages 148–154 of: Proceedings,* 24^{th} *International Symposium.* Application of Computers in the Mineral Industry (APCOM), Montreal, Quebec, Canada.

Topal, E. 1998. *Long and Short Term Production Scheduling of Kiruna Iron Ore Mine, Kiruna, Sweden.* Master of Science thesis, Colorado School of Mines, Golden, CO.

Topal, E. 2003. *Advanced underground mine scheduling using mixed integer programming.* Ph.D. dissertation, Colorado School of Mines, Golden, CO.

Trout, L.P. 1995. Underground mine production scheduling using mixed integer programming. *Pages 395–400 of: Proceedings,* 25^{th} *International Symposium.* Application of Computers in the Mineral Industry (APCOM), Brisbane, Australia.

Underwood, R., & Tolwinski, B. 1998. A mathematical programming viewpoint for solving the ultimate pit problem. *European Journal of Operational Research*, **107**(1), 96–107.

Wilke, F.L., & Reimer, T.H. 1977. Optimizing the short term production schedule for an open pit iron ore mining operation. *Pages 425–433 of: Proceedings,* 15^{th} *International Symposium.* Application of Computers in the Mineral Industry (APCOM), Brisbane, Australia.

Williams, J., Smith, L., & Wells, M. 1972. Planning of underground copper mining. *Pages 251–254 of: Proceedings,* 10^{th} *International Symposium.* Application of Computers in the Mineral Industry (APCOM), Johannesburg, South Africa.

Winkler, B.M. 1996. Using MILP to optimize period fix costs in complex mine sequencing and scheduling problems. *Pages 441–446 of: Proceedings,* 26^{th} *International Symposium.* Application of Computers in the Mineral Industry (APCOM), Pennsylvania State University, PA.

Winkler, B.M. 1998. Mine production scheduling using linear programming and virtual reality. *Pages 663–673 of: Proceedings,* 27^{th} *International Symposium.* Application of Computers in the Mineral Industry (APCOM), Royal School of Mines, London, United Kingdom.

Winkler, B.M., & Griffin, P. 1998. Mine production scheduling with linear programming—Development of a practical tool. *Pages 673–681 of: Proceedings,* 27^{th} *International Symposium.* Application of Computers in the Mineral Industry (APCOM), Royal School of Mines, London, United Kingdom.