Block-Cave Extraction Level and Production Scheduling Optimization under Grade Uncertainty

by

Saha Malaki

A thesis submitted in partial fulfillment of the requirements for the degree of

Master of Science

in
Mining Engineering

Department of Civil and Environmental Engineering
University of Alberta

ABSTRACT

Nowadays, application of massive mining methods has been increased due to the economic condition of mining companies. It is a step change for the industry, from the traditional open-pit to a move underground. Among the underground mining methods available, caving methods are favored because of their low-cost and high-production rates. They offer a much smaller environmental footprint compared to equivalent open-pit operations due to the much smaller volume of waste to be moved and handled.

Planning of caving operations poses complexities in different areas such as safety, environment, ground control, and production scheduling. As the mining industry is faced with more marginal resources, it is becoming essential to generate production schedules that will provide optimal operating strategies while meeting practical, technical, and environmental constraints. Unfortunately, common methodologies and tools used in mining industry for block-cave scheduling are not adequate in dealing with the complexity of the optimal production scheduling of mineable deposits. Also, the traditional long-term mine planning is based on deterministic ore-body models, which ignore the uncertainty in the geological resources. Grade uncertainty has profound impact on production targets which also impacts the financial expectations of the project. Initial evaluation of a range of levels for starting the extraction of block-cave mining is an important issue that needs to consider a variety of parameters including extraction rate, block height, discount rate, block profit, cost of mining and processing and revenue factors.

The objective of this study is to present a methodology to find the best extraction level and the optimum sequence of extraction for that level under grade uncertainty. A set of simulated realizations of the mineral grade is modeled based on stochastic sequential simulation to address this problem. The average grade of all the realizations is calculated and a new block model is generated, called average-simulated block model (first case study). Another case study is original block model which is created from the drillhole data. A method is introduced to find the best level to start extraction based on the maximum discounted ore profit. The best level of extraction is determined for all the realizations, original and average-simulated block models. Then, Maximum net present value (NPV) is obtained using a mixed-integer linear programming (MILP) model given some constraints such as mining capacity, production grade, extraction rate and precedence. Application of the method has been verified on both original and average-simulated block models for block cave production scheduling over 15 Periods. The best level for the original bock model was 38 and for average-simulated block model was 39. The obtained NPVs were $0.925B and $0.726B for the original and average-simulated block models, respectively, and all the constraints were satisfied. Finally, risks associated with grade uncertainty are investigated and analyzed which considerably helps the decision makers in better understanding of various cases and conditions. Among all the examined scenarios with unique scheduling parameters, the worst and the best case for the NPV were $0.85B and $1.081B, respectively. The ore tonnage also varies between 28.68Mt and 39.64Mt.

This Thesis is Proudly Dedicated To:

My wonderful parents, Maryam and Saeed,

my sweet brother, Sahand,

and my beloved husband, Kiarash.

ACKNOWLEDGMENT

First and foremost, I would like to thank my supervisor, Dr. Yashar Pourrahimian, who has consistently supported me throughout my MSc program. His professional advice, friendly behavior, and infinite patience, made the completion of this thesis possible. Moreover, I appreciate the guidance of all my friends in both Center for Computational Geostatistics (CCG) and Mining Optimization Laboratory (MOL), who remarkably helped me with their consultation and ideas. Especial thanks are extended to Firouz, Mohammad and Shiv who assisted me in learning mining engineering concepts and Mostafa who shared his Geostatistics knowledge with me. Additionally, I am grateful to my other colleagues for making office a friendly environment to pursue our purposes.

I am profoundly indebted to my devoted parents who mean a lot to me, Maryam and Saeed, for all encouragement and guidance during my life. They let me take a great opportunity to continue my education far from them through these two years. I would like to express my sincere appreciation to my mother, who always is a best source of peace and love, without her sacrifice, today I could not stand proudly with my head held high. I am thankful to my brother, Sahand, who I cannot imagine the world without him.

Along this journey, I met a very special person, the love of my life, Kiarash. I owe thanks to him for his continued and unfailing love. I deeply value his belief in me and his contribution. He always was around and constantly motivated me when the research seemed arduous and insurmountable.

Saha Malaki

September 2016

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

Parameters

EPGAP	Relative MILP Gap Tolerance
GP	Goal Programming
GSLIB	Geostatistical Software Library
LP	Linear Programming
MILP	Mixed Integer Linear Programming
NPV	Net Present Value
QP	Quadratic Programming
SGS	Sequential Gaussian Simulation
SIP	Stochastic Integer Programming
SIS	Sequential Indicator Simulation

LIST OF NOMENCLATURE

Set

S^{bbl}	For each big-block, bbl , there is a set S^{bbl} , which define the predecessor big-blocks that must be started prior to extracting the big-block bbl .

Indices

$bbl \in \{1,...,\mathrm{BBL}\}$	Index for the ore big-blocks
$bl \in \{1,...,\mathrm{BL}\}$	Index for small blocks.
$t \in \{1,....,T\}$	Index for scheduling periods.

Decision variables

$B_{bbl,t} \in \{0,1\}$	Binary variable controlling the precedence of the extraction of big-blocks. It is equal to one if the extraction of big-block bbl has started by or in period t; otherwise it is zero.
$\mathrm{x}_{bbl,t} \in [0,1]$	Continuous variable, representing the portion of big-block bbl to be extracted in period t.
$y_{bbl,t} \in \{0,1\}$	Binary variable used for activating either of two constraints.

Parameters

BBL	Number of ore big-blocks in the model.
$ExtL_t\,(Mt)$	Minimum possible extraction rate from each big-block in period t.
$ExtU_t\,(Mt)$	Maximum possible extraction rate from each big-block in period t.
g_{bbl}	Average grade of the element to be studied in big-block bbl

$GL_t\,(\%)$	Lower bound of acceptable average head grade of considered element in period *t*.
$GU_t\,(\%)$	Upper bound of acceptable average head grade of considered element in period *t*.
L	Arbitrary big number.
$MCL_t\,(Mt)$	Lower bound of mining capacity in period *t*.
$MCU_t\,(Mt)$	Upper bound of mining capacity in period *t*.
n	Number of predecessor big-blocks of big-block bbl
$\underline{N}_{NBBL,t}$	Lower bound for the number of new big-blocks, the extraction from which can start in period t
$\overline{N}_{NBBL,t}$	Upper bound for the number of new big-blocks, the extraction from which can start in period t .
$\mathrm{Pr}ofit_{bbl}$	Profit of each big-block which is equal to the summation of the small ore blocks' profit within that big-block.
T	Maximum number of scheduling periods.
Ton_{bbl}	Tonnage of each big-block.

CHAPTER 1

INTRODUCTION

This chapter is a general overview of the research. It concludes the background of the research; the problem statement; the objectives of the study, scope and limitations; the research methodology; and the contributions of the research.

1.1 Background

Mine planning determines the source, destination and sequence of extraction of ore and waste during the mine life. Production scheduling because of its significant impact on the project's value, has been considered a key issue to be improved. Production scheduling defines the tonnages and input grades to the plant throughout the mine life. The scheduling problems are difficult due to the natural complexity of minerals and variety of constraints on the system. Moreover, a production schedule must provide a mining sequence with respect to the operational and technical constraints and, to the extent possible, meet the target capacity of the processing plant.

The use of massive mining methods is increasing in major mining companies due to the economic issues of today's mining industry. Among the underground mining methods, block-cave mining could be considered as an appropriate alternative because of its low operation cost and high production rates (Pourrahimian, 2013). Block caving is the method that uses gravity to fracture a block of unsupported ore-body, allowing it to be extracted through pre-constructed drawpoints. It is more suitable for large, massive and low-grade ore-bodies. Block-cave mining offer a much smaller environmental footprint compared to equivalent open-pit operations due to the much smaller volume of waste to be moved and handled.

The uncertainties have profound impact on optimality of production schedules. Dimitrakopoulos (1998) has categorized these uncertainties into three types: (i) the ore-body model and variability of in-situ grade and material type; (ii) technical mining parameters such as mining capacity and (iii) economic factors such as capital and operating costs.

The initial evaluation of a range of levels for starting the extraction of block-cave mining is an important issue which affects the NPV of the project. To do this, it is necessary to consider a variety of parameters including extraction rate, block height, discount rate, block profit, cost of mining and processing and revenue factors.

One of the main steps involved in optimizing underground mines is determining a mining outline and inventory. The open-pit corollary to this is open-pit optimization, which is completed with algorithms such as those by Lerchs and Grossmann (1965).

To optimize block-cave mine scheduling, most researchers have used mathematical programming: linear programming (LP) (Guest et al. (2000), Hannweg and Van Hout (2001)); mixed-integer linear programming (MILP) (Song (1989), Chanda (1990), Winkler (1996), Rubio (2002), Rahal et al. (2003), Rubio and Diering (2004), Rahal (2008), Rahal et al. (2008), Weintraub et al. (2008), Smoljanovic et al. (2011), Pourrahimian et al. (2012), Parkinson (2012), Pourrahimian (2013), Pourrahimian and Askari-Nasab (2014)); and quadratic programming (QP) (Rubio and Diering (2004), Diering (2012)). LP is the simplest program for modelling and solving.

In this research, the main focus is to present a methodology to find the best extraction level and the optimum sequence of extraction for that level under grade uncertainty. A set of simulated realizations of the mineral grade is modeled based on stochastic sequential simulation. Maximum net present value (NPV) is determined for the block models using a MILP model after choosing the best level of extraction.

1.1 Statement of the problem

The proposed research is categorized in the area of applied operations research and stochastic sequential simulation.

The production schedule for a block-cave mine primarily is to define the amount of material to be mined from each block in each period of production and achieve a specific planning purpose (Pourrahimian, 2013).

Geostatistical simulation is used to model two kinds of geological uncertainty: rock type uncertainty and grade uncertainty within each rock type. The generated realizations provide reasonable outcomes to determine and assess uncertainty (Koushavand, 2014).

In traditional mine planning, Kriging (Goovaerts, 1997; Deutsch and Journel, 1998) as a common estimation is used to assign a grade to each block. But kriging does not reproduce uncertainty and its estimations are too smooth which causes consistently biased reserve estimates (McLennan and Deutsch, 2004).

The following research question drives this dissertation.

> *Is it possible to create a strategic production schedule for block-cave mines that will result in near optimal NPV, considering grade uncertainty and all the technical and operational constraints, such as mining capacity, grade blending, extraction rate, continuous extraction, number of new blocks, sequence of extraction, and reserves?*

In this research, block-cave production scheduling under grade uncertainty is studied. The ore-body is represented by a geological block model. Numerical data are used to represent each block's attributes, such as tonnage, density, grade, rock type, elevation, and profit data.

The first step is to construct an original block model based on the drillhole data and the grid definition. The next step is a geostatistical study to generate the realizations using drillholes data. Average grade of all the realizations (block models) for each cell is calculated to consider one block model instead of all the block models. This generated block model called average-simulated block model. Then, the best level of extraction based on maximum discounted profit is found for all the realizations, original and average-simulated block models. The most frequent level is chosen from the simulated block models. According to the results from both average-simulated block model and frequent level from the realizations, the best level is selected. The outline of the ore-body is determined at the best level and based on distances between drawpoints and the assumed footprint size, the blocks are placed into bigger blocks to decrease the number of variables and size of the problem. Finally, the optimal sequence of extraction is determined to maximize the NPV. The material in each big block is scheduled over T periods respecting the constraints. Figure 1.1 shows the summary of the methodology.

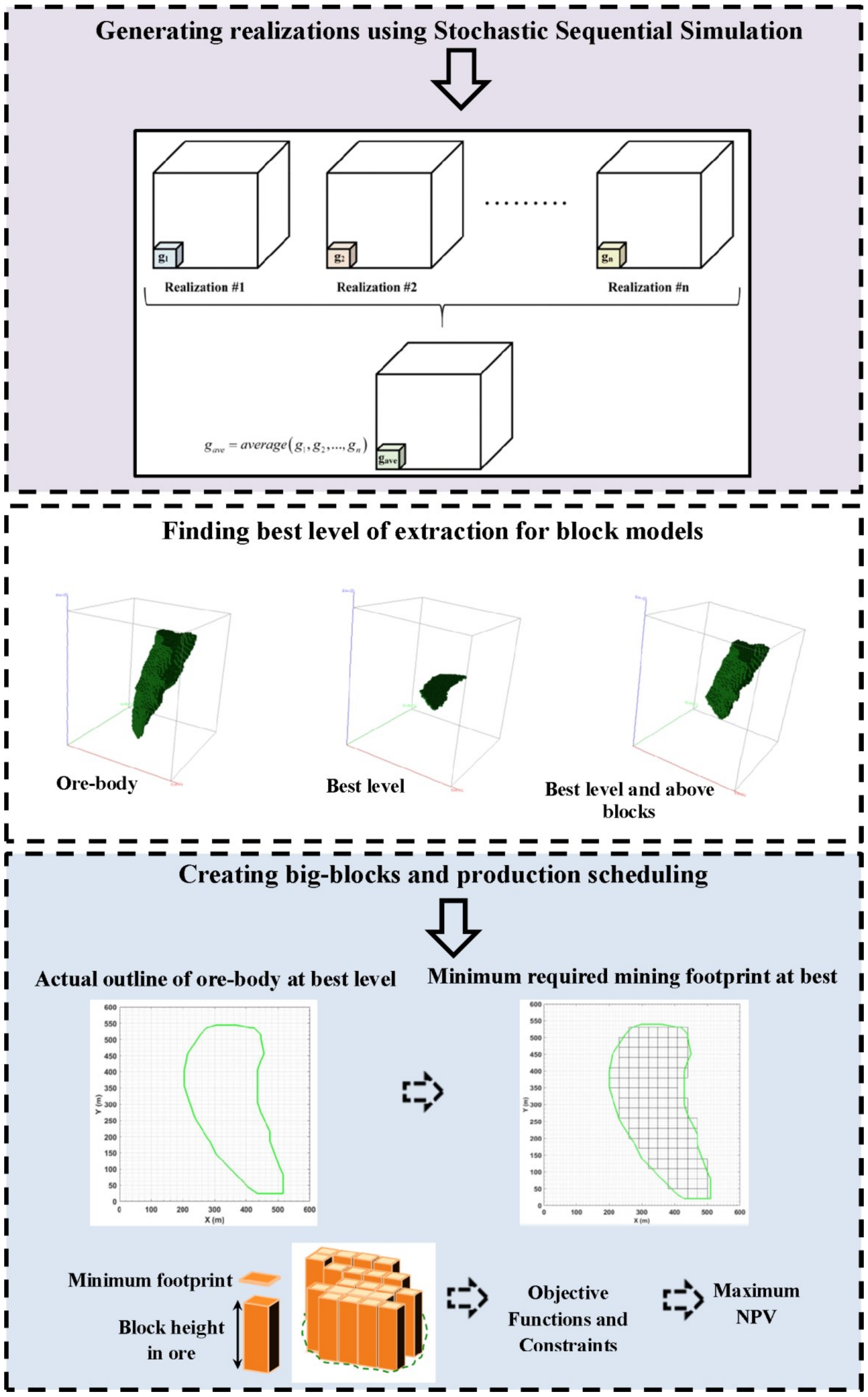

Generating realizations using Stochastic Sequential Simulation
g1
g2
gn
Realization #1
Realization #2
Realization #n
$g_{ave} = average(g_1, g_2, ..., g_n)$
gave
Finding best level of extraction for block models
Ore-body
Best level
Best level and above blocks
Creating big-blocks and production scheduling
Actual outline of ore-body at best level
Minimum required mining footprint at best
Y (m)
X (m)
Minimum footprint
Block height in ore
Objective Functions and Constraints
Maximum NPV

Figure 1.1. Schematic representation of problem definition

1.2 Summary of Literature Review

Mathematical programming optimization with exact solution methods have been proved to be powerful in solving production scheduling problems (Pourrahimian, 2013). Relying only on manual planning methods or computer software based on heuristic algorithms for solving a production scheduling problem cannot secure the optimal global solution. Scheduling more complicated mining systems is possible with great enhancements in recent years in computing power and scheduling algorithms (Alford et al., 2007).

Few numbers of researchers to date have studied production scheduling in underground mining. The complexity of underground mining is much more than the complexity of surface mining (Kuchta et al., 2004). Geotechnical, space and equipment constraints make underground mining less adoptable than the surface mining (Topal, 2008).

Various types of scheduling algorithms have been applied in production scheduling of underground mines. There are two main research areas in production scheduling algorithms and formulations: 1) heuristic methods and 2) exact solution methods for optimization (Pourrahimian, 2013).

Mathematical programming models that have been implemented in optimization of block-caving scheduling include: linear programming (LP), mixed-integer linear programming (MILP), goal programming (GP), and quadratic programming (QP) (Song (1989); Chanda (1990); Guest et al. (2000); Rubio (2002); Diering (2004); Rubio and Diering (2004); Rahal et al. (2008); Weintraub et al. (2008); Smoljanovic et al. (2011); Parkinson (2012); Epstein et al. (2012); Diering (2012); Pourrahimian et al. (2013); Alonso-Ayuso et al. (2014); Pourrahimian and Askari-Nasab (2014)).

Grade uncertainty can lead to significant differences between actual production and planning expectations and, as a result, the net present value (NPV) of the project (Osanloo et al., 2008; Koushavand and Askari-Nasab, 2009). Various researchers have considered the effects of grade uncertainty in open-pit mines. They introduced different methodologies to address those effects (Ravenscroft

(1992); Dowd (1994); Godoy and Dimitrakopoulos (2003); Ramazan and Dimitrakopoulos (2004); Leite and Dimitrakopoulos (2007); Dimitrakopoulos and Ramazan (2008); Koushavand (2014)). But less numbers of publications have concentrated on grade uncertainty in underground mining especially in block caving, where it is not so easy to revise production plans after caving has begun (Vargas et al. (2014); Grieco and Dimitrakopoulos (2007); Vargas et al. (2014); Montiel et al. (2015); Carpentier et al. (2016)).

The major limitations of the current production scheduling optimization in block-cave mining reviewed on chapter 2 are:

- Not taking into account detailed geotechnical constraints into real-scale production scheduling, extraction sequence and advancement directions.
- Not considering a solution for solving large-scale problems.
- Not considering the uncertainties associated with grade and only relying on deterministic models.

1.3 Objectives of the study

One of the objectives of this study is to consider grade uncertainty in finding the best level of extraction. A set of simulated realizations of the mineral grade is modeled based on stochastic sequential simulation.

Another objective of this study is to present a mathematical formulation to optimize a block-cave production scheduling in which the objective function is to maximize NPV with respect to technical and operational constraints. This objective consists of two elements: (i) developing a mixed-integer linear mathematical programming, (ii) verification of the mathematical formulation with real mining data.

The proposed methodology generates near-optimal schedules with respect to the following operational constraints: mining capacity, minimum required mining footprint, number of new big-blocks, extraction rate (production rate per block and per period), grade blending, continuous extraction, reserves, and extraction sequence.

The objectives of this study are to:

- Considering grade uncertainty using stochastic sequential simulation to decrease the effect of this uncertainty on the optimality of the project.
- Finding best level of extraction to start the mining operation based on the maximum discounted profit and grade uncertainty.
- Maximizing NPV of the mining operations with taking into account the effects of technical and operational constraints.
- Creating big-block columns to decrease number of blocks which reasonably reduces the amount of CPU time.
- Developing computer codes and tools to implement those mathematical formulations.
- Evaluating the results in terms of both feasibility and optimality of the solution using real mining data.

1.4 Scope and Limitations of the Study

This research is concerned with developing, implementing and verifying a MILP model to generate an optimal production schedule for block-cave mining in presence of some operational and geomechanical constraints and grade uncertainty. The model's objective is to maximize NPV of the block-caving operations subject to real world requirements and constraints.

The following assumptions are made in developing the mathematical model and stochastic sequential simulation:

- Size of the layout is fixed.
- Stationary domain within each rock type is assumed to be able to perform geostatistical modeling for each rock type separately.
- Developed MILP model is used to generate a production schedule only for block-cave mining.

There are some limitations in the problem due to the incorporated assumptions. In other words, detailed parameters such as drilling, blasting and ventilation are not considered in the model. Although grade uncertainty is examined to capture one of the uncertain inputs but other uncertainties such as prices, costs, recoveries and other practical mining constraints are still assumed to be deterministic. Any change in one of the mentioned input parameters needs re-schedule and re-optimization like all the deterministic models in capturing uncertainty.

1.5 Research Methodology

The main motivation for conducting this research is to improve block-cave production scheduling with considering more practical constraints in presence of grade uncertainty. There are two main steps to achieve this research's objectives: conducting *n* realizations based on drillhole data and perform geostatistical analysis and maximizing NPV through MILP model subject to a set of constraints. The following tasks should be completed to achieve this research's objectives:

- Implement geostatistical simulation to take grade uncertainty into account and generate a set of realizations.
- Develop a model using mathematical programming, specifically MILP which is a robust operations research method because of allowing binary variables.
- Verify the proposed model to make sure that whether the model works correctly according to the expectations or not.
- Implement the model through block-cave mine case studies to generate production schedule for the life of mine. For large scale mine problems, clustering small blocks into big-blocks will be used to not only decrease the number of variables but also decrease the run time.
- Assess the results of case study from practical point of view.

In the first part of this research geostatistical software library (GSLIB) (Deutsch and Journel, 1998) is used for geostatistical modeling. A number of realizations should be constructed using stochastic sequential simulation, based on the drillhole data. The following steps are implemented to achieve the objectives of the research in the first part:

1. Perform declustering which is used to get the representative distribution of each rock type to decrease the weight of clustered samples.
2. Perform multivariate statistical analysis to determine the correlation between the multivariate data.
3. Determine the principle direction of continuity for calculating variograms.
4. Transform data to Gaussian units.
5. Calculate the variograms.
6. Define the grid for simulation according to three parameters: (1) distance between the grid nodes in each direction, (2) the number of grid nodes in each direction and (3) the coordinates of the first grid node (Koushavand, 2014).
7. Generate *n* realizations (block models) using stochastic sequential simulation.

In the second part of this research the main focus is on developing a code to find the best level to start the extraction and an create big-block columns to reduce the size of the problem and developing MILP model to maximize the NPV with respect to practical constraints. The following steps are followed to complete this objective:

1. Find the best level to start the extraction for the average-simulated block model and all the simulated block models based on the maximum discounted profit. Based on the obtained results the best level is selected.
2. Determine the actual outline of the ore-body at the best level.
3. Create big-blocks based on the minimum required mining footprint inside the outline of ore-body at the best level to reduce the size of the problem. Weighted average grade, profit and total tonnage for each ore big-block are calculated. The big-blocks are the whole big-blocks above every footprint block at the selected level.

4. Define scheduling parameters.
5. Create objective function and constraints in MATLAB (Math Works Inc., 2015).
6. Solve the model using IBM/CPLEX (IBM, 2015) which uses a branch-and-bound algorithm to solve the MILP model, assuring an optimal solution if the algorithm is run to completion. An optimal production schedule is generated for different advancement directions. The direction with maximum NPV is employed as the mining direction.
7. A gap tolerance (EPGAP) is used as an optimization termination criterion. This is an absolute tolerance between the gap of the best integer objective and the objective of the remained best node.

1.6 Scientific Contributions and Industrial Significance of the Research

The main scientific contribution of this research is to develop, evaluate and implement a mathematical programming in context of block-cave mine production scheduling and in presence of grade uncertainty to find the best level of extraction and to maximize NPV. A need for an absolute solution for block-cave production planning under grade uncertainty is necessary especially with increasing use of massive mining methods. The summary of the contributions are:

- Combination of mixed-integer linear programming with geostatistical simulation in the context of block-cave mine planning.
- Analysis of grade uncertainty and studying its effect on mine planning and as a result on NPV of the underground mining operations.
- Determination of best level to initiate the extraction based on the maximum discounted profit and grade uncertainty.
- A method is presented which contributes significantly to reducing the number of variables and consequently solution time of the model.
- The proposed model can be applied on real size industrial applications as it has been tested on a real block-cave mine case study.

- The MILP model is subjected to a set of practical constraints including: mining capacity, grade blending, mining precedence, extraction rate, continuous mining, reserves and number of new big blocks.
- Introducing a methodology to determine the predecessor big-blocks and its implementation as prototype software with a graphical user interface.
- Development of a prototype open-source software application with the graphical user interface called Block Cave Footprint Optimizer (BCFO). The prototype software helps transfer knowledge and optimization technology developed in this thesis to practitioners and end-users in the field of block-cave production scheduling.

1.7 Organization of Thesis

Chapter 1 of this thesis is an introduction to the study. It concludes a general description about the background of the research followed by the statement of the problem, objectives, scope and limitation of the study, the proposed methodology and the contribution of the research.

Chapter 2 contains a literature review of the block-caving, production scheduling methodologies in both surface and underground mining. It provides a review about mathematical programming and its usage in mining operations. Also methods and models have captured grade uncertainty to date are discussed in both surface and underground mining.

Chapter 3 is concerned with geostatistical study followed by a theoretical framework for the MILP formulation for block-cave production scheduling optimization. At the first part, it provides steps for stochastic sequential simulation to assess the grade uncertainty. Calculating average-simulated block model is the next step of this part. At the second part, the best level to start extraction is found based on the maximum discounted profit. To overcome the size of real mining projects, a method is presented which lead to creating big-blocks. The MILP model aims to maximize the NPV of the project while controlling mining capacity, extraction rate, advancement direction and number of new big blocks in each period. The result is the optimal production schedule for the big-blocks which determines the active and new big-blocks in each period and the sequence and amount of extraction for each big-block. Finally the implementation of the MILP formulation and the creation of the matrices for each constraint and objective function are highlighted.

Chapter 4 provides the implementation of the proposed model and steps in chapter 3 on a real block-cave mining case studies. It describes how various components of the MILP model can be set in MATLAB (Math Works Inc., 2015) and how IBM/CPLEX (IBM, 2015) can be used to solve the problem. Various analysis and comparisons have been done to address the risks and instabilities in block-cave mining due to presence of grade uncertainty.

Finally, summary, contribution of the research and suggestions for future work are discussed in chapter 5.

CHAPTER 2

LITERATURE REVIEW

Chapter 2 provides a review of production scheduling algorithms in mining industry. Production scheduling methods in both underground and surface operations are discussed and compared. It contains a brief description about block-cave mining system, as well as usage of mathematical programming in the context of production scheduling of the block-cave mining. On the other hand, to deal with geological uncertainty several researchers provided various algorithms in open-pit mines but less in underground mining systems.grade uncertainty and ways to address its effects on mine planning are concluded as an important factor from the profitability perspective. The chapter contains the remarks and summary.

2.1 Block Caving

The use of massive mining methods is increasing in major mining companies due to the economic issues of today's mining industry. Among the underground mining methods, block-cave mining could be considered as an appropriate alternative because of its low operation cost and high production rates (Pourrahimian, 2013).

Additionally, block-caving is much more suitable for weak rock masses which needs small footprint to reach continuous mining. The advantages of using small area are: better controlling of cave foundation and production and more stability of panel cave front. On the other hand, current relatively near surface ore-bodies will be exhausted and the new ore-bodies now are found in much deeper depths. As a result the use of block-caving methods will be incredibly increased in the near future (Flores, 2014).

Block caving concerns with a mass mining operation and the extraction of ore material is done with the help of gravity. As a thin horizontal layer of the ore column is removed at the production level by using conventional mining methods, the vertical support of the ore column above is also removed. Then, ore caves by gravity. By removing the broken ore from the production level of the ore column, the above ore proceeds to break and cave by gravity (Julin, 1992). There are three forms of block caving: 1) creating rectangular or square blocks in the horizontal area and in this case the drawing should be constant over the whole area; 2) creating panels in the horizontal area across the ore-body and keep a sloping plane of contact between the broken ore and caved capping; and 3) the horizontal area is not divided into blocks or panels. In this method the total production demand and the size of the block are key factors in specifying the total active caved area (Tobie and Julin, 1998).

There are three principle methods in block caving: grizzly, slusher and LHD. Grizzly or gravity method is more appropriate for finer material essentially free-flowing to the drawpoint. For coarser material, the slusher method is used and for relatively coarsely fragmented deposits, Load-Haul-Dump (LHD) can be the best option (Brannon et al., 2011). Ore-body characteristics, types of ore, size and shape of the ore-body are important factors in selecting a mining method. Toughness or softness of the ore is another controlling factor that must take into account.

There are 25 parameters mentioned by Laubscher (1994) that should be considered before implementing any cave-mining. Some of these parameters are caveability, fragmentation, drawzone spacing, draw control, dilution entry, layout, undercutting, and support requirements. Caveability is an underlying factor for mine design. The important aspect in this element is hydraulic radius, which is the calculation of the area of the caved zone divided by the length of the perimeter (Brannon et al., 2011). Three steps have been defined for fragmentation. First one is in-situ fragmentation which is the natural movement of the blocks before initiating the caving operation. The second one is primary fragmentation which happens during the start of caving. The last one is secondary fragmentation in which the blocks moves through the draw column throughout mining process (Brannon et al., 2011). Various parameters should be considered in drawpoint spacing. Some of these parameters are: isolated draw zone diameter, the fragmentation for the bulk of the draw, the internal friction of the material to be drawn, and the number of drawpoints (Laubscher, 2003). Draw control means to govern the tonnages drawn from individual drawpoints while taking into account some objectives such as: minimizing the overall dilution, ensuring maximum ore recovery, preventing any destruction to the load concentrations, and avoiding having air blasts (Laubscher, 2003). Dilution can be happened due to extremely soft material which prevents the extraction of clean ore from blocks. To have minimum dilution, it is better that the dilution zone breaks into the same size as the ore or larger (Tobie and Julin, 1998). The rate of drawing fluctuates according to the caveability of the ore. The drawing of the caved material should be fast which allow continuing to cave upward through the rock mass (Tobie and Julin, 1998). The height of ore column is a principal component of block caving. It should be as much as that after deducting all the costs such as development, production, processing, and overhead still be profitable (Julin, 1992). The applications of block caving are in the following deposits: porphyry copper, molybdenite, limonite, diamond, asbestos, nickel, and magnetite (Tobie and Julin, 1998).

A block-cave mine consists of multiple horizontal levels including: undercut level, extraction level, ventilation level, and haulage level (Brannon et al., 2011). Undercut level lies directly with some distance above the production level. This level is made up of a series of parallel drifts from which a series of longholes are drilled. The longholes are loaded and blasted, then a thin horizontal layer of ore is removed and afterwards, removing the vertical support from the ore column above is continued (Tobie and Julin, 1998). Undercut level can be divided into three types: post-undercutting, pre-undercutting and advance-undercutting in which according to its name is started after, before or in advance of development of extraction level. The drawpoint drifts are developed in extraction level which lies above the haulage level. The ventilation level is appropriate for providing intake and exhaust ventilation across the production footprint for the other levels. Haulage level is also designed based on the required type of transportation (Brannon et al., 2011). Tobie and Julin (1998) have highlighted some advantages and disadvantages for block caving. The advantages include: 1) the cost of mining is low compared to other systems because of less number of drilling, blasting, and small amount of development operation per ton; 2) adequate control can be achieved through concentrated production which result in higher labor productivity and safe working conditions; 3) more control on ventilation system; 4) high rate of production; and 5) appropriate for low grade ore-bodies. The disadvantages are: 1) necessity to have more time and money for preparing blocks before starting production; 2) maintaining drifts in the draw area needs spending money; 3) ore recovery can be low due to unfavorable conditions; 4) changeable production due to increased demand for the product needs more time to prepare additional blocks for production; and 5) the method is inflexible and change to other underground mining method is hard.

2.2 Mine Production Scheduling

Mining industry is made up of different types of operations such as exploration, planning, extraction, transportation and processing. Huge investments and large amount of material are the common characteristics of the mining operations. Therefore, even a small improvement in any stage of the procedure can have a large effect on the net present value (NPV) of the project (Tabesh, 2015).

Evaluations of mineral resources to date showed that mine-planning decisions play a key role in profitability of the project (Askari-Nasab and Awuah-Offei, 2009; Newman et al., 2010). Osanloo et al. (2008) and Newman et al. (2010) have presented a complete review on mine planning.

Three time horizons have been defined for production scheduling: long-, medium- and short-term. Long-term mine-production scheduling develops a strategic procedure for mining operations, while medium-term scheduling prepares more detailed operational plan towards strategic procedure for ore extraction and purchasing necessary equipment. The medium-term schedule can be divided into short-term periods as well (Osanloo et al., 2008).

Operations research techniques can be helpful in improving the efficiencies of the operations and optimizing the plans, operation of equipment and other resources (Tabesh, 2015). Different operations research methods including linear programming (LP), mixed-integer linear programming (MILP), and quadratic programming (QP) have been used in field of mine planning.

Some of the benefits of production scheduling include optimum recovery of marginal ores, reduced costs, increased equipment utilization, reduced costs, high production rates, and consistent product quality (Dagdelen and Johnson, 1986; Chanda, 1990; Wooller, 1992; Chanda and Dagdelen, 1995; Winkler, 1996).

Most of the available scheduling publications are devoted to surface-mining operations, although complication of underground mining is much more than surface mining (Kuchta et al., 2004). Also underground mining is less adoptable than the surface mining due to the geotechnical, equipment, and space constraints

(Topal, 2008). As a consequence, many of the scheduling algorithms developed for surface mining are implemented in underground mining as well.

2.2.1 Open-Pit Production Scheduling

There are two steps in open-pit mine planning: 1) finding ultimate pit limit and 2) order of extraction of blocks in the determined ultimate pit limit (Tabesh, 2015).

The floating cone method developed in 1961 by Kennecott Copper (Kim et al., 1988) and the graph theory based approach by Lerchs and Grossmann (1965) which is known as the LG algorithm, are two popular techniques in finding the ultimate pit limit.

Researchers in surface-mining scheduling have tried to develop the optimum ultimate pit-limit algorithms instead of solving the optimal open-pit production schedule. Although, ultimate pit-limit method is almost easy to solve, but it just imposes limitations on pit slope, not on production. In optimizing the mine production various methodologies have been utilized as more production constraints should be taken into account by mining companies (Kim and Zhao, 1994).

Current production scheduling algorithms in literature have been divided into two categories: 1) heuristic methods and 2) exact solution methods for optimization (Askari-Nasab and Awuah-Offei, 2009).

Heuristic methods are the basis of developing most of the commercial software. These methods implement different alternatives to generate the ultimate pit limit which result in different discounted cash flow and as a result different NPV. The negative point of these methods is the possibility of sub-optimality of the solution (Pourrahimian, 2013).

Optimization with exact solution methods using mathematical programming have been proved to be powerful in solving long-term production scheduling problems. Using these methods contributes significantly to generating production schedules with higher NPVs than those obtained from heuristic optimization methods (Pourrahimian, 2013).

Implementation of LP model by Johnson (1969) was the earliest mathematical method. To decide on the extraction of the block in each period and its destination, continuous variables were defined. Afterwards, Gershon (1983) presented two mathematical models: LP and IP. The author used continuous variables for LP model indicating the portion of extraction for each block in each period. IP model consisted of binary variables that determine whether a block should be extracted in a period or not. Then, the differences between two models were studied. Integer variables are necessary for adding different constraints. It was one of the drawbacks of LP model. Another disadvantage of both of the models was the large size of the problem which was mentioned by Gershon (1983).

Caccetta and Hill (2003) used an MILP model for production scheduling and proposed a branch-and-cut algorithm for solving the problem. The disadvantage of the model stated by the authors was disability to find the optimal solution specifically for large-scale problems (Tabesh, 2015).

An MIP model was developed by Ramazan and Dimitrakopoulos (2004) which tried to reduce the number of variables, considering ore blocks as binary and the remaining variables as linear. This technique has not been applied to production scheduling in the mining industry due to its complexity in implementation. Moreover, Ramazan (2007) proposed a MILP model for production scheduling of open-pit mines in which a new algorithm called "Fundamental Tree Algorithm" was developed. This algorithm helped noticeably to decrease the number of integer variables and required constraints by aggregating blocks of material.

Boland et al. (2009) reduced the number of variables using blocks aggregation and MIP formulation for solving open-pit production scheduling problem. Aggregates were used in excavation decisions while individual blocks were used for processing decisions. In their work, iterative disaggregation method was proposed to refine the aggregates up to the point that optimal solution of aggregates level gained from LP relaxation of the MIP was equal to the optimal solution of LP relaxation of the blocks level. Different disaggregation methods were used and the resulted NPVs and CPU time were compared.

Another noticeable study in the field of aggregation was done by Askari-Nasab et al. (2011). The authors aggregated blocks into larger units called mining-cuts to decrease the number of integer variables. Four MILP formulations were presented which two of them were new and the other two were the modification of the available models. The performance between them was compared based on the generated net present value, practical mining production constraints, size of the model, number of required integer variables, and the computational time. An iron ore mine case study over 12 periods in TOMLAB/CPLEX (Holmstrom, 2011) environment was implemented to test the practicality of the formulations.

Meta-heuristics are also used in production scheduling problem. Sattarvand and Niemann-Delius (2008) reviewed meta-heuristics algorithms in open-pit mine planning.

Kumral and Dowd (2005) used a simulated annealing approach for mine production scheduling. In their method, three minimization objective functions were defined: deviation from the required tonnage, penalty and opportunity cost for each variable, and content variability of each variable. Lagrangean parameterization was applied toward an initial sub-optimal solution and then improved that solution by multi-objective simulation annealing.

Samanta et al. (2005) also used genetic algorithm for mine planning problems. In their research layering approach was utilized towards minimizing the cumulative grade deviations from the target ore grade for the entire schedule period of 12 months. Five best grade control schedules were generated which one of them could be chosen by the management.

Another technique for long-term open-pit production scheduling presented by Sattarvand and Niemann-Delius (2013) was ant colony optimization. A series of variables for each block were considered named pheromone trails that illustrate the desirability of the block for being at the deepest point of the mine for a specified period of time. The objective function was maximizing net present value.

Sayadi et al. (2011) determined the optimum pit limit using artificial neural network in presence of constraints of wall slopes and impurities. The 3D blocks coordinates were the inputs of neural network and blocks net economic values were the output of the model. Comparing the results with LG algorithm through a case study of phosphate mine showed that this new model generates a pit limit with higher profit because of considering ore impurity constraints.

Back to mathematical programming, another fresh paper in the field of mine production scheduling is proposed by Bley et al. (2010). They reduced the number of decision variables by combining the block precedence constraints with the production constraints. They have implemented and compared different techniques for single-block, multiple-block and, block conflicts cases which helped enormously to decrease the CPU time for the solver.

Cullenbine et al. (2011) presented a sliding time window heuristic for solving the open-pit mine block sequencing problem. They used this approach to divide the problem into smaller IP models and the time window was moved forward yearly to solve another problem. Their algorithm is able to solve the small problems with reasonable run time but not large model with 53,668 blocks.

Chicoisne et al. (2012) implemented three steps to solve the open-pit production scheduling problem: (1) solve the linear programming relaxation of the problem using decomposition method and with one resource constraint per time period, (2) rounding heuristic approach to the fractional solution of the previous step, and (3) improve the solutions by applying local-search heuristic.

Kumral (2012) used two MILP models: (1) with pre-defined cut-off grade and (2) without cut-off grade. The second model was able to discriminate ore and waste, so the binary decision variables contained the destinations as well. Using a case study of gold mine over five periods, authors reported 5% improvement in the NPV of the second model.

2.2.2 Underground Mines Production Scheduling

Current publications in underground-mine scheduling implement simulation and heuristic software which determine feasible solutions rather than optimal solutions. Almost identical to open-pit mines, heuristic methods and exact solution methods are two principal categories for production scheduling algorithms in literature (Pourrahimian, 2013). Besides, there are other methods that have been used for production scheduling and/or material transport including: queuing theory (Su, 1986; Huang and Kumar, 1994), dynamic programming (Sherer and Gentry, 1982; Muge et al., 1992) and network analysis (Russell, 1987; Brazil et al., 2000; Brazil et al., 2003).

There are successful applications of simulation in underground mining. Hanson and Selim (1975) implemented mine simulation for room-and-pillar mining. In this model stochastic variables were all event times. Drilling, blasting, mucking and proof bolting were integrated into a system that described equipment utilization, production tonnage, and the number of holes drilled using mine simulation. Another study using simulation system in mining was performed by Maxwell (1978). Mill feed grades were estimated given a considered drawpoint production rate. This work emphasized on the power of simulation to tackle the non-linearity of the ground mining system. But the optimality of the schedule was not guaranteed by the authors in both recent works. The simulation was used by Gerling and Helms (1986) to deal with the deposit reserves through a series of sub-models including the drift system, the development drifting operations, and the stopping operations.

Nehring et al. (2012) used two objective functions to incorporate short- and medium-term production schedules of a sublevel stopping mine. Minimizing deviation from target head grade for the short-term schedule and maximizing the net present value for medium-term schedule were the two objectives of the model. They applied their model in a case-study and a minor improvement in integrated model was shown in compared with solving two models separately. Newman et al. (2010) and Alford et al. (2007) presented complete reviews about the underground mine planning literature.

2.2.2.1 Production Scheduling in Block-cave Mines

Mathematical programming that have been implemented by various researchers for optimizing of block-cave mine production scheduling are: linear programming (LP) (Guest et al. (2000)); mixed-integer linear programming (MILP) (Chanda (1990), Song (1989), Rubio (2002), Diering (2004), Rubio and Diering (2004), Rahal et al. (2008), Weintraub et al. (2008), Smoljanovic et al. (2011), Parkinson (2012), Epstein et al. (2012), Pourrahimian et al. (2013), Alonso-Ayuso et al. (2014), Pourrahimian and Askari-Nasab (2014)); and quadratic programming (QP) (Diering (2012), Rubio and Diering (2004)). Khodayari and Pourrahimian (2015) presented a comprehensive review of mathematical programming applications in block-caving scheduling. They summarized authors' attempts to develop methodologies to optimize production scheduling in block-cave mining.

Song (1989) has studied the geological and geomechanical environments of Tong Kuang Yu mine in china. The author used displacement discontinuity method and undercut parameters to simulate the caving process. The result of simulation were analysed in order to obtain an optimal mining sequence. The objective function was to minimize the total mining cost. The drawback of this work was the run time particularly in long-term production scheduling.

Chanda (1990) used a combination of MILP formulation and simulation to model a short-term production scheduling problem at Chingola mine of Zambia. To solve the integer programming the authors applied the branch-and-bound algorithm and the decision variables were whether to draw or not to draw from a particular finger raise during a shift. The objective function was to minimize the fluctuation of average grade between shifts.

Guest et al. (2000) considered different constraints including geotechnical constraints, mining capacity, metallurgical limitations, economic parameters and geological limitations such as grade. In this case, maximizing tonnage would not necessarily result in maximum NPV. The objective function can be maximizing NPV, minimizing dilution or maximizing mine life (Rubio, 2002).

Rubio (2002) has defined two goals in his research: 1) maximization of NPV 2) optimization of the mine life in block caving. He considered geomechanical aspects, resource management, mining system and metallurgical parameters as the constraints of the proposed model. In this formulation he integrated the draw control factor and the angle of draw to compute production schedules with high performance in draw control.

Rahal et al. (2003) applied a MILP goal program at De Beers Kimberlite mine to obtain two objective functions: 1) minimizing deviation from the ideal draw profile, 2) achieving the production target. The constraints defied were: deviation from ideal practice, panel state, material flow conservation, production quality, material flow capacity, and production control.

A non-linear optimization method was introduced by Diering (2004) to minimize the deviation between a current draw profile and the goal of mine planner. The applied constraints were: geotechnical constraints, cave shape, draw point development sequence, draw point productivity, production block limits such as loader capacity and variable shut-off grade. He also stated that this algorithm can link the short-term plan with the long-term plan.

Rubio and Diering (2004) utilized mathematical programing to optimize the production strategy for the block-cave mine. Maximizing the NPV, minimizing the difference between actual height of draw and the target, and computing best height of draw by merging opportunity cost into PC-BC (GEOVIA) achieved using this algorithm. Various techniques such as direct iterative methods, linear programming, golden section search technique, and integer programming were performed for solving the proposed problem.

Rahal (2008) presented a goal programming model through two strategic goals of total monthly production tonnage and cave shape. Three main production control constraints were defined in the MILP: 1) the draw maturity rules that aims to have stability between drawpoint production and cave propagation rates, 2) minimum draw rate constraint to limit production and 3) relative draw rate (RDR) securing the consistent extraction across the cave.

Weintraub et al. (2008) developed an approach to aggregate the MIP original models in order to reduce the size of the problem. Two types of aggregation procedures based on clustering analysis were applied: a priori and a posteriori.

Smoljanovic et al. (2011) introduced a model for optimizing the sequence of opening of the drawpoints in a panel caving mine. They combined capacity and sequencing constraints in their approach. In this case, the objective function was not only maximizing NPV, but also robustness and constructability of the plans, subjected to other constraints such as geometallurgy, stocks and geometric constraints.

Parkinson (2012) formulated five models for sequence optimization in block cave mining in which three of them were integer programming models named: Basic, Malkin, and 2Cone. They helped to provide an appropriate input to the PC-BC program.

Epstein et al. (2012) presented a methodology for long-term production plans considering underground and open-pit deposits sharing multiple downstream processing plants. Solving the LP relaxation of formulation contributes significantly to decrease the size of the problem. Limitation on extraction rate, safety restrictions caused by rock spillage and instability of wall, and controls on pollutants were the constraints implemented by the authors.

Diering (2012) introduced QP applications instead of LP to the block cave production scheduling problems. The main reason for applying QP instead of LP is the solution space in which LP always seeks for the edges of the solution space even if there is a good average solution. Three types of constraints were defined: mandatory constraints, modifying constraints, and grade related constraints. Also the objective function in this case was the shape of the cave not maximizing some form of profit function.

Pourrahimian et al. (2012) solved two MILP formulations for the long-term production scheduling of block caving: first, at drawpoint level and then at cluster level in which drawpoints were aggregated into larger units. Maximizing NPV determined as the objective function with respect to operational constraints such as development rate, vertical mining rate, lateral mining rate, mining capacity, maximum number of active-drawpoints, and advancement direction. The authors implemented Fussy c-means clustering in order to create clusters and as a result, reduce the number of binary variables.

Pourrahimian et al. (2013) implemented a multi-step method for long-term production scheduling of block caving in order to reduce the size of the model. They presented three different formulations for three level of resolution; cluster level, drawpoint level, and drawpoint-and-slice level. The authors showed that both single-step method, in which each formulation was used independently; and multi-step method, in which the solution of each step was used to reduce the number of variables at the next step, can be utilized in the formulations.

Pourrahimian and Askari-Nasab (2014) presented a MILP formulation to determine the best height of draw (BHOD) in block-caving mine sequence optimization. Using a modified hierarchical algorithm they were able to aggregate the slices within each draw column into selective units. The optimal schedule is generated for the clustered slices.

2.3 Uncertainty-based Production Scheduling

Mining is different from most businesses because knowledge of the product is essentially based on estimates, which by their very nature include a degree of uncertainty. World commodity prices and exchange rates largely control potential changes in revenue, and consequently the size of the economic mineral inventory. Efficient mining is effectively about managing risk.

Production scheduling is a critical mechanism in the planning of a mine that deals with the effective management of a mine’s production and cash flows in the order of millions of dollars.

The importance of incorporating uncertainty and risk from the technical, geological, and mining sources in mine production schedules, particularly the possible in situ variability of pertinent ore-body grade and ore quality characteristics, is well appreciated.

Dimitrakopoulos (1998) classified the uncertainties of mining projects into three groups; (i) uncertainty of the ore-body model and related in-situ grade variability and material distribution, (ii) uncertainty of technical mining specifications such as slope constraints and excavation capacities, and (iii) uncertainty of economic issues including capital and operating costs, and commodity price. Among these uncertainties, the uncertainty related to the ore-body model and in-situ grade variability is the major one that affects the pre-feasibility and feasibility studies.

Although over the last decades, stochastic optimization methods have been developed to deal with geological uncertainty in open-pit mine design and life-of-mine production scheduling, similar efforts in underground mine planning and production scheduling are very limited.

Grade uncertainty can lead to significant differences between actual production and planning expectations and, as a result, the NPV of the project (Osanloo et al., 2008; Koushavand and Askari-Nasab, 2009). Various researchers have considered the effects of grade uncertainty in open-pit mines and introduced different methodologies to address those effects. Dowd (1994) presented a risk-based algorithm for surface mine planning. In their algorithm, for different variables such as commodity price, processing cost, mining cost, investment required, grade and tonnages, a predefined distribution function was implemented. Several types of schedules were generated for a number of realizations of the grades. This methodology produced various schedules that account for grade uncertainty. Ravenscroft (1992) and Koushavand and Askari-Nasab (2009) used simulated ore-bodies to show the influence of the grade uncertainty on production scheduling. Ramazan and Dimitrakopoulos (2004) used a MILP model to maximize the NPV for each realization. Then they calculated the probability of extraction of a block at each period. These probabilities were the input of a second stage of the optimization, which is necessary in order to generate one schedule at the end. Dimitrakopoulos and Ramazan (2008) presented an integer linear programming (ILP) model to generate optimal production schedules. This model considered multiple realizations of the block model and defined a penalty function that is the cost of deviation from the target production. This cost was calculated based on the geological risk discount rate which was the discounted unit cost of deviation from the target production. Linear programming was used to maximize a new function: NPV less penalty costs. Leite and Dimitrakopoulos (2007) implemented an approach that incorporated the geological uncertainty in the open-pit mine scheduling process. This new scheduling approach was based on a simulated annealing (SA) technique and stochastically simulated representations of the ore-body. Albor and Dimitrakopoulos (2009) developed a method which was based on scheduling with an SA algorithm and equally probable realizations of a mineral deposit. To generate production schedules, the equally probable realizations were utilized to minimize the possibility of deviations from production targets. Sabour and Dimitrakopoulos (2011) presented a procedure that combined geological uncertainty and operational flexibility in the design of open-pits. When designing an optimal production schedule and ultimate pit limit, Asad and Dimitrakopoulos (2013) considered both geological uncertainty and commodity prices with respect to the production capacity restrictions. Two-stage stochastic integer programming (SIP) was used in an optimization model to consider uncertainty (Ramazan and Dimitrakopoulos, 2013). Lamghari and Dimitrakopoulos (2012) also considered metal uncertainty in the open-pit production scheduling problem using a metaheuristic solution approach based on a Tabu search. Lamghari et al. (2013) proposed two variants of a variable neighborhood decent algorithm to solve the open-pit mine production scheduling problem under geological uncertainty. Maleki and Emery (2015) worked on the joint simulation of copper grade and rock type in a given deposit. To conduct the joint simulation, they implemented multi-Gaussian and pluri-Gaussian models in a combined form. They studied three main rock types with various grade distributions in which three auxiliary Gaussian random fields were considered. One of the rock types was used for copper grade simulation and the other two for rock-type simulation. Moreover, they looked at cross correlations between these Gaussian random fields before reproducing the dependence between copper grade and rock types.

Other than the aforementioned authors, few authors have examined geological uncertainty in underground mining. Grieco and Dimitrakopoulos (2007) implemented a new probabilistic mixed-integer programming model which optimized the stope designs in sublevel caving. Vargas et al. (2014) developed a tool that took geological uncertainty into account by using a set of conditional simulations of the mineral grades and defining the economic envelope in a massive underground mine. Montiel et al. (2015) incorporated geological uncertainty into their methodology that optimized mining operation factors such as blending, processing, and transportation. They used a simulated annealing algorithm to deal with uncertainty. Carpentier et al. (2016) introduced an optimization formulation that looked at a group of underground mines under geological uncertainty. Their formulation evaluated the project's influence on economic parameters including capital investments and operational costs. Alonso-Ayuso et al. (2014) considered two resources of uncertainty in their model: copper price along a given time horizon and grade. Uncertainty was described using a multi-stage scenario tree and then resulting stochastic model was transformed into MIP model.

2.4 Summary and Remarks

In chapter 2 of this dissertation, the relevant literature has been presented. Production scheduling specifically in block-cave mine defines the amount of material to be mined in each period, and in total, determines a strategic plan for the life of mine. It plays a key role in the operation's economics, and any deviation from the target plan can cause irrecoverable damages to the mine economics. The majority of current studies tended towards using simulation and heuristic methodologies to deal with the underground production scheduling optimization problems because of the complex nature of underground mines. The drawback of these methodologies is that they generate feasible solutions instead of optimal global schedules. The literature review showed that usage of mathematical programming and operation research algorithms in block-cave mine planning such as LP, MILP and QP has been extended recently and they became more powerful in generating exact optimal solutions.

On the other hand, grade uncertainty has profound impact on the NPV of the mining projects as it may induce large differences between the actual and expected production target. To deal with geological uncertainty several researchers provided various algorithms in open-pit mines but less in underground mining systems. Some of the implemented algorithms are: stochastic integer programming, risk-based algorithms, simulated annealing, and stochastic sequential simulation.

The major shortcomings of the current block-cave mine planning optimization can be summarized as: 1) limitations in considering uncertainties such as grade and commodity price; 2) limitations in solving the large-scale problems; 3) implementing fewer number of geotechnical constraints. These restrictions and deficiencies should be addressed in the new optimization tools to have viable mining projects.

CHAPTER 3

THEORETICAL FRAMEWORK: BLOCK-CAVE PRODUCTION SCHEDULING

Chapter 3 describes using stochastic sequential simulation approach to overcome the grade uncertainty through generating a set of realizations. A methodology is presented for finding the best level of extraction. To overcome the size of the problem and according to footprint dimensions, a method for creating big-block columns is reported. This chapter includes mixed-integer linear programming (MILP) formulations to obtain the optimal production schedule for the block-cave mine. The production scheduler aims to maximize the net present value (NPV) of the project while handling practical constraints including mining capacity, grade blending, number of new big-blocks, precedence, continuous extraction, draw rate, minimum required mining footprint, and reserves. At the end of this chapter, the MILP model's implementation is discussed. The numerical modeling of the MILP model is reviewed and how different parts of it such as objective function and constraints can be set in MATLAB programming environment is illustrated. IBM/CPLEX is implemented as solver for optimization.

3.1 Introduction

Uncertainty clearly affects the solution of long-term production planning. Several authors mentioned by Osanloo et al. (2008) who considered geological uncertainties such as grade in long-term mine planning. They reviewed that grade uncertainty can cause differences between designed and actual production, especially in early years of extraction.

Long-term mine planning purpose is to find the mining sequence, amount of material to be extracted, and input grade to the plant over the mine life. Production scheduling typically optimizes the company's objective with respect to some technical and operational constraints.

This chapter describes steps for block-cave mine planning in presence of grade uncertainty. It focuses on stochastic sequential simulation to deal with grade uncertainty, formulating, and developing the MILP model for production scheduling optimization. The best level to start extraction based on the maximum discounted profit will be found on the output of simulation. To reduce the size of problem and according to the distances between drawpoints, the blocks are placed into big-block columns along the advancement direction. The objective function of the MILP model is maximizing NPV, while controlling over: 1) mining capacity, 2) grade blending, 3) extraction rate, 4) continuous extraction, 5) number of new big-block columns, 6) mining precedence, and 7) total reserves. The production scheduler defines the number of active big-block columns in each period, the number of new opened big-block columns in each period, ore production tonnage and average grade in each period, and the sequence of extraction for each big- block column.

3.2 Steps of the proposed methodology

More accurate long-term production planning requires taking grade uncertainty into account. Production planning aims to generate a strategic plan throughout the mine life while honoring physical, economical, and environmental limitations. The steps that should be followed to generate a schedule for block-cave mine under grade uncertainty using the developed MILP model in this research include:

1. Creating an original block model from drillhole data and grid definition using GEOVIA GEMS software (6.7.1)
2. Implementing geological study to create several realizations (block models) based on drillholes data to consider grade uncertainty with the help of Geostatistical software library (GSLIB) (Deutsch and Journel, 1998).
3. Finding the best level to start the extraction through maximum discounted ore profit for all realizations, original and average-simulated block model.
4. Determining the best advancement direction at the obtained best level.
5. Determining the actual outline of the ore-body at the best level.
6. Creating big-block columns based on the minimum required mining footprint inside the outline of the ore-body at the best level.
7. Defining the input scheduling parameters.
8. Creating the objective function and the constraints of the MILP model.
9. Solving the problem to maximize NPV.
10. Discussing the results.

The ore-body is represented by a geological block model. Numerical data are used to represent each block's attributes, such as tonnage, density, grade, rock type, elevation, and profit data.

The first step is a geostatistical study to generate the realizations based on the drillholes data. At the next step, the best level of extraction is found. Finally, the optimal sequence of extraction is determined to maximize the NPV. Figure 3.1 shows the workflow that has to be followed to generate an optimal production schedule for the block-cave mine using MILP model under grade uncertainty in this research.

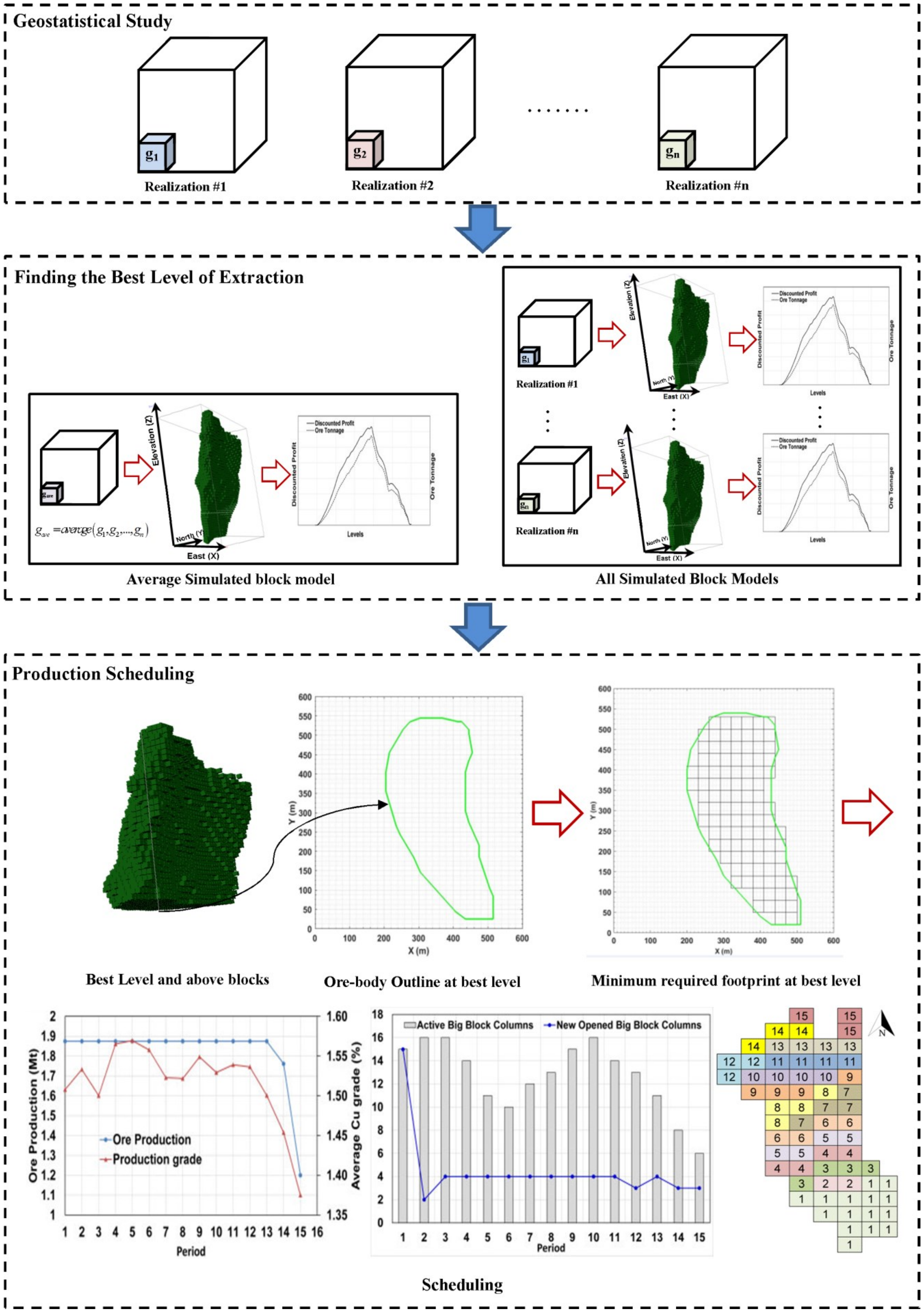


Figure 3.1. Required steps for the proposed methodology

3.2.1 Geostatistical Modeling

The first step for a geostatistical study is to define different rock types based on the drillhole data. In this study, which assumes a stationary domain within each rock type, the geostatistical modeling is performed for each rock type separately. The following steps are common for generating a geological model:

First, a declustering algorithm is used to get the representative distribution of each rock type to decrease the weight of clustered samples. Then, the correlation of the multivariate data is determined. To determine the principle directions of continuity, global kriging is performed using arbitrary variograms with a high range. Indicator kriging is used for rock type modeling, and simple kriging is used for grade modeling. The data is transformed to Gaussian units to remove the correlation between the variables in each rock type. The experimental variograms are calculated by using the determined directions of continuity in the previous step and a model is fitted to these variograms in different directions. An indicator variogram is used for rock type modeling and a traditional variogram is used for grade modeling. A rock type model is generated for the chosen grid definition by using a sequential indicator simulation algorithm (SIS). A grade model for each rock type is generated based on a Sequential Gaussian Simulation algorithm (SGS). Then, the data is back-transformed to original units. Finally, grade modeling is done within each rock type. The result will be several number of block models, the same as assumed number of realizations.

To investigate the effect of the grade uncertainty, the methodology of finding the best level of extraction and the MILP model should be applied on all the simulated block models.

3.2.2 Placement of Extraction Level

To find the best level of extraction, the ore tonnage and discounted profit are calculated for each level of each block model. The discounted profit of each ore block (Diering et al., 2008) and the total discounted profit of each level are calculated using equations (3.1) and (3.2).

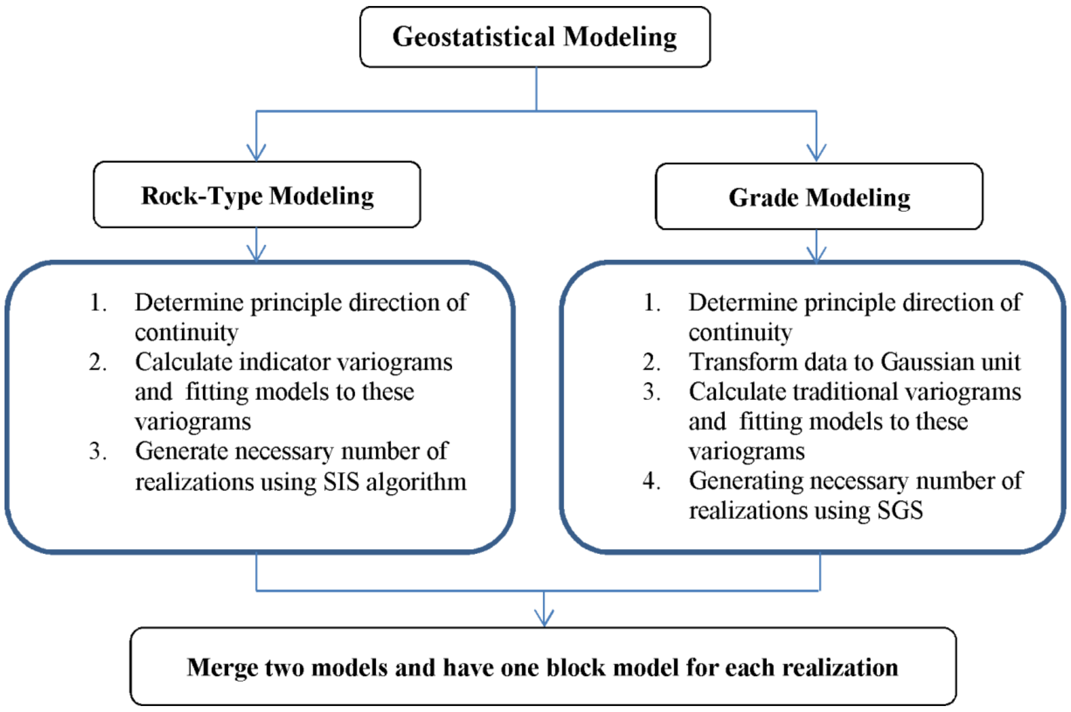

Geostatistical Modeling
Rock-Type Modeling
Grade Modeling
1. Determine principle direction of continuity
2. Calculate indicator variograms and fitting models to these variograms
3. Generate necessary number of realizations using SIS algorithm
1. Determine principle direction of continuity
2. Transform data to Gaussian unit
3. Calculate traditional variograms and fitting models to these variograms
4. Generating necessary number of realizations using SGS
Merge two models and have one block model for each realization

Figure 3.2. Summary of steps in geostatistical modeling

$$Dis\ \mathrm{P}_{blL} = \sum_{l=L}^{1} \frac{\mathrm{Pr}_{bll}}{(1+i)^{d/ER}}, \quad \forall bl \tag{3.1}$$

$$Dis\ \mathrm{P}_{L} = \sum_{bl=1}^{BL} Dis\ \mathrm{P}_{blL} \tag{3.2}$$

Where $Dis\ \mathrm{P}_{blL}$ is the discounted profit of ore block bl in level L; $Dis\ \mathrm{P}_{L}$ is the total discounted profit of level L , which is the summation of discounted profit of all the blocks in that level; Pr_{bll} is the profit (undiscounted) of ore block bl at level l; i is the discount rate; d is the distance between the center points of ore block bl in level L and the ore blocks above it; ER is the extraction rate per period; BL is the total number of ore blocks in level L. The profit of each ore block is calculated using the following equations:

$$T_R = g \times Ton \times R \times (P - S_C) \tag{3.3}$$

$$T_C = Ton \times (M_C + P_C) \tag{3.4}$$

$$P = T_R - T_C \tag{3.5}$$

Where T_R is the total revenue; R is the processing plant recovery; P is the price per ton of the product; S_C is the selling cost per ton of material; g is the element grade; T_C is the total cost; P_C is the processing plant cost and M_C is the cost of mining per ton of material which is assumed to be a constant number and development cost is considered in this cost. The following example (Figure 3.3) clearly shows how to calculate the discounted profit of a block at a given level. Two blocks are assumed to be in each level. Equations (3.6) and (3.7) show the calculation of $Dis\,\mathrm{P}_{14}$ and $Dis\,\mathrm{P}_{24}$, which are the discounted profit of $bl=1$ and $bl=2$ at level 4, respectively. ER is a constant number representing extraction rate which its unit is meter per period. Equation (3.8) shows the total discounted profit of blocks 1 and 2 at level4.

$$Dis\,\mathrm{P}_{14}=\frac{\mathrm{Pr}_{14}}{(1+i)^{d_0/ER}}+\frac{\mathrm{Pr}_{13}}{(1+i)^{d_1/ER}}+\frac{\mathrm{Pr}_{12}}{(1+i)^{d_2/ER}}+\frac{\mathrm{Pr}_{11}}{(1+i)^{d_3/ER}} \tag{3.6}$$

$$Dis\,\mathrm{P}_{24}=\frac{\mathrm{Pr}_{24}}{(1+i)^{d_0/ER}}+\frac{\mathrm{Pr}_{23}}{(1+i)^{d_1/ER}}+\frac{\mathrm{Pr}_{22}}{(1+i)^{d_2/ER}}+\frac{\mathrm{Pr}_{21}}{(1+i)^{d_3/ER}} \tag{3.7}$$

$$Dis\,\mathrm{P}_{4}=Dis\,\mathrm{P}_{14}+Dis\,\mathrm{P}_{24} \tag{3.8}$$

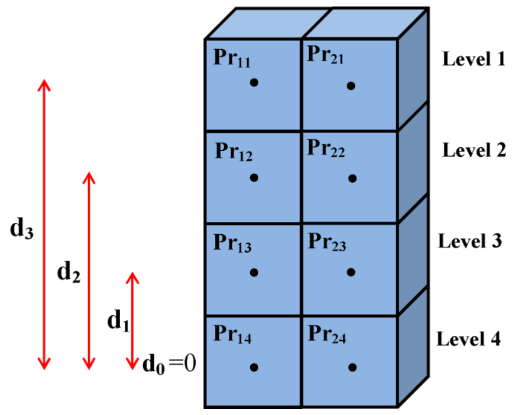


Figure 3.3. Schematic example of calculating discounted profit of ore block at a given level

At the next step, the tonnage-profit curve is plotted and the level with the highest profit is selected for starting the extraction (Figure 3.4). The best level of extraction is found for the created average-simulated block model. When the method is applied on each realization, the best elevation for each block model is determined and the most frequent level is specified. According to the results of both procedures (best level of all realizations and best level of average-simulated block model) the best level will be selected.

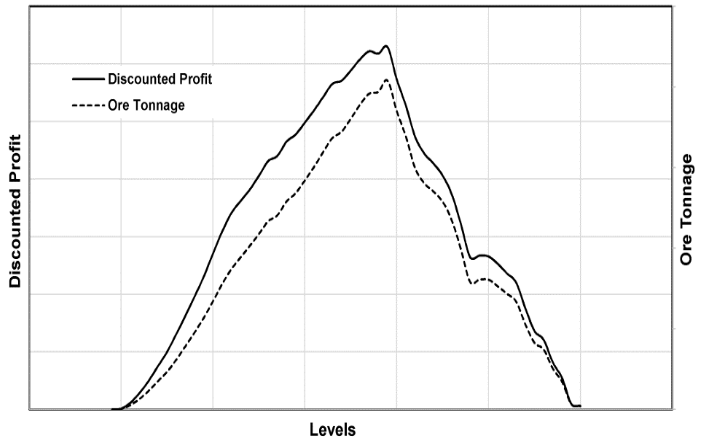
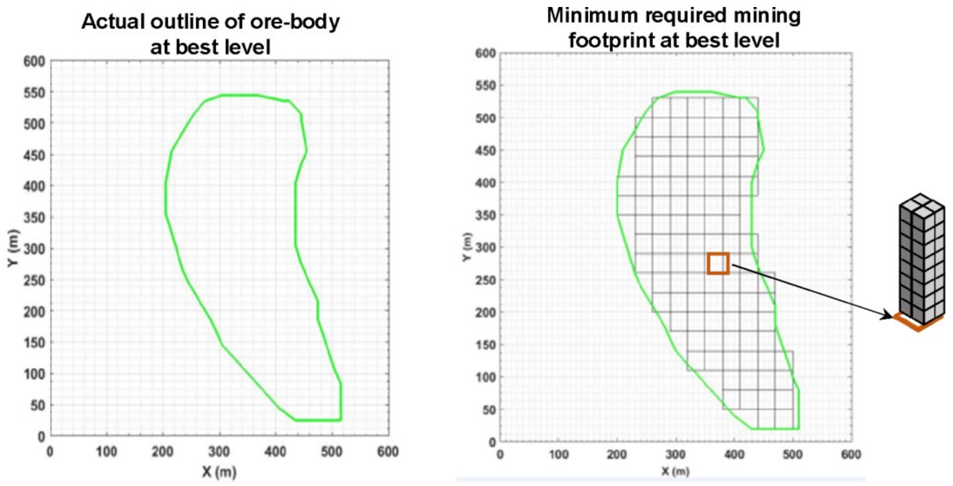


Figure 3.4. Schematic view of finding best level of extraction methodology

3.2.3 Production Scheduling

After determining the best elevation, the interior of the ore-body outline is divided into rectangles based on the minimum required mining footprint (Figure 3.5). The minimum mining footprint (plan view) represents the minimum sized shape that will induce and sustain caving. This is similar to the hydraulic radius in a caving operation. Then all blocks inside of the rectangle and above that, create a big-block. In the next step, the sequence of extraction of the big-blocks is optimized.


Actual outline of ore-body at best level
Minimum required mining footprint at best level
Y (m)
X (m)
Figure 3.5. Schematic view of production scheduling methodology

3.3 Mathematical Programming Formulation

In this part, the MILP formulations are presented. The purpose is to maximize the NPV of the mining operation, while controlling mining capacity, grade blending, extraction rate, continuous extraction, binary, number of new big blocks, precedence, and reserve. The production scheduler defines the extraction rate from each big-block, the number of active and new big-block in each period, and the sequence of extraction from each big-block.

3.3.1 Model Assumptions

The following assumptions are used in the MILP formulations:

1. Numerical data are used to represent ore-body attributes in each block, such as tonnage, density, grade of elements, coordinates, and profit data.
2. There is no material mixing between blocks as a function of draw. The source model is assumed to be static with time.
3. The big-block columns are created according to the minimum required mining footprint.
4. The portion scheduled to be extracted from each big-block is assumed to be taken from all the small blocks inside of the big-block.

3.3.2 Objective Function

Various strategic targets could be considered by different companies such as cost minimization or reserve maximization. Usually, the aim is to maximize the mining operation’s NPV with respect to the existing technical, physical, and environmental constraints.

In this research, the objective function of the MILP formulation is to maximize the mining operation’s NPV which depends on the value of the big-block columns.

3.3.3 Constraints

The following set of constraints is included in the formulation:

3.3.3.1 Mining capacity

The desired production target can be achieved by this constraint. It ensures that the total tonnage of material extracted from big-blocks in each period is within the acceptable range. The constraints are controlled by the continuous variables.

3.3.3.2 Grade blending

The desired grade can be achieved by this constraint. It ensures that the production's average grade is in the acceptable range in each period.

3.3.3.3 Big-block extraction rate and continuous extraction

This constraint ensures that the extraction rate from each big-block per period is between the defined maximum and minimum extraction rates.

3.3.3.4 Binary

This constraint ensures that if the extraction of a big-block is started, its binary variable takes the value of 1 and stays 1 till the end of the mine life, otherwise it takes value of 0.

3.3.3.5 Number of new big-blocks

These constraints ensure that the number of new big-blocks which are opened in each period are in an acceptable range.

3.3.3.6 Mining precedence

These constraints ensure that all the predecessor big-blocks of a given big-block have been started prior to extracting this big-block.

To apply this constraint, first the adjacent big-blocks of each big-block are determined and then an advancement direction is defined. Afterwards, a perpendicular line to the advancement direction is imagined at the center point of the considered big-block. Then a point should be found on the perpendicular line using equation (3.9). The coordinate of this point is (X_{new} , Y_{new}).

$$Y_{new} - y_{bbl} = -\frac{1}{m}(X_{new} - x_{bbl}) \tag{3.9}$$

Where m is the slope of the advancement direction; y_{bbl} and x_{bbl} are the coordinates of the considered big-block in the extraction level; X_{new} is an arbitrary coordinate and as a result, Y_{new} is calculated by equation (3.9). Then, using equation (3.10), the value of D is calculated for each adjacent big-block.

$$D = (x_{adj} - x_{bbl})(Y_{new} - y_{bbl}) - (y_{adj} - y_{bbl})(X_{new} - x_{bbl}) \tag{3.10}$$

Where x_{adj} and y_{adj} are the coordinates of the adjacent big-blocks of each big-block. By calculating D, if the mining direction points to the direction that y increases, big-blocks with $D < 0$ are below the perpendicular line and considered as the predecessors of a given big-block and if not, big-blocks with $D > 0$ are considered as predecessors of the specified big-block.

The following example contributes significantly to a clear understanding of the methodology used for precedence constraint.

Figure 3.6 shows how to select the predecessors for different advancement directions. A big-block (red block) is considered and its adjacent big-blocks are BL1-BL8. In Figure 3.6, it assumes that the blue arrow shows the advancement direction and the orange line is the imaginary perpendicular line at the center of the considered big-block (red block). The related calculation has been summarized in Table 3.1. According to Figure 3.6a, the advancement direction is from Southwest to Northeast (SW to NE) which means y is increasing; therefore the extraction of the big-blocks with the negative value of *D* should be started before the considered block (see Table 3.1). Figure 3.6b and c are examples of positive values with similar directions. Big blocks 4, 6, 7, and 8 and 6, 7, and 8 are predecessor big-blocks for red block in b and c respectively, as they have positive *D*. Also, in Figure 3.6d, as the advancement direction is from East to West (E to W) and *D* should be negative, big-blocks 3, 5, and 8 are chosen as the predecessors.

Figure 3.7 shows the summary and schematic view of determining the sign of D for different advancement directions.

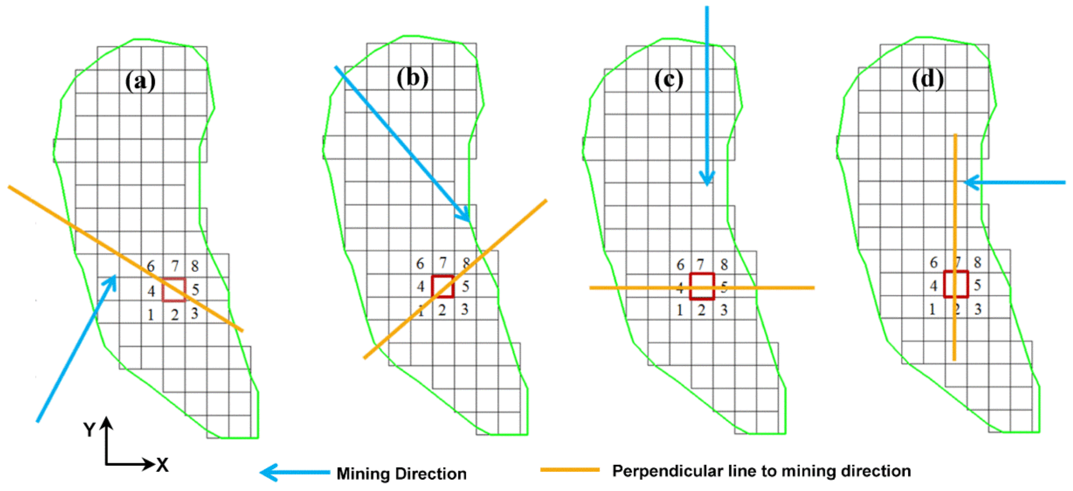


Figure 3.6. Schematic examples of methodology used in precedence constraint

Table 3.1. Example of calculation to find the predecessors of a big-block in the considered advancement direction in Figure 3.6a

Direction: SW → NE
Slope of advancement direction: m = 1.8

Considered block's coordinates: (395,215)
$X_{new} = 200$

Adjacent blocks	1	2	3	4	5	6	7	8
Coordinates	(365,185)	(395,185)	(425,185)	(365,215)	(425,215)	(365,245)	(395,245)	(425,245)
D (Eq.(3.10))	< 0	< 0	< 0	< 0	> 0	> 0	> 0	> 0
predecessor	Yes	Yes	Yes	Yes	No	No	No	No

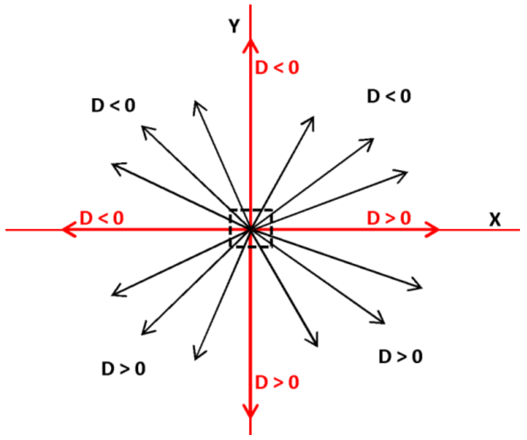


Figure 3.7. Schematic presentation of determining sign of D based on various advancement directions

3.3.3.7 Reserves

This constraint ensures that the extracted amount of material of a big-block over the scheduling periods sum to one, which means all material inside of the big-block, should be extracted.

3.3.4 MILP Formulations

The profit of each big-block is equal to the summation of the profit of all the small blocks within the big-block column. To solve the problem, three sets of decision variables are employed: one continuous variable and two binary variables. The continuous variable indicates the portion of extraction from each big-block in each period. One of the binary variables controls the precedence of the extraction of big-blocks and the other one is used for activating either of two constraints. The objective function tries to mine the big-blocks with higher profit earlier than the others according to the defined constraints. The notation used to formulate the problem is divided into indices, sets, decision variables, and parameters.

Indices

$t \in \{1,....,T\}$	Index for scheduling periods.
$bl \in \{1,...,\mathrm{BL}\}$	Index for small blocks.
$bbl \in \{1,...,\mathrm{BBL}\}$	Index for the ore big-blocks

Set

S^{bbl}	For each big-block, bbl , there is a set S^{bbl} , which define the predecessor big-blocks that must be started prior to extracting the big-block bbl .

Decision variables

$B_{bbl,t} \in \{0,1\}$	Binary variable controlling the precedence of the extraction of big-blocks. It is equal to one if the extraction of big-block bbl has started by or in period *t*; otherwise it is zero.
$\mathrm{x}_{bbl,t} \in [0,1]$	Continuous variable, representing the portion of big-block bbl to be extracted in period *t*.
$y_{bbl,t} \in \{0,1\}$	Binary variable used for activating either of two constraints.

Parameters

$\mathrm{Pr}ofit_{bbl}$	Profit of each big-block which is equal to the summation of

	the small ore blocks' profit within that big-block.
Ton_{bbl}	Tonnage of each big-block.
$MCL_t\,(Mt)$	Lower bound of mining capacity in period *t*.
$MCU_t\,(Mt)$	Upper bound of mining capacity in period *t*.
g_{bbl}	Average grade of the element to be studied in big-block *bbl*
$GL_t\,(\%)$	Lower bound of acceptable average head grade of considered element in period *t*.
$GU_t\,(\%)$	Upper bound of acceptable average head grade of considered element in period *t*.
$ExtU_t\,(Mt)$	Maximum possible extraction rate from each big-block in period *t*.
$ExtL_t\,(Mt)$	Minimum possible extraction rate from each big-block in period *t*.
L	Arbitrary big number.
T	Maximum number of scheduling periods.
BBL	Number of ore big-blocks in the model.
n	Number of predecessor big-blocks of big-block *bbl*
$\overline{N}_{NBBL,t}$	Upper bound for the number of new big-blocks, the extraction from which can start in period t .
$\underline{N}_{NBBL,t}$	Lower bound for the number of new big-blocks, the extraction from which can start in period t .

Objective function:

$$Max \quad \sum_{t=1}^{T}\sum_{bbl=1}^{BBL}\left[\frac{Profit_{bbl}}{(1+i)^t}\right]x_{bbl,t} \tag{3.11}$$

Constraints:

$$MCL_t \le \sum_{bbl=1}^{BBL} Ton_{bbl}\times x_{bbl,t} \le MCU_t\,, \quad \forall t\in\{1,....,T\} \tag{3.12}$$

$$GL_t \le \frac{\sum_{bbl=1}^{BBL} g_{bbl}\times Ton_{bbl}\times x_{bbl,t}}{\sum_{bbl=1}^{BBL} Ton_{bbl}\times x_{bbl,t}} \le GU_t\,, \quad \forall t\in\{1,...,T\} \tag{3.13}$$

$$Ton_{bbl}\times x_{bbl,t} \le ExtU_{bbl,t}\,, \quad \forall bbl\in\{1,...,BBL\},\, t\in\{1,...,T\} \tag{3.14}$$

$$(ExtL_{bbl,t} \times B_{bbl,t}) - (Ton_{bbl} \times x_{bbl,t}) \leq L \times y_{bbl,t}, \quad \forall bbl \in \{1,...,BBL\}, t \in \{1,...,T\} \tag{3.15}$$

$$\sum_{t'=1}^{t} x_{bbl,t} \geq y_{bbl,t}, \quad \forall bbl \in \{1,...,BBL\}, t \in \{1,...,T\} \tag{3.16}$$

$$\mathrm{x}_{bbl,t} \leq B_{bbl,t}, \quad \forall bbl \in \{1,...,BBL\}, t \in \{1,...,T\} \tag{3.17}$$

$$B_{bbl,t} - B_{bbl,t+1} \leq 0, \quad \forall bbl \in \{1,...,BBL\}, t \in \{1,...,T\} \tag{3.18}$$

$$\underline{N}_{NBBL,1} \leq \sum_{bbl=1}^{BBL} B_{bbl,t} \leq \overline{N}_{NBBL,1} \ , \quad t = 1 \tag{3.19}$$

$$\underline{N}_{NBBL,t} \leq \sum_{bbl=1}^{BBL} B_{bbl,t} - \sum_{bbl=1}^{BBL} B_{bbl,t-1} \leq \overline{N}_{NBBL,t} \quad , \ \forall t \in \{2,...,T\} \tag{3.20}$$

$$n \times B_{bbl,t} \leq \sum_{k=0}^{n} B_{S^{bbl}(k),t} \quad , \ \forall bbl \in \{1,...,BBL\}, t \in \{1,...,T\} \tag{3.21}$$

$$\sum_{t=1}^{T} x_{bbl,t} = 1, \quad \forall bbl \in \{1,...,BBL\} \tag{3.22}$$

The objective function, equation (3.11), is composed of the big-blocks' profit value, discount rate, and a continuous decision variable that indicates the portion of a big-block, which is extracted in each period. The most profitable big-blocks will be chosen to be part of the production in order to maximize the NPV.

The constraints are presented by equations (3.12) to (3.21). Equation (3.12) represents the mining capacity. This constraint is controlled by the continuous variable $\mathrm{x}_{bbl,t}$. There is one constraint per period.

Equation (3.13) ensures that the production's average grade is in the acceptable range. This constraint is controlled by the continuous variable $\mathrm{x}_{bbl,t}$. There is one constraint per period.

Equations (3.14), (3.15), and (3.16) are related to extraction rate and continuous extraction constraints. Equation (3.14) ensures that the extraction rate from each big-block per period does not exceed the maximum extraction rate. $y_{bbl,t}$ in equations (3.15) and (3.16) is a binary variable which is used to activate either equation (3.15) or (3.16). Whenever equation (3.15) is active, it ensures that minimum extraction rate from each big-block per period is extracted. If the remaining tonnage of a big-block is less than the minimum extraction rate, equation (3.16) will be activated and forces that big-block to be extracted as much as the remaining tonnage which results in continuous extraction from each big-block.

Equations (3.17) and (3.18) are defined for binary constraints. Equation (3.17) ensures that if the extraction of a big-block is started its binary variable should be one. Also equation (3.18) controls the fact that if the extraction of a big block in period t has been started ($B_{bbl,t} = 1$), the related binary variable should be kept one till end of the mine life. Both equations (3.16) and (3.18) contribute to the continuity of the extraction. The results of these constraints will be used for the precedence constraint for which the maximum number of active big-blocks is needed.

Equations (3.19) and (3.20) ensure that the number of new big-blocks in each period should be in an acceptable range. It is obvious that the number of new big-blocks in period one is more than other periods; therefore equation (3.19) is applied to period one and equation (3.20) is applied from period two to the end of the mine life.

Equation (3.21) ensures that all the predecessor big-blocks of a given big-block *bbl* have been started prior to extracting the big-block.

In this formulation, all material inside of the big blocks should be extracted. This is controlled by equation (3.22).

3.4 MILP Formulation Implementation

The application of numerical models is discussed in this part through instructions and methods. Two important elements in formulation and application of the MILP model are: (1) objective function, and (2) constraints (Pourrahimian, 2013).

Selecting the appropriate numerical platform is the first step in each MILP formulation development. In this research, MATLAB (Math Works Inc., 2015) was used as the numerical modeling platform and IBM/CPLEX (IBM, 2015) as the solver to optimize the production scheduling. MATLAB is a powerful language for numerical computation and programming. It enables the users to reach a solution faster than traditional programming languages, such as C/C++ or spreadsheets. CPLEX is useful for solving the large-scale mixed-integer linear and quadratic programming. The package contains simplex and barrier solvers (Pourrahimian, 2013).

A generalized structure of MILP problem which is used by IBM/CPLEX is stablished and is implemented as a basis for numerical modeling. The main components of the MILP model are built in MATLAB to be transmitted to IBM/CPLEX for optimization (Pourrahimian, 2013).

3.4.1 Numerical Modeling

MILP formulation for mine optimization usually results in large-scale problems. CPLEX (IBM, 2015) as a commercial optimization solver is capable of tackling this issue. It uses a branch-and-cut algorithm that is a method of combinatorial optimization (Horst and Hoang, 1996; Wolsey, 1998).

The proposed MILP model is developed in MATLAB (Math Works Inc., 2015), and solved in the IBM ILOG CPLEX environment (IBM, 2015). A branch-and-bound algorithm is used to solve the MILP model, assuring an optimal solution if the algorithm is run to completion. Gap tolerance (EPGAP) is used as an optimization termination criterion in CPLEX. This is an absolute tolerance between the gap of the best integer objective and the objective of the remained best node.

3.4.2 General Formulation

Equations are the general structure of a MILP problem which is used by IBM/CPLEX.

$$\min \quad f^{'}.\mathrm{x} \tag{3.23}$$

Subject to:

$$Aineq\,.x \le bineq \tag{3.24}$$

$$Aeq\,.x = beq \tag{3.25}$$

$$lb \le x \le ub \tag{3.26}$$

Where

- f is a column vector for linear objective function .
- x is the decision variable of the MILP model (a column vector).
- *Aineq* is a matrix for linear inequality constraints.
- *bineq* is a column vector for linear inequality constraints (boundary vector).
- *Aeq* is a matrix for linear equality constraints.
- *beq* is a column vector for linear equality constraints (boundary vector).
- *lb* and *ub* column vector of lower and upper bounds.

3.4.2.1 The MILP Objective Function

The objective function of this block-cave production scheduling problem as sated by equation (3.11) is to maximize NPV. The general form of the objective function in CPLEX according to equation (3.23) is minimization. Therefore, the objective function coefficient vector for equation (3.11) should be multiplied by a negative sign and consequently, the objective function will change to minimizing the –NPV of the mining operation. The objective function of this model as presented in equation (3.11), has a coefficient vector, f . Table 3.2 shows the size and values of this vector. In the second column of Table 3.2, $D_{\mathrm{Pr}ofit}$ is a $(N \times T) \times 1$ vector which is the big-blocks' discounted profit values shown by equation (3.11). Also, 0 is a $(2 \times N \times T) \times 1$ vector with all elements equal to zero; N is the number of the big-blocks and T is the number of scheduling periods. The matrix vertical concatenation operator, ';' is used for notation simplification that creates a matrix by concatenating them along the vertical dimension of the matrix.

Table 3.2. Size of the coefficient vector of objective function

Size of the coefficient vector	Coefficient vector
$(3 \times N \times T) \times 1$	$\left[D_{\mathrm{Pr}ofit}; 0\right]$

Different units of the coefficient matrices of the objective function and constraints lead to transforming them into unitless vectors and matrices. Normalizing the vectors and matrices is performed for this issue by dividing them by norm(s) of their multiplier vector(s). Table 3.3 shows the size of the vector for decision variables and its order. X is a $(N \times T) \times 1$ vector holding the continuous decision variables controlling the extraction portion of each big block in each period: $\mathrm{x}_{bbl,t} \in [0,1]$. B is a $(N \times T) \times 1$ vector holding the binary decision variables controlling the precedence of the extraction of each big-block in each period: $B_{bbl,t} \in \{0,1\}$. Y is a $(N \times T) \times 1$ vector holding the binary decision variables controlling activating of either of two constraints: $y_{bbl,t} \in \{0,1\}$.

Table 3.3. Size of the decision variables' vector and its order

Size of the decision variable vector	Structure
$(3 \times N \times T) \times 1$	$[\mathrm{X}; B; Y]$

3.4.2.2 The Constraints of the MILP Models

The constraints of the MILP model is presented by equations (3.12) to (3.22). A numerical model for these equality and inequality constraints has been developed in the following section. In this formulation the binary variable defines whether the extraction of a big-block has been started by/in each period or not. Figure 3.8 shows the structure of the constraints' coefficient matrix. Table 3.4 demonstrates the number of rows for each constraint.

The constraints' coefficient matrix itself is divided into various parts based on the decision variables. Figure 3.9 shows these parts in the formulation. The number of the decision variables determines the number of parts. In this research there are three types of decision variables and therefore the number of the parts are three, belonging to the related variables. Also each part is sub-divided into smaller parts according to the number of scheduling periods. Figure 3.10 shows the structure of each variable in the constraints' coefficient matrix and decision variable vector. The number of columns in each period is equal to the maximum number of big-blocks for all variables X, B, and Y.

Table 3.4. Number of rows in constraint' coefficient matrix

Constraints	Number of rows in coefficient matrix
Mining capacity	$2 \times T$
Grade blending	$2 \times T$
Draw rate	$3 \times N \times T$
Binary	$N \times T + N \times (T-1)$
Precedence	$N \times T$
Number of new big blocks	$2 \times T$
Reserves	N

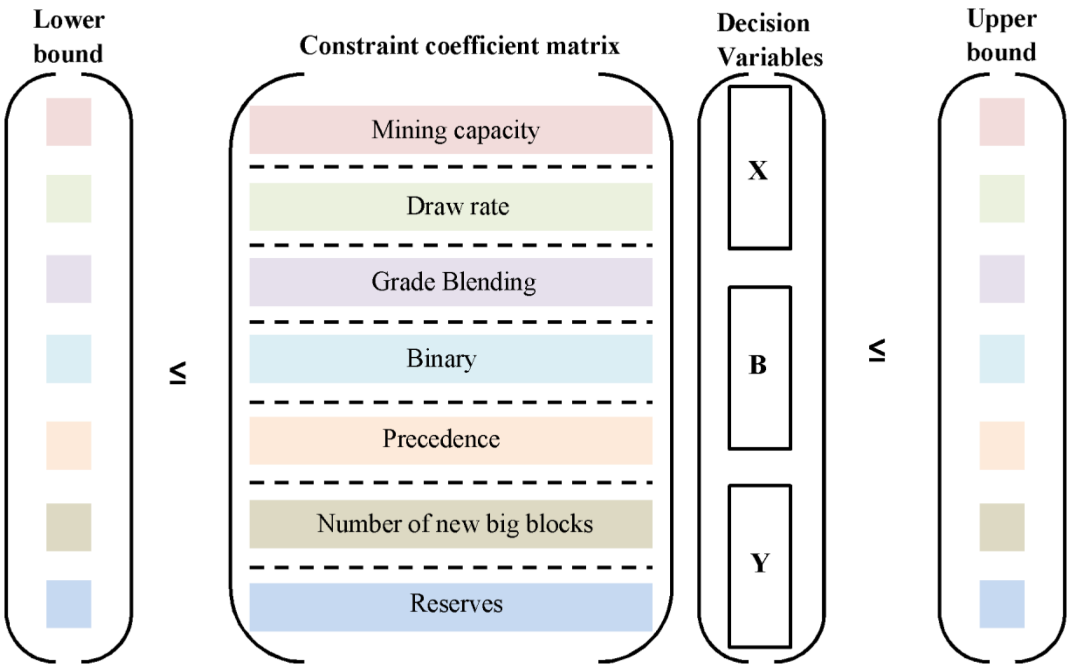


Figure 3.8. Order of constraints in the constraint' coefficient matrix in the formulation

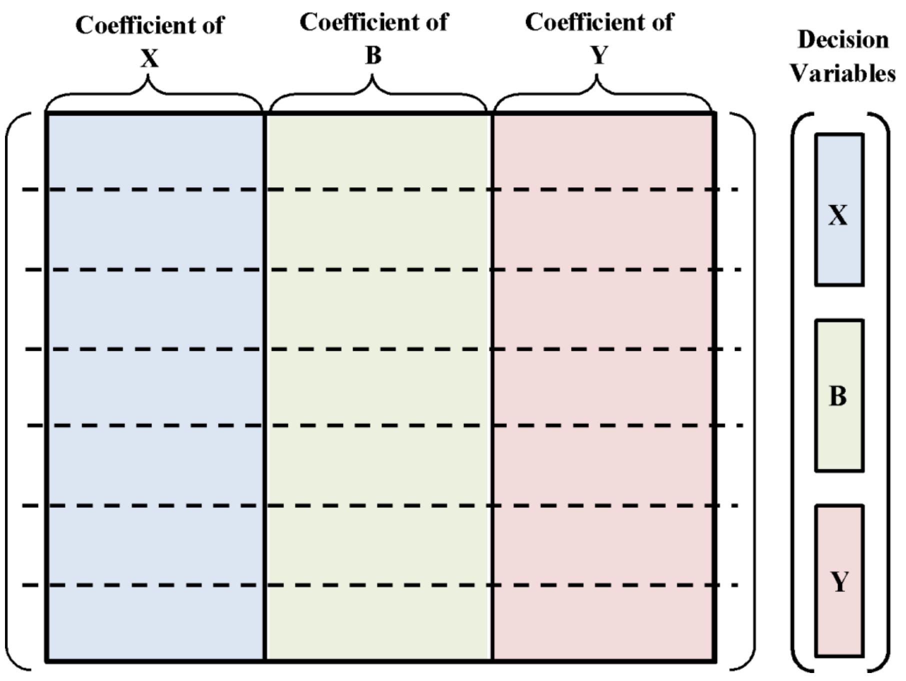


Figure 3.9. Division of the constraints' coefficient matrix based on the decision variables

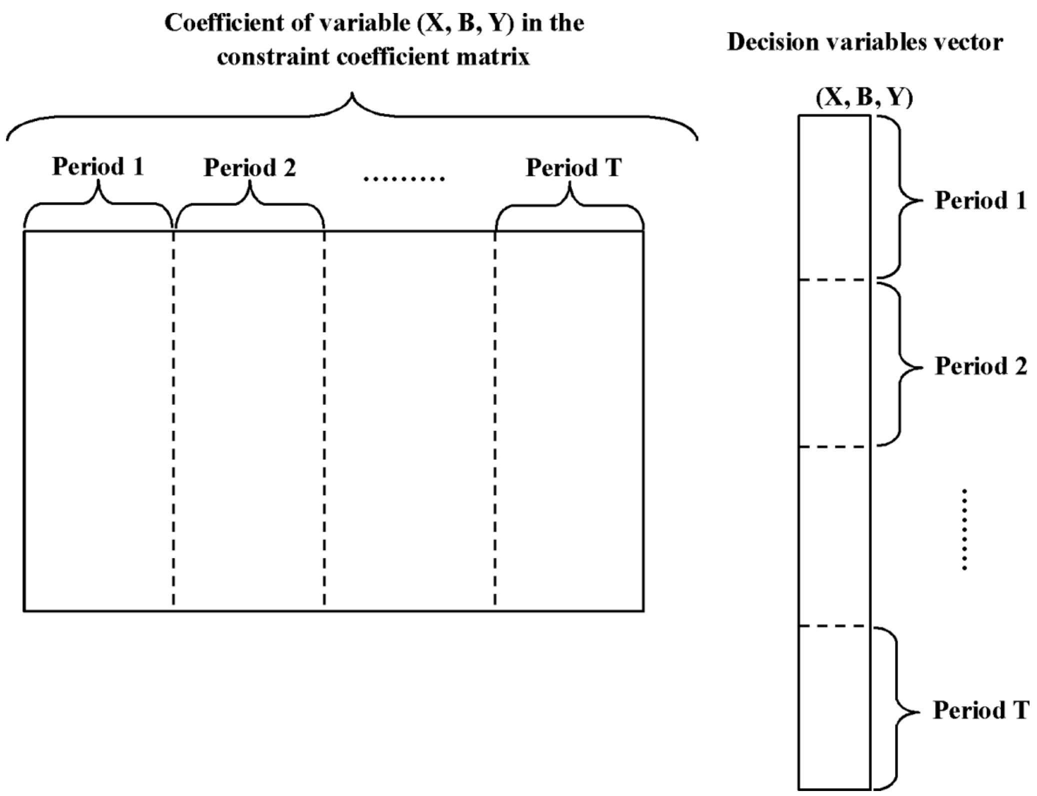

Coefficient of variable (X, B, Y) in the constraint coefficient matrix
Decision variables vector
(X, B, Y)
Period 1
Period 2
....
Period T
Period 1
Period 2
Period T

Figure 3.10. The structure of each variable in the constraints coefficient matrix and decision variables vector

The detail description of the structure for all the constraints is explained in the following section. For saving the memory space and decrease the size of the constraints' coefficient matrix, MATLAB's sparse function is used through squeezing out any zero elements.

Mining Capacity

$2T$ rows of the constraint coefficient matrix is belonged to this constraint. Equation (3.12) itself is divided into the following equations:

$$\left(\sum_{bbl=1}^{BBL} Ton_{bbl} \times x_{bbl,t}\right) - MCU_t \le 0, \quad \forall t \in \{1,....,T\} \tag{3.27}$$

$$MCL_t - \left(\sum_{bbl=1}^{BBL} Ton_{bbl} \times x_{bbl,t}\right) \le 0, \quad \forall t \in \{1,....,T\} \tag{3.28}$$

The first T rows form the upper bound of the mining capacity equation (equation(3.27)) and the next T rows is belonged to the lower bound of this equation (equation(3.28)). Equation (3.29) demonstrates the structure of this constraint. $\boldsymbol{X_{Mc}}$, $\boldsymbol{B_{Mc}}$, and $\boldsymbol{Y_{Mc}}$ are $2T \times (N \times T)$ matrices in the formulations. All the elements of B_{Mc} and Y_{Mc} matrices are equal to zero.

Equation (3.30) shows the structure of the decision variable $\boldsymbol{X_{Mc}}$ in the coefficient matrix of the mining capacity constraint. In this structure, $\boldsymbol{K^t_{ton}}$ and $\boldsymbol{K^t_{-ton}}$ are a $1 \times N$ row vector containing the tonnage and -tonnage of the big-blocks in period t . $\mathbf{0}$ is a $1 \times N$ vector in which all the elements of this vector is equal to zero. In this matrix, each row is related to each period and the corresponding equations (3.27) or (3.28). Therefore, in the first T rows, $\boldsymbol{K^t_{ton}}$ is placed in period t , and $\boldsymbol{K^t_{-ton}}$ is related to the second T rows belonging to period t . For example, $\boldsymbol{K^1_{ton}}$ should be placed in row 1 at period 1 and $\boldsymbol{K^1_{-ton}}$ should be placed in row $T+1$ at period 1.

$$\left[Lb_{Mc}\right] \le \left[X_{Mc} \quad B_{Mc} \quad Y_{Mc}\right] \le \left[Ub_{Mc}\right] \tag{3.29}$$

$$\boldsymbol{X}_{Mc}=\begin{bmatrix} \boldsymbol{K}_{ton}^{1} & \mathbf{0} & . & \mathbf{0} \\ \mathbf{0} & \boldsymbol{K}_{ton}^{2} & . & \mathbf{0} \\ . & . & . & \mathbf{0} \\ . & . & . & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & . & \boldsymbol{K}_{ton}^{t} \\ \boldsymbol{K}_{-ton}^{1} & \mathbf{0} & . & \mathbf{0} \\ . & \boldsymbol{K}_{-ton}^{2} & . & \mathbf{0} \\ . & . & . & \mathbf{0} \\ . & . & . & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & . & \boldsymbol{K}_{-ton}^{t} \end{bmatrix} \tag{3.30}$$

Draw rate

$3\times N\times T$ rows of the constraints' coefficient matrix is belonged to this constraint. Equations (3.14), (3.15), and (3.16) control this constraint. Each of the equations forms $N\times T$ rows of the draw rate constraint coefficient matrix. Equation (3.31) illustrates the structure of this constraint. Indices 1, 2, and 3 are associated with equations (3.14), (3.15), and (3.16), respectively. $\boldsymbol{X}_{DR_1}$, $\boldsymbol{X}_{DR_2}$, $\boldsymbol{X}_{DR_3}$, $\boldsymbol{B}_{DR_1}$, $\boldsymbol{B}_{DR_2}$, $\boldsymbol{B}_{DR_3}$, $\boldsymbol{Y}_{DR_1}$, $\boldsymbol{Y}_{DR_2}$, and $\boldsymbol{Y}_{DR_3}$ are $(N\times T)\times(N\times T)$ matrices in the formulations. All the elements of matrices $\boldsymbol{B}_{DR_1}$, $\boldsymbol{B}_{DR_3}$, and $\boldsymbol{Y}_{DR_1}$ are equal to zero.

The following example clearly shows the structure of the coefficient matrix of this constraint for each of decision variables.

$N=3$ and $T=2$ are the assumptions of this example. In the following matrices, notation ton_{bbl}^{t} contains the tonnage of big-block bbl in period t, and notation $ExtL_{bbl}^{t}$ contains the minimum possible extraction rate from each big-block bbl in each period t. Also, L is an arbitrary big number. For instance, in equation (3.35) , $ExtL_{2}^{1}$ stands for the minimum extraction rate of big-block 2 in period 1.

$$\begin{bmatrix} Lb_{DR_1} \\ Lb_{DR_2} \\ Lb_{DR_3} \end{bmatrix} \le \begin{bmatrix} X_{DR_1} & B_{DR_1} & Y_{DR_1} \\ X_{DR_2} & B_{DR_2} & Y_{DR_2} \\ X_{DR_3} & B_{DR_3} & Y_{DR_3} \end{bmatrix} \le \begin{bmatrix} Ub_{DR_1} \\ Ub_{DR_2} \\ Ub_{DR_3} \end{bmatrix} \tag{3.31}$$

$$X_{DR_1} = \begin{bmatrix} ton_1^1 & 0 & 0 & 0 & 0 & 0 \\ 0 & ton_2^1 & 0 & 0 & 0 & 0 \\ 0 & 0 & ton_3^1 & 0 & 0 & 0 \\ 0 & 0 & 0 & ton_1^2 & 0 & 0 \\ 0 & 0 & 0 & 0 & ton_2^2 & 0 \\ 0 & 0 & 0 & 0 & 0 & ton_3^2 \end{bmatrix} \tag{3.32}$$

$$X_{DR_2} = \begin{bmatrix} -ton_1^1 & 0 & 0 & 0 & 0 & 0 \\ 0 & -ton_2^1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -ton_3^1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -ton_1^2 & 0 & 0 \\ 0 & 0 & 0 & 0 & -ton_2^2 & 0 \\ 0 & 0 & 0 & 0 & 0 & -ton_3^2 \end{bmatrix} \tag{3.33}$$

$$X_{DR_3} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 \end{bmatrix} \tag{3.34}$$

$$B_{DR_2} = \begin{bmatrix} ExtL_1^1 & 0 & 0 & 0 & 0 & 0 \\ 0 & ExtL_2^1 & 0 & 0 & 0 & 0 \\ 0 & 0 & ExtL_3^1 & 0 & 0 & 0 \\ 0 & 0 & 0 & ExtL_1^2 & 0 & 0 \\ 0 & 0 & 0 & 0 & ExtL_2^2 & 0 \\ 0 & 0 & 0 & 0 & 0 & ExtL_3^2 \end{bmatrix} \tag{3.35}$$

$$Y_{DR_2} = \begin{bmatrix} L & 0 & 0 & 0 & 0 & 0 \\ 0 & L & 0 & 0 & 0 & 0 \\ 0 & 0 & L & 0 & 0 & 0 \\ 0 & 0 & 0 & L & 0 & 0 \\ 0 & 0 & 0 & 0 & L & 0 \\ 0 & 0 & 0 & 0 & 0 & L \end{bmatrix} \tag{3.36}$$

$$Y_{DR_3} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.37}$$

Grade blending

$2T$ rows of the constraints' coefficient matrix is belonged to this constraint. Equation (3.13) itself is divided into two following constraints:

$$\sum_{bbl=1}^{BBL} \left(Ton_{bbl} \times \left(g_{bbl} - GU_t\right)\right) \times x_{bbl,t} \le 0, \ \ \forall t \in \{1,...,T\} \tag{3.38}$$

$$\sum_{bbl=1}^{BBL} \left(Ton_{bbl} \times \left(GL_t - g_{bbl}\right)\right) \times x_{bbl,t} \le 0, \ \ \forall t \in \{1,...,T\} \tag{3.39}$$

Each of these constraints forms T rows of the coefficient matrix. The first T rows represent the upper bound and the rest represent the lower bound of the grade blending equation. Equation (3.40) shows the structure of this constraint. $\boldsymbol{X_g}$, $\boldsymbol{B_g}$, and $\boldsymbol{Y_g}$ are $2T \times (N \times T)$ matrices in which all the elements of the $\boldsymbol{B_g}$ and $\boldsymbol{Y_g}$ matrices are equal to zero. In equation (3.41), $\boldsymbol{G_l}$ and $\boldsymbol{G_u}$ are $1 \times N$ vectors in which each element of these vectors is calculated based on the tonnage of each big block, grade of the big block, acceptable range for the grade in each period, and the period of extraction. The lower and upper bounds of the grade constraint are $2T \times 1$ vectors. All the elements in the lower bound are –infinity and all the upper bound elements are zero.

$$\left[Lb_g\right] \le \left[X_g \quad \mathrm{B}_g \quad \mathrm{Y}_g\right] \le \left[Ub_g\right] \tag{3.40}$$

$$\boldsymbol{X}_g = \begin{bmatrix} \boldsymbol{G}_u & 0 & . & 0 \\ 0 & \boldsymbol{G}_u & . & 0 \\ . & . & . & . \\ . & . & . & . \\ 0 & 0 & . & \boldsymbol{G}_u \\ \boldsymbol{G}_l & 0 & . & 0 \\ 0 & \boldsymbol{G}_l & . & 0 \\ . & . & . & . \\ . & . & . & . \\ 0 & 0 & . & \boldsymbol{G}_l \end{bmatrix} \tag{3.41}$$

Binary

$N \times T + N \times (T-1)$ rows of the constraints' coefficient matrix is belonged to this constraint. Equations (3.17) and (3.18) control this constraint. The first $N \times T$ rows are for equation (3.17) and the next $N \times (T-1)$ rows are for equation (3.18). Equation (3.42) illustrates the structure of this constraint. Indices 1 and 2 are associated with equation (3.17) and (3.18) respectively. $\boldsymbol{X}_{\boldsymbol{B}_1}$, $\boldsymbol{B}_{\boldsymbol{B}_1}$, and $\boldsymbol{Y}_{\boldsymbol{B}_1}$ are $(N \times T) \times (N \times T)$ matrices while $\boldsymbol{X}_{\boldsymbol{B}_2}$, $\boldsymbol{B}_{\boldsymbol{B}_2}$, and $\boldsymbol{Y}_{\boldsymbol{B}_2}$ are $(N \times (T-1)) \times (\mathrm{N} \times \mathrm{T})$ matrices in the formulation. All the elements of the $\boldsymbol{Y}_{\boldsymbol{B}_1}$, $\boldsymbol{X}_{\boldsymbol{B}_2}$, and $\boldsymbol{Y}_{\boldsymbol{B}_2}$ are equal to zero. Equation (3.43) shows the structure of the coefficient matrix of decision variable $\boldsymbol{X}_{\boldsymbol{B}_1}$ all the elements are equal to zero except the diagonal of the matrix, which is equal to 1. For decision variable $\boldsymbol{B}_{\boldsymbol{B}_1}$ all the elements are equal to zero except the diagonal of the matrix, which is equal to -1. Equation (3.45) is the form of coefficient matrix for the decision variable B_{B_2} . In this matrix $K^t_{bbl,1,-1}$ is a $1 \times (N+1)$ vector with all the elements equal to zero except the *bbl* th and $(bbl+N)$ th elements, which are equal to 1 and -1, respectively.

$$\begin{bmatrix} Lb_{B_1} \\ Lb_{B_2} \end{bmatrix} \le \begin{bmatrix} X_{B_1} & B_{B_1} & Y_{B_1} \\ X_{B_2} & B_{B_2} & Y_{B_2} \end{bmatrix} \le \begin{bmatrix} Ub_{B_1} \\ Ub_{B_2} \end{bmatrix} \tag{3.42}$$

$$X_{B_1}=\begin{bmatrix}1 & 0 & . & . & . & . & 0\\ 0 & 1 & 0 & . & . & . & .\\ . & 0 & . & 0 & . & . & .\\ . & . & 0 & . & 0 & . & .\\ . & . & . & 0 & . & 0 & .\\ . & . & . & . & 0 & . & 0\\ 0 & . & . & . & . & 0 & 1\end{bmatrix} \tag{3.43}$$

$$B_{B_1}=\begin{bmatrix}-1 & 0 & . & . & . & . & 0\\ 0 & -1 & 0 & . & . & . & .\\ . & 0 & . & 0 & . & . & .\\ . & . & 0 & . & 0 & . & .\\ . & . & . & 0 & . & 0 & .\\ . & . & . & . & 0 & . & 0\\ 0 & . & . & . & . & 0 & -1\end{bmatrix} \tag{3.44}$$

$$\boldsymbol{B}_{\boldsymbol{B}_2}=\begin{bmatrix}\boldsymbol{K}^1_{1,1,-1} & 0 & . & . & . & . & 0\\ 0 & \boldsymbol{K}^1_{2,1,-1} & 0 & . & . & . & .\\ . & 0 & . & 0 & . & . & .\\ . & . & 0 & . & 0 & . & .\\ . & . & . & 0 & . & 0 & .\\ . & . & . & . & 0 & . & 0\\ 0 & . & . & . & . & 0 & \boldsymbol{K}^{\boldsymbol{t}}_{\boldsymbol{bbl},1,-1}\end{bmatrix} \tag{3.45}$$

Number of new big blocks

$2\times T$ rows of the constraints' coefficient matrix is belonged to this constraint. Equation (3.19) is divided into two equations ((3.46) and (3.47)) and equation (3.20) is divided into two equations((3.48) and (3.49)). Equations (3.46) and (3.48) form the upper bound of the equations (3.19) and (3.20), respectively. Equations (3.47) and (3.49) form the lower bound of the equations (3.19) and (3.20), respectively. Equation (3.50) shows the structure of this constraint. The first two rows of this constraint' coefficient matrix belong to equations (3.46) and (3.47). On the other hand, the next $2\times(T-1)$ rows of the constraint' coefficient matrix is for equations (3.48) and (3.49). Index 1 is used for equations (3.46) and

(3.47), whereas index 2 is used for equations (3.48) and (3.49). $\boldsymbol{X}_{\boldsymbol{NBBL}_1}$, $\boldsymbol{B}_{\boldsymbol{NBBL}_1}$, and $\boldsymbol{Y}_{\boldsymbol{NBBL}_1}$ are $2\times(N\times T)$ matrices while $\boldsymbol{X}_{\boldsymbol{NBBL}_2}$, $\boldsymbol{B}_{\boldsymbol{NBBL}_2}$, and $\boldsymbol{Y}_{\boldsymbol{NBBL}_2}$ are $(2\times(T-1))\times(\mathrm{N}\times\mathrm{T})$ matrices in the formulation. All the elements of decision variables $\boldsymbol{X}_{\boldsymbol{NBBL}_1}$, $\boldsymbol{X}_{\boldsymbol{NBBL}_2}$, $\boldsymbol{Y}_{\boldsymbol{NBBL}_1}$, and $\boldsymbol{Y}_{\boldsymbol{NBBL}_2}$ are equal to zero. Equation (3.51) illustrates the structure of coefficient matrix related to decision variable $\boldsymbol{B}_{\boldsymbol{NBBL}_1}$ in which $\boldsymbol{K}_1^1$ is a $1\times(N\times T)$ vector with all the elements equal to zero except the first N elements which are equal to 1. Also, $\boldsymbol{K}_{-1}^1$ is a $1\times(N\times T)$ vector with all the elements equal to zero except the first N elements which are equal to -1. Equation (3.52) shows the structure of the coefficient matrix related to decision variable $\boldsymbol{B}_{\boldsymbol{NBBL}_2}$ in which $\boldsymbol{K}_1^t$ is a $1\times N$ vector with all the elements equal to 1 related to period t , whereas $\boldsymbol{K}_{-1}^{t-1}$ is a $1\times N$ vector with all the elements equal to -1 at period $t-1$.

$$\sum_{bbl=1}^{BBL} B_{bbl,t} - \overline{N}_{NBBL,1} \le 0\,, \quad t=1 \tag{3.46}$$

$$\underline{N}_{NBBL,1} - \sum_{bbl=1}^{BBL} B_{bbl,t} \le 0, \quad t=1 \tag{3.47}$$

$$(\sum_{bbl=1}^{BBL} B_{bbl,t} - \sum_{bbl=1}^{BBL} B_{bbl,t-1}) - \overline{N}_{NBBL,t} \le 0, \quad t\in\{2,...,T\} \tag{3.48}$$

$$\underline{N}_{NBBL,t} - (\sum_{bbl=1}^{BBL} B_{bbl,t} - \sum_{bbl=1}^{BBL} B_{bbl,t-1}) \le 0, \quad t\in\{2,...,T\} \tag{3.49}$$

$$\begin{bmatrix} Lb_{NBBL_1} \\ Lb_{NBBL_2} \end{bmatrix} \le \begin{bmatrix} X_{NBBL_1} & B_{NBBL_1} & Y_{NBBL_1} \\ X_{NBBL_2} & B_{NBBL_2} & Y_{NBBL_2} \end{bmatrix} \le \begin{bmatrix} Ub_{NBBL_1} \\ Ub_{NBBL_2} \end{bmatrix} \tag{3.50}$$

$$\boldsymbol{B}_{\boldsymbol{NBBl}_1} = \begin{bmatrix} \boldsymbol{K}_1^1 \\ \boldsymbol{K}_{-1}^1 \end{bmatrix} \tag{3.51}$$

$$\boldsymbol{B}_{NBBl_2} = \begin{bmatrix} \boldsymbol{K}_{-1}^{1} & \boldsymbol{K}_{1}^{2} & 0 & . & . & 0 \\ 0 & \boldsymbol{K}_{-1}^{2} & \boldsymbol{K}_{1}^{3} & 0 & . & . \\ . & 0 & . & . & 0 & . \\ . & . & 0 & . & . & 0 \\ 0 & 0 & . & 0 & \boldsymbol{K}_{-1}^{t-1} & \boldsymbol{K}_{1}^{t} \\ \boldsymbol{K}_{1}^{1} & \boldsymbol{K}_{-1}^{2} & 0 & . & 0 & 0 \\ 0 & \boldsymbol{K}_{1}^{2} & \boldsymbol{K}_{-1}^{3} & 0 & . & . \\ . & 0 & . & . & 0 & . \\ . & . & 0 & . & . & 0 \\ . & . & . & 0 & \boldsymbol{K}_{1}^{t-1} & \boldsymbol{K}_{-1}^{t} \end{bmatrix} \qquad (3.52)$$

Precedence

$N \times T$ rows of the constraints' coefficient matrix is belonged to this constraint. Equation (3.21) controls this constraint. Equation (3.53) shows the structure of precedence constraint. $\boldsymbol{X_P}$, $\boldsymbol{B_P}$, and $\boldsymbol{Y_P}$ are $(N \times T) \times (N \times T)$ matrices in the formulation. All the elements of the decision variables, $\boldsymbol{X_P}$ and $\boldsymbol{Y_P}$ are equal to zero. Equation (3.54) illustrates the structure of the decision variable $\boldsymbol{B_P}$ in which $\boldsymbol{K}_{-1,\boldsymbol{n}}^{\boldsymbol{t}}$ is a $N \times N$ vector that the elements are placed according to the number of predecessors for each big block, -1 for the predecessors of a given big block and n for that given big block in period t .

$$\left[Lb_P\right] \leq \left[X_P \quad B_P \quad Y_P\right] \leq \left[Ub_p\right] \qquad (3.53)$$

$$\boldsymbol{B_P} = \begin{bmatrix} \boldsymbol{K}_{-1,\boldsymbol{n}}^{1} & 0 & . & . & 0 \\ 0 & \boldsymbol{K}_{-1,\boldsymbol{n}}^{2} & 0 & . & . \\ . & 0 & . & 0 & . \\ . & . & 0 & . & 0 \\ 0 & . & . & 0 & \boldsymbol{K}_{-1,\boldsymbol{n}}^{\boldsymbol{t}} \end{bmatrix} \qquad (3.54)$$

Reserves

N rows of the constraints' coefficient matrix belonged to this constraint. Equation (3.22) controls this constraint. Equation (3.55) shows the structure of this constraint. $\boldsymbol{X_{res}}$, $\boldsymbol{B_{res}}$, and $\boldsymbol{Y_{res}}$ are $(N)\times(N\times T)$ matrices in the formulation. All the elements of the matrices $\boldsymbol{B_{res}}$, and $\boldsymbol{Y_{res}}$ are zero. Equation (3.56) demonstrates the structure of the $\boldsymbol{X_{res}}$, in which $\boldsymbol{K}_{\boldsymbol{bbl}}^{\boldsymbol{T},1}$ is a $1\times N$ vector that all the elements are zero except the *bbl th* element, which is equal to 1. *bbl* represents the ID number of the big-block. Lb_{res} and Ub_{res} are $N\times 1$ vectors of ones in the formulation.

$$\left[Lb_{res}\right]\leq\left[X_{res}\quad B_{res}\quad Y_{res}\right]\leq\left[Ub_{res}\right] \tag{3.55}$$

$$\boldsymbol{X_{res}}=\begin{bmatrix} \boldsymbol{K}_1^{1,1} & \boldsymbol{K}_1^{2,1} & . & \boldsymbol{K}_1^{\boldsymbol{T}-1,1} & \boldsymbol{K}_1^{\boldsymbol{T},1} \\ \boldsymbol{K}_2^{1,1} & \boldsymbol{K}_2^{2,1} & . & \boldsymbol{K}_2^{\boldsymbol{T}-1,1} & \boldsymbol{K}_2^{\boldsymbol{T},1} \\ . & . & . & . & . \\ . & . & . & . & . \\ . & . & . & . & . \\ . & . & . & . & . \\ \boldsymbol{K}_{\boldsymbol{bbl}}^{1,1} & \boldsymbol{K}_{\boldsymbol{bbl}}^{2,1} & . & \boldsymbol{K}_{\boldsymbol{bbl}}^{\boldsymbol{T}-1,1} & \boldsymbol{K}_{\boldsymbol{bbl}}^{\boldsymbol{T},1} \end{bmatrix} \tag{3.56}$$

3.5 Summary and Conclusion

In summary, stochastic sequential simulation is implemented to deal with the grade uncertainty in the first part of this chapter. Average-simulated block model is created through calculating weighted average grade of all the realizations for each cell. Afterwards, the best extraction level is determined according to the maximum discounted profit. The discounted ore profit is calculated based on the discount rate, profit values of each block, extraction rate, and the distance between the center points of each ore block and its above ore block. Determining the outline of the ore-body for each level and creating the big-blocks are the next parts that discussed in this chapter. Then, MILP formulation framework is developed in which the main objective is to maximize NPV in presence of a number of practical constraints.

Finally, the numerical model of the MILP formulations is created in MATLAB (Math Works Inc., 2015) and a generalized form is used by IBM/CPLEX (IBM, 2015) to solve large-scale MILP problems. The structure of all the vectors and matrices related to objective function and constraints is explained at the next part and the results are passed on to IBM/CPLEX for optimization.

CHAPTER 4

VERIFICATION, EXPERIMENTS, AND DISCUSSION OF RESULTS

Chapter 4 presents experimentation with the stochastic sequential simulation, best level determination, and MILP model framework. This includes two case studies called original and average-simulated block models and verification of the methods. The MILP formulation is carried out and certified for both the block models. The simulation methodology is implemented and verified on a dataset from drillholes. The best level determination method and MILP model are applied on the simulated block models. Finally, the near-optimal realistic production plan under grade uncertainty is generated. At the end of this chapter, risk analysis has been done to investigate the effect of grade uncertainty on NPV and tonnage.

4.1 Introduction

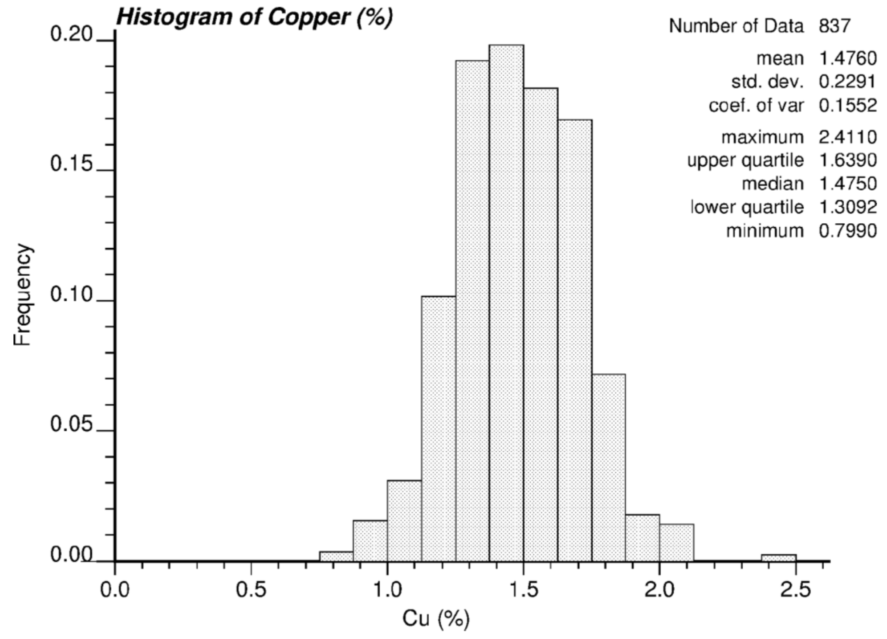
In this chapter, the stochastic sequential simulation algorithm and mathematical formulations are implemented on a standard drillhole dataset in order to demonstrate how the methodology works. First the simulation method was applied on the dataset to consider grade uncertainty by creating average-simulated block model, and then the methodology to find the best level of extraction is tested on all the realizations, original and average-simulated block models, and finally MILP model was examined and verified on both original and average-simulated block models to generate the near-optimal production schedule. Then, some investigation and comparison were studied to assess the involved risks and NPV changes due to existence of grade uncertainty.

4.2 Grade uncertainty

A geostatistical study based on the drillhole data of a copper deposit and according to what is mentioned in section 3.2.1 was performed. Geostatistical software library (GSLIB) (Deutsch and Journel, 1998) was used for geostatistical modeling in this research. The only rock property that is used in this study is copper. The histogram and cumulative density function (CDF) of the data is shown in Figure 4.1. The data set has 837 samples of percent copper with the mean and standard deviation of 1.47 and 0.22 %, respectively. Also, from the CDF curve, most of the data are between 1.2 and 1.8 % of copper. As the copper grade is univariate data, there is no need for multivariate statistical analysis and transferring data to multivariate Gaussian framework to find the correlation between the variables. The initial inspection of the locations of the drillholes showed that the drillholes were equally spaced. As a result, the declustering algorithm was not implemented.

The next step in the study was to define the grid for the simulation. The distance between the grid nodes in each direction, the number of grid nodes in each direction, and the coordinates of the first grid node are important parameters for defining a grid. As it is demonstrated in Table 4.1, all of these parameters were considered in order to choose the size of the grid.

(a)



(b)

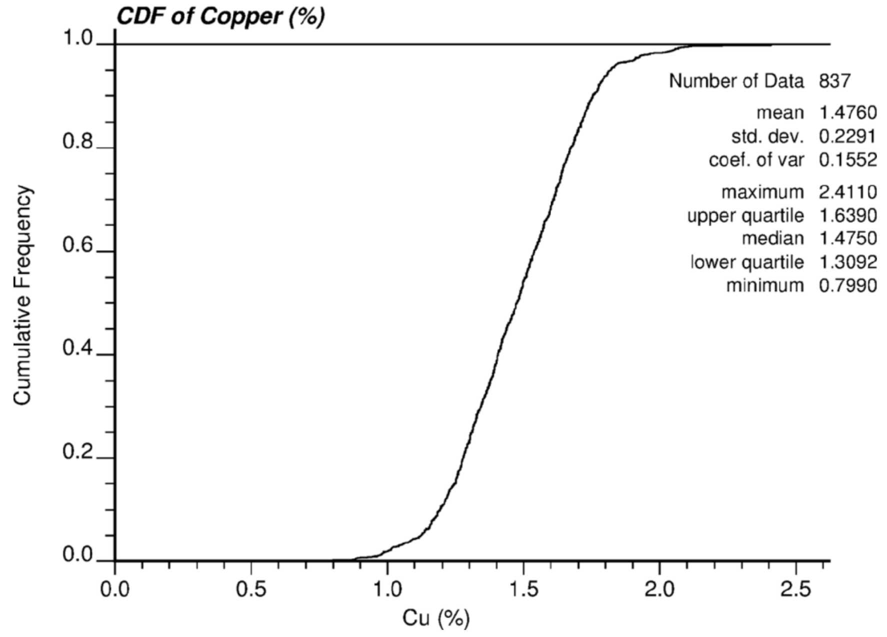


Figure 4.1. (a) Histogram and (b) the CDF of copper grade (%) for drillholes

Table 4.1. Grid definition for geostatistical study

Direction	Number of nodes	Center coordinates of first node (m)	Grid Spacing
Easting	45	105	10
Northing	60	5	10
Elevation	70	305	10

There were two parts to the modeling: rock type modeling and grade modeling. The grade modeling should be implemented for both rock types (ore and waste) separately, but as the grade for all the waste blocks were zero, the grade modeling was performed just for ore blocks. In this research, rock type 1 and 0 represent ore and waste blocks, respectively.

4.2.1 Rock type modeling

The principal directions of continuity were found using indicator kriging (*ik3d* program) based on two categories: 0 (waste) and 1 (ore). The azimuths of major and minor directions were chosen to be 0 and 90 degrees and were used at the next step to calculate the variograms. Afterwards, the indicator variograms were calculated and a theoretical variogram model was fitted with three structures, the nugget effect of 0, and the sill of 0.14. *Varcalc*, *varmodel* and *varplot* programs are utilized for this purpose, respectively. Figure 4.2 shows the plan view of maximum direction of continuity for rock type 1 at Elevation 40. Experimental directional variograms and the fitted models are illustrated in Figure 4.3. At the next step, 20 realizations for rock type 1 were generated using Sequential Indicator Simulation (SIS) algorithm. *Blocksis* program is used for this purpose that generates multiple realizations. Plan view of rock type simulation for first realization at Elevation 40 is shown in Figure 4.4.

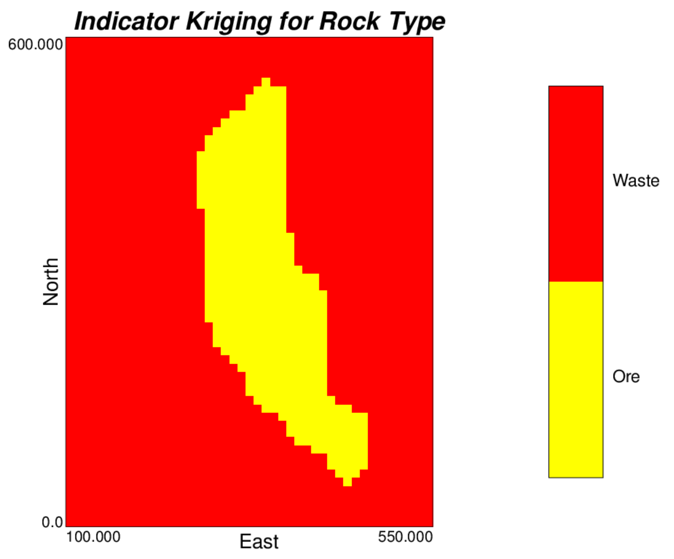


Figure 4.2. Plan view of maximum direction of continuity for rock types at Elevation 40

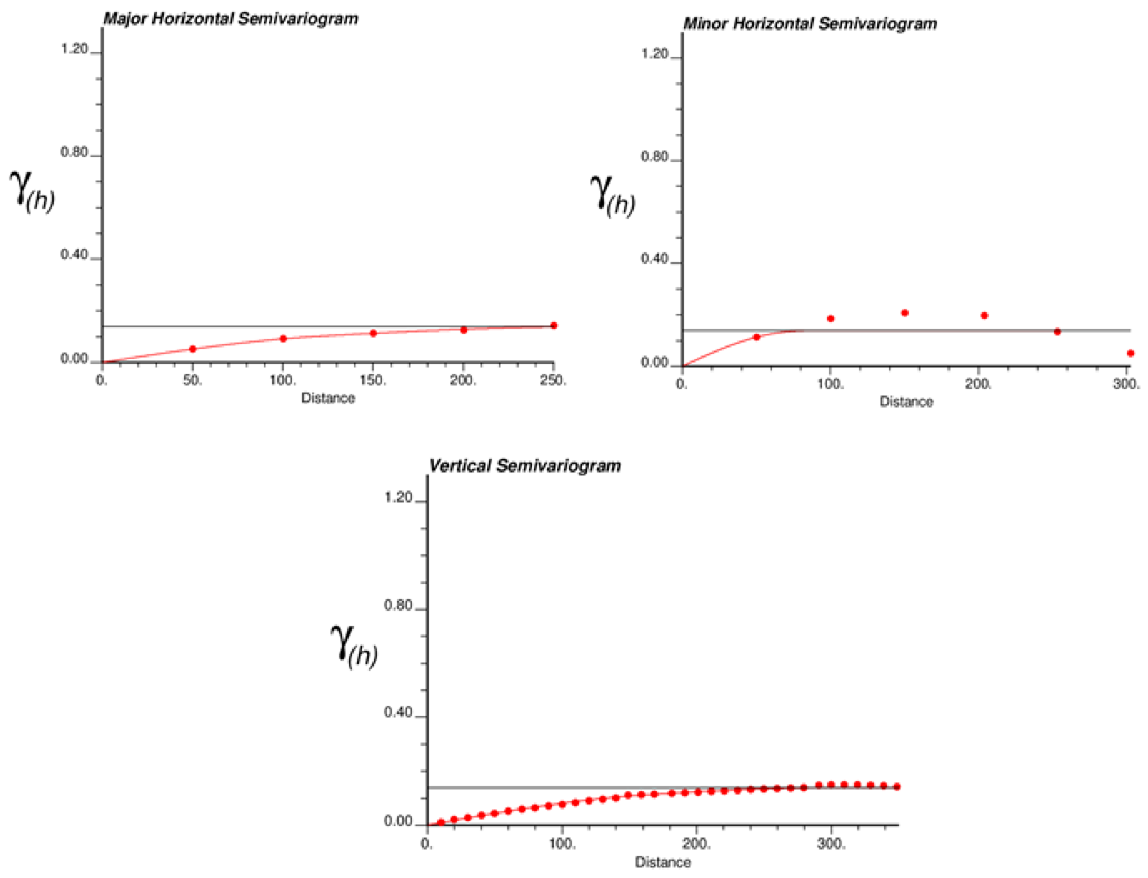


Figure 4.3. Experimental directional variograms (dots) and the fitted variogram models (solid lines) for rock type and distance units in meters

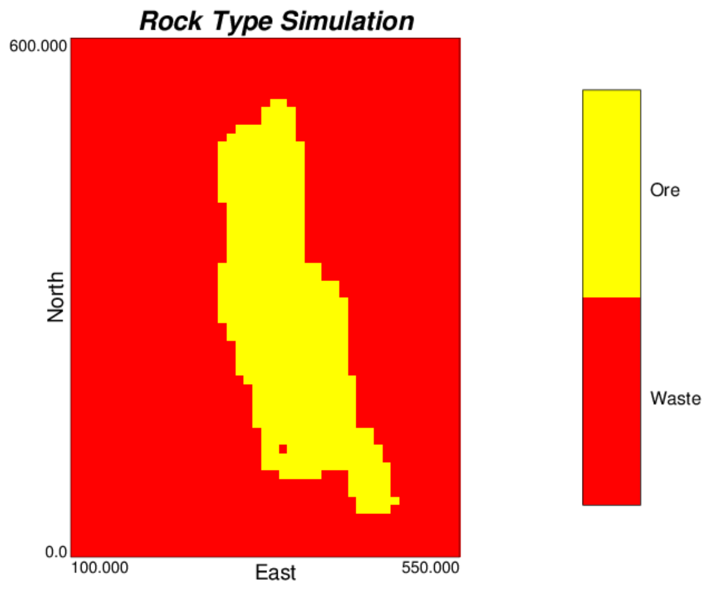


Figure 4.4. Plan view of rock type simulation for first realization at Elevation 40

4.2.2 Grade modeling

For ore modeling, the principal directions of continuity were extracted by doing simple kriging with the help of arbitrary variograms. As the mean of data was known, simple kriging was used instead of ordinary kriging. *kt3dn* program was implemented for this purpose. As it can be seen in Figure 4.5, the azimuth of 90° (major) and zero (minor) in the horizontal direction were selected for variogram calculation at the next step. Then the copper grades were transformed to Gaussian space with the help of *nscore* program. Traditional variogram calculation and modeling with three structures and a nugget effect of 0.1 were done for the copper grade. Following the same procedure as rock type modeling, *Varcalc*, *varmodel* and *varplot* programs are utilized in this step, respectively. Figure 4.6 shows the experimental directional variogarms and their fitted models. Afterwards, 20 realizations for the copper grade were generated by using *sgsim* program which is based on Sequential Gaussian Simulation algorithm (SGS). The SGS needs a back-transformation to original units by using *backtr* program. The plan view of copper grade simulation for first realization at Elevation 40 is shown in Figure 4.7.

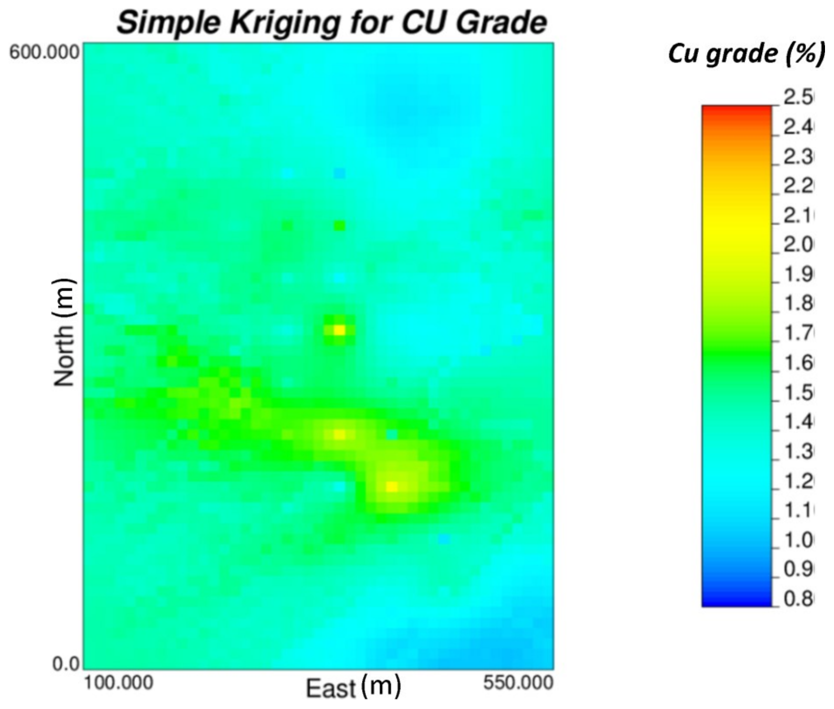


Figure 4.5. Plan view of maximum direction of continuity for copper grade (%) at Elevation 40

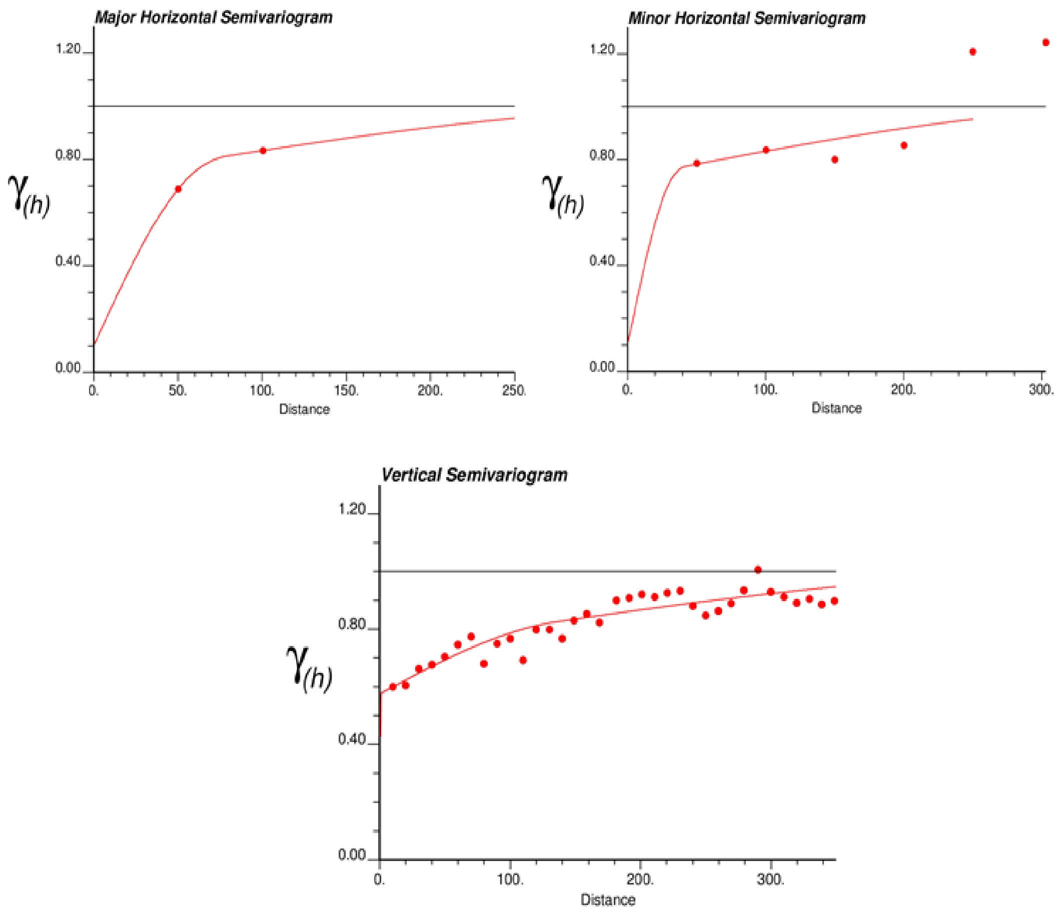


Figure 4.6. Experimental directional variograms (dots) and the fitted variogram models (solid lines) for the Cu grade of ore blocks, distance units in meters

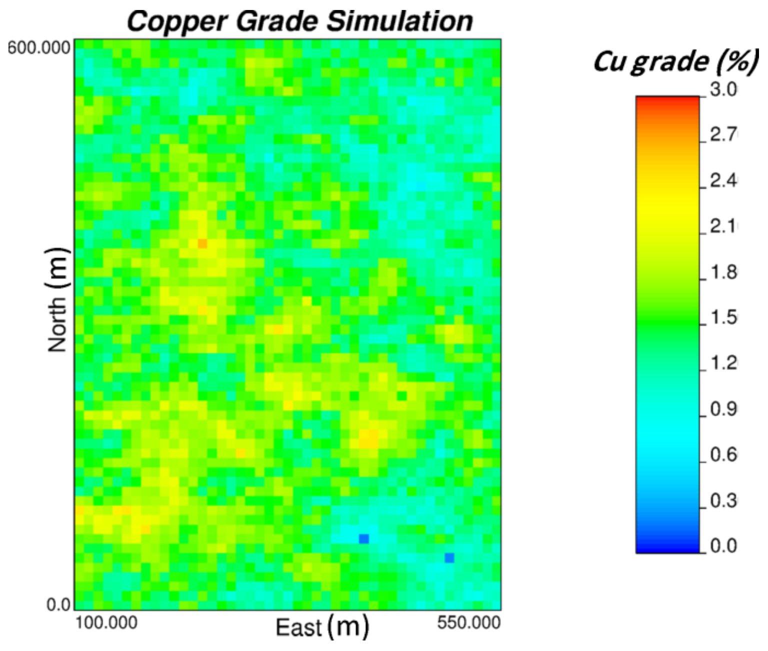


Figure 4.7. Plan view of Cu grade (%) simulation for first realization at Elevation 40

4.2.3 Merging grade models into rock type models

The next step was to match and merge the rock type model with the grade model for each realization using *mergemod* program. Figure 4.8 shows the plan view of the final simulation for the first realization. Figure 4.9 shows the variogram reproduction of the copper (ore) simulation (left) and rock type simulation (right) in three major, minor, and vertical directions. Since the variograms were reproduced quite reasonably, the generated realizations were considered representative of the grade uncertainty.

In order to create average-simulated block model only grids that were ore in 85% of the realizations were considered and average grade was calculated for those grids.

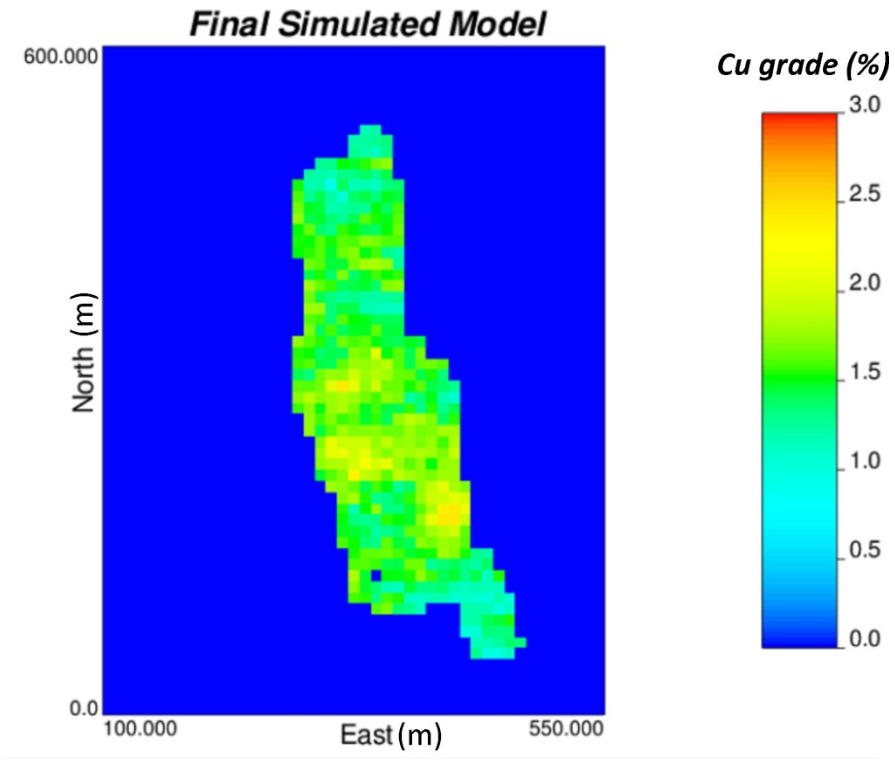


Figure 4.8. Final simulation of first realization at Elevation 40

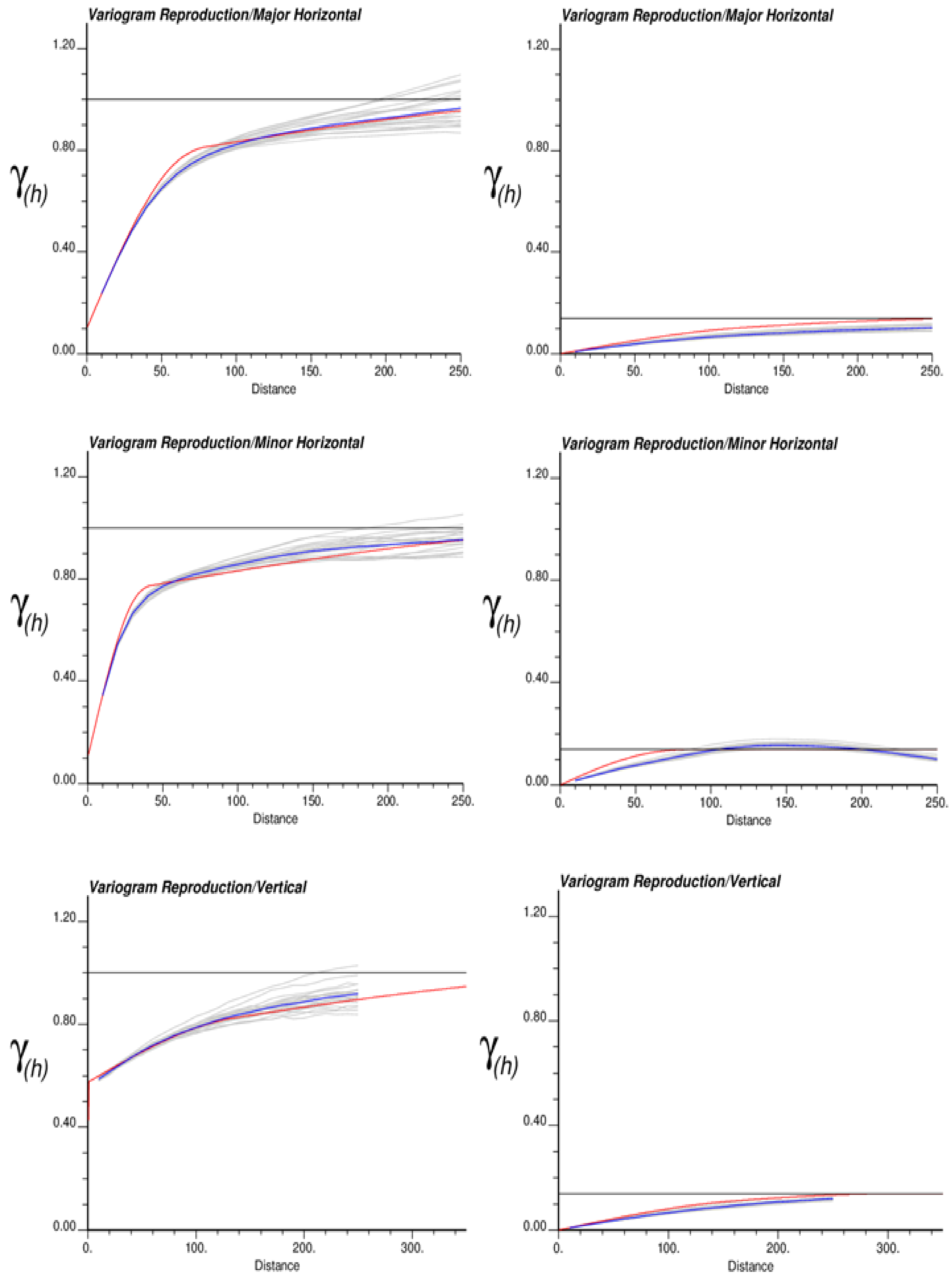


Figure 4.9. Variogram reproduction at Gaussian units of copper grade (left) and rock type (right) realizations (gray lines), the reference variogram model (red line), and the average variogram from realizations (blue line) in three directions.

4.3 Placement of the extraction level

The discounted profit and tonnage of the ore blocks above each ore block in each level were calculated and the profit-tonnage curve was plotted. The input parameters for calculating discounted profit as mentioned in section 3.2.2 are block height and extraction rate which were assumed to be 10 meters and 15 (meter/period), respectively. This led to selecting the best level for starting extraction based on maximum discounted profit for each realization.

According to the procedure described in section 3.2.1 and 3.2.2, the appropriate level of extraction was determined for original block model, average-simulated block model, and all realizations. Figure 4.10, and Figure 4.11 illustrate the best level of extraction for average-simulated and original block models, respectively. For original block model level 38, and for average-simulated block model level 39 have the maximum discounted profit. Figure 4.12 shows the histogram of the obtained extraction levels for all realizations, in 40 % of the realizations, level 39 is the best level of extraction. In this case, level 39 of average-simulated block model with the highest discounted profit was selected for starting the extraction.

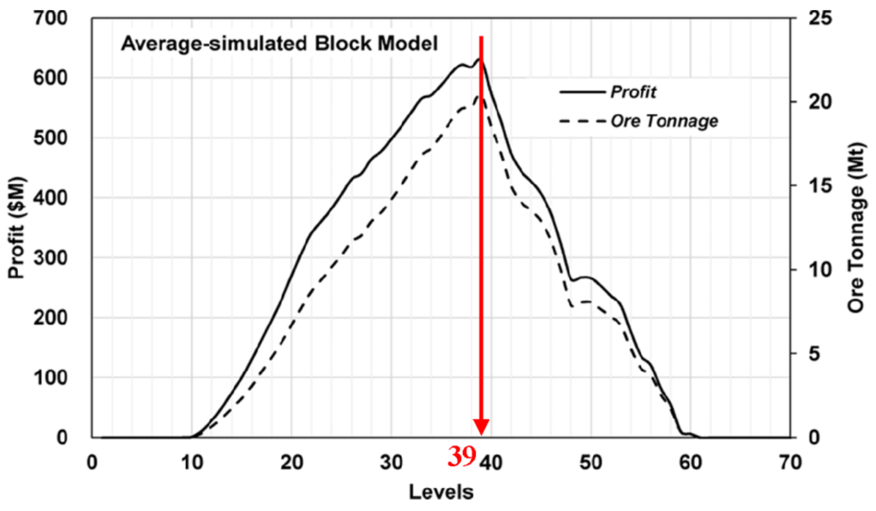


Figure 4.10. Best level selection based on tonnage-profit curve of average-simulated block model

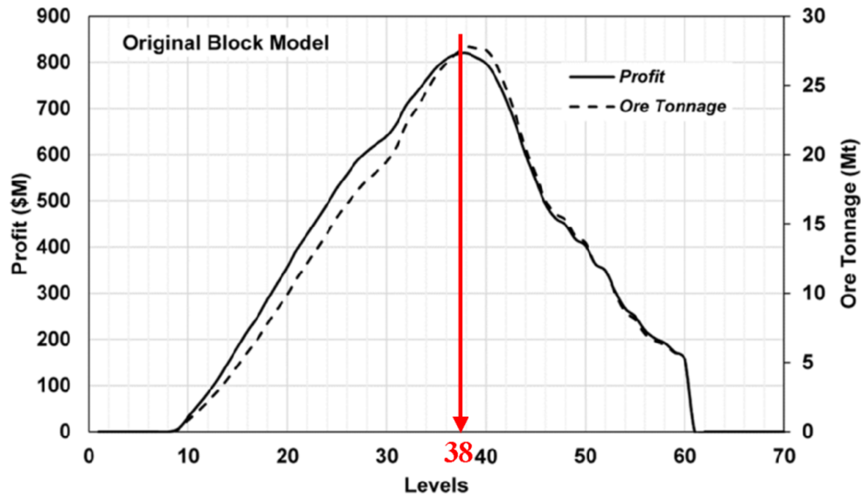


Figure 4.11. Best level selection based on tonnage-profit curve of original block model

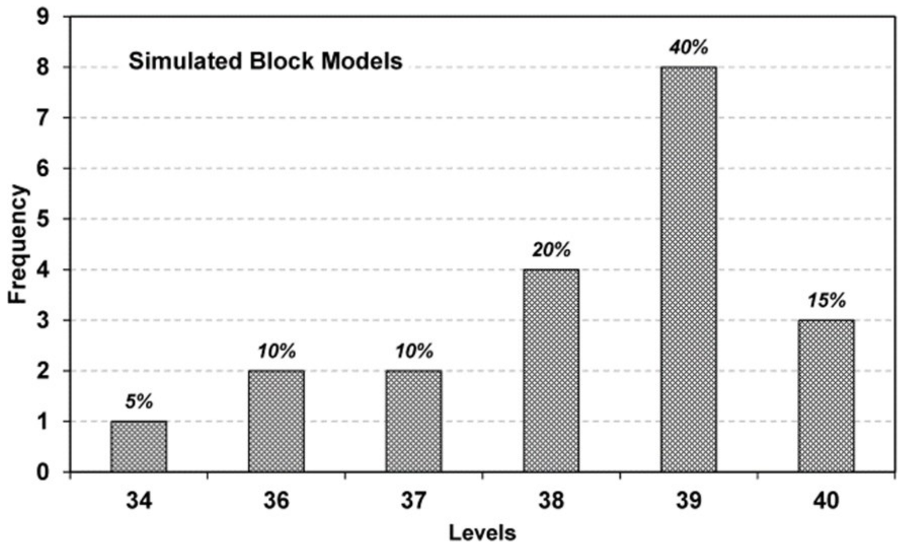


Figure 4.12. Histogram of best level of extraction for all the realizations

4.4 Production scheduling

In this section, the procedure explained in section 3.2.3 to obtain the optimal production schedule for the block-cave mine has been examined. At first, the verification of the MILP model has been done on the original and average-simulated block models to make sure that the models work properly. The gap tolerance (EPGAP) of 1% was used as an optimization termination criterion to solve the models.

4.4.1 Original block model

To maximize the NPV, the proposed mathematical model was applied to generate the production schedule for the best level of original model which was level 38 from the previous section. The case consists of 189,000 blocks which 20,889 of them were ore blocks. Each block is 10×10×10 m (Figure 4.13).

The ore blocks layout for level 38 was determined (Figure 4.13b). Then based on the method presented by Khodayari and Pourrahimian (2015) the best advancement direction for level 38 was determined to be from Southeast to Northwest ($SE \rightarrow NW$),see Figure 4.14a. Then, because of the distances between drawpoints and the assumed footprint size (30m × 30m), the blocks are placed into bigger blocks along the advancement direction. Additionally, as the big-blocks close to the boundaries did not constitute a complete set (with nine small blocks), only sets with seven or more blocks were considered (see Figure 4.14b). A big-block contains seven, eight, or nine small ore blocks and small ore blocks above that big-block in the extraction level. Afterwards, the average grade of new big-blocks column was calculated using a weighted average method. Also, the total ore tonnage and profit values of each big-block column were calculated. After the big-block columns were created, the optimal production schedule was generated for the columns using MILP model. The objective was to maximize the NPV.

Table 4.2 illustrates the number of variables and constraints used in the formulation. Table 4.3 shows the scheduling parameters to generate the production schedule. The coefficient matrices were created in MATLAB (Math Works Inc., 2015). IBM/CPLEX (IBM, 2015) was used to solve the problem. The model was run for level 38 with 90 big-block columns over 15 periods. The amount of extracted ore was 37.45 Mt with the NPV of $925.1 M. Figure 4.15 shows the production average grade and production tonnage in each period for this level. As it can be seen from production graph, the maximum amount of material has been extracted in all the periods except period 15 which slightly drops. Also from grade graph, it has been increased gradually in early periods and the material with higher grades were extracted at first and then it has been decreased slowly near the end of the mine life.

Figure 4.16 shows the number of active and new big-blocks in which the number of new big-blocks was within the defined range. The formulation tries to open more big-blocks at first period in order to maximize the NPV and because of that 25 big-blocks were opened at period one. Moreover, maximum number of active big-blocks was 28. The precedence of extraction is shown in Figure 4.17.

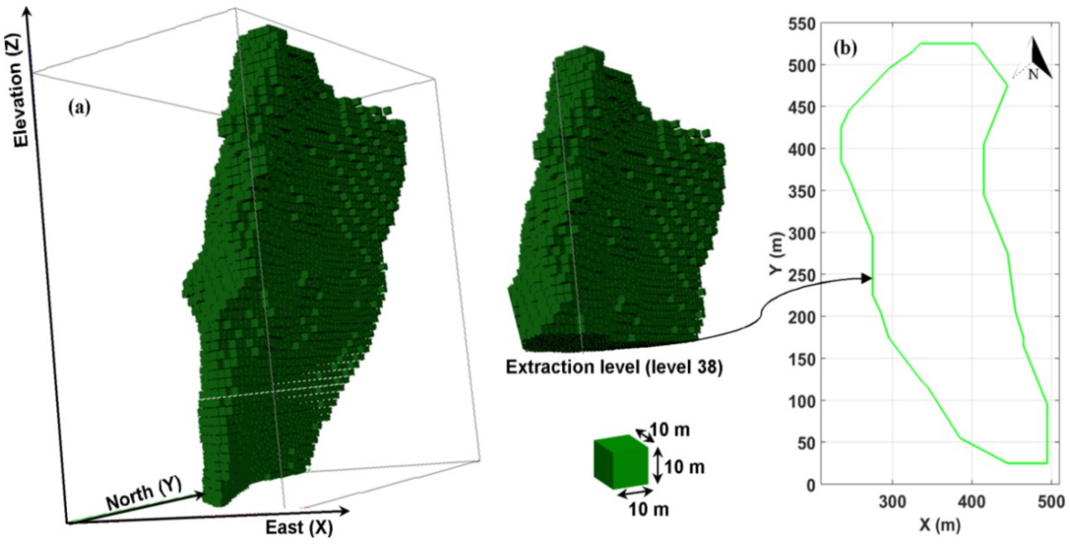


Figure 4.13. (a) Block model of ore-body, (b) outline of ore-body at level 38

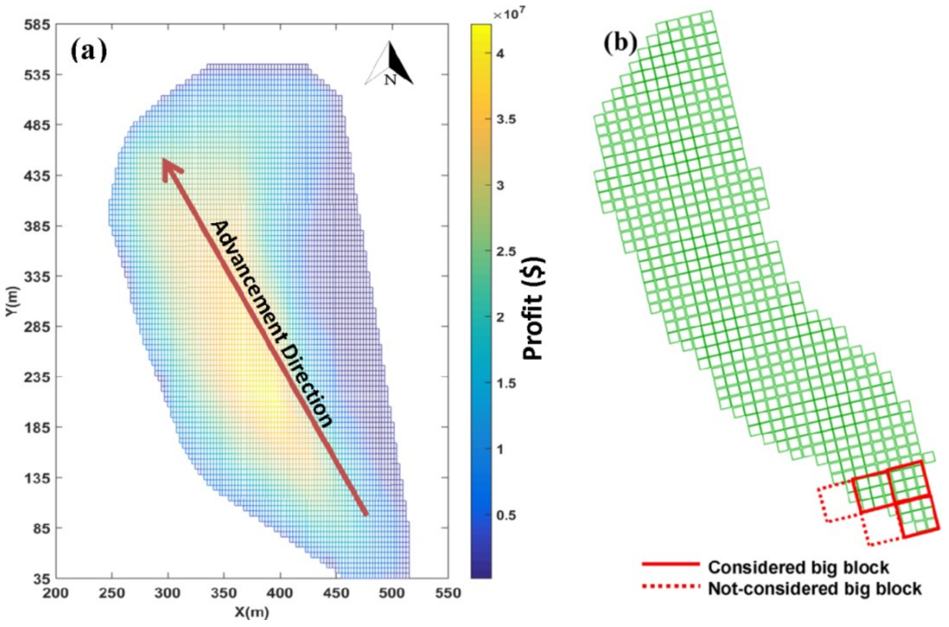


Figure 4.14. (a) Best advancement Direction based on the profit at the best level, (b) Schematic view of considering big blocks with more than seven small blocks

Table 4.2. Number of variables and constraints for original block model

Number of big-blocks	Number of constraints	Decision variables		
		Total	Continuous	Binary
90	8,190	4,050	1,350	2,700

Table 4.3. Scheduling parameters for original block model

Parameter	Value	Parameter	Value
T	15	$P\,(\$/tonne)$	6,000
$MCL\,(Mt)$	1	$S_C\,(\$/tonne)$	0.5
$MCU\,(Mt)$	2.5	$M_C\,(\$/tonne)$	10
$GL\,(\%)$	1.1	$P_C\,(\$/tonne)$	16.1
$GU\,(\%)$	1.7	$\overline{N}_{NBBL,1}$	25
$ExtL\,(Kt)$	90	$\underline{N}_{NBBL,1}$	0
$ExtU\,(Kt)$	350	$\overline{N}_{NBBL,t}$	5
$\mathrm{i}\,(\%)$	10	$\underline{N}_{NBBL,t}$	4
$R\,(\%)$	85	L	100,000,000

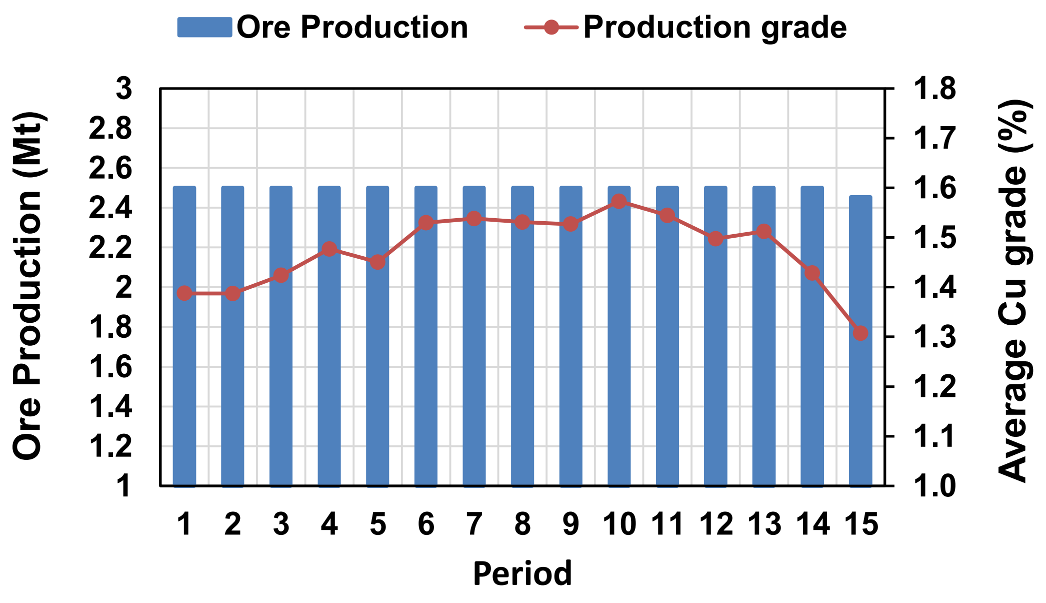


Figure 4.15. Production tonnage and average grade of production at level 38

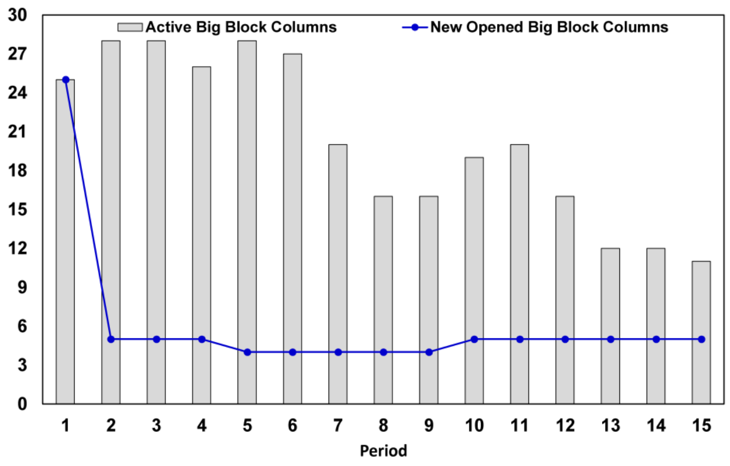


Figure 4.16. Number of active and new big-blocks for each period at level 38

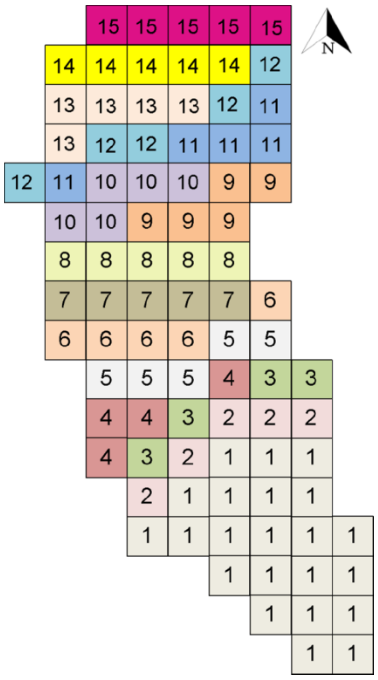


Figure 4.17. Starting extraction period of big-blocks at level 38 (numbers represent the starting period)

4.4.2 Average-simulated block model

In this section, the algorithm and MILP formulation were applied on the average-simulated block model to generate the optimal production plan in presence of grade uncertainty. Average-simulated block model contains 189,000 blocks which 14,956 of them were ore blocks. Each block is 10×10×10 m. As described in section 4.3, level 39 with the highest discounted profit was selected for starting the extraction. Figure 4.19a shows level 39 ore blocks and above. Figure 4.18a and b show the histogram and cumulative density function (CDF) of copper above best level (level 39) of average-simulated block model, respectively. The numbers of ore blocks above level 39 were 10,580. As it can be seen from the graphs, mean and standard deviation were 1.517 and 0.137, respectively. And CDF shows that most of the ore blocks have a range grade between 1.2% and 1.6%.

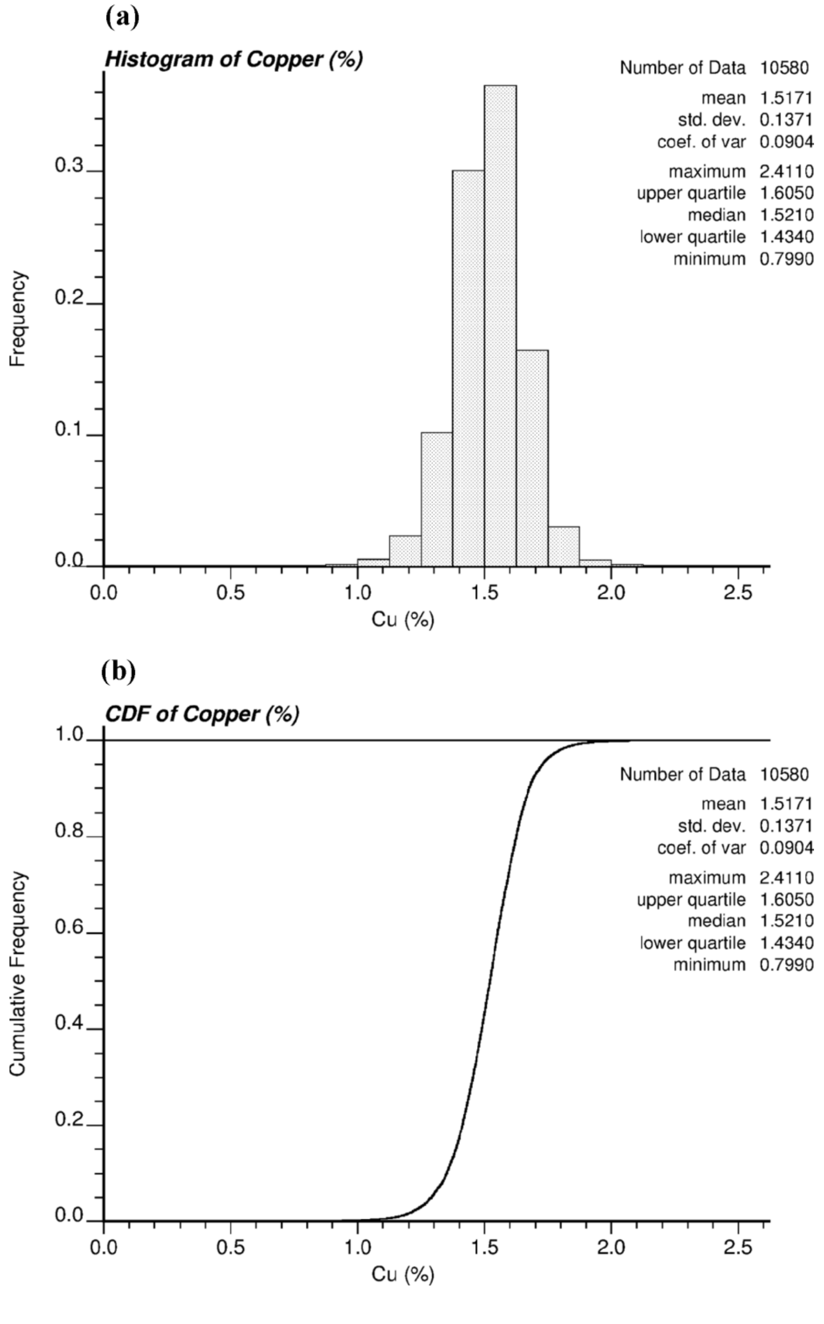


Figure 4.18. (a) Histogram and (b) the CDF of copper grade (%) above level 39 for average-simulated block model

According to the methodology presented by Khodayari and Pourrahimian (2015) the best advancement direction for this level was specified which in this case was from $SE \rightarrow NW$ (Figure 4.20a). Following the same procedure as original block model, the big-block columns were created (Figure 4.19b). The next step is calculating the average grade and total ore tonnage and profit values of the created big block columns. The optimal production schedule was generated for these columns using proposed mathematical formulation.

The number of constraints, decision variables, and big-block columns which is 66 in this case is shown in Table 4.4. Table 4.5 shows the scheduling parameters that were used in running the mathematical programming. The constraints' coefficient matrix was created in MATLAB (Math Works Inc., 2015) according to what described in section 3.4. IBM/CPLEX (IBM, 2015) was implemented as a solver to maximize the NPV. The results were 27.33 Mt of extracted ore and NPV of $ 726.5 M.

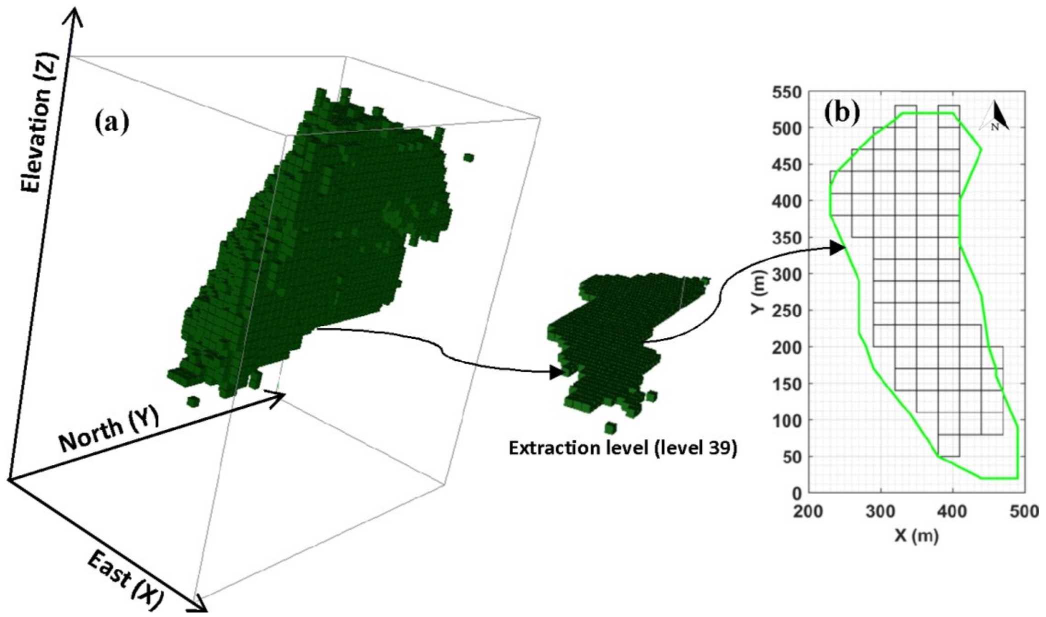


Figure 4.19. (a) Block model of ore-body above level 39, (b) outline of ore-body and created big blocks at level 39

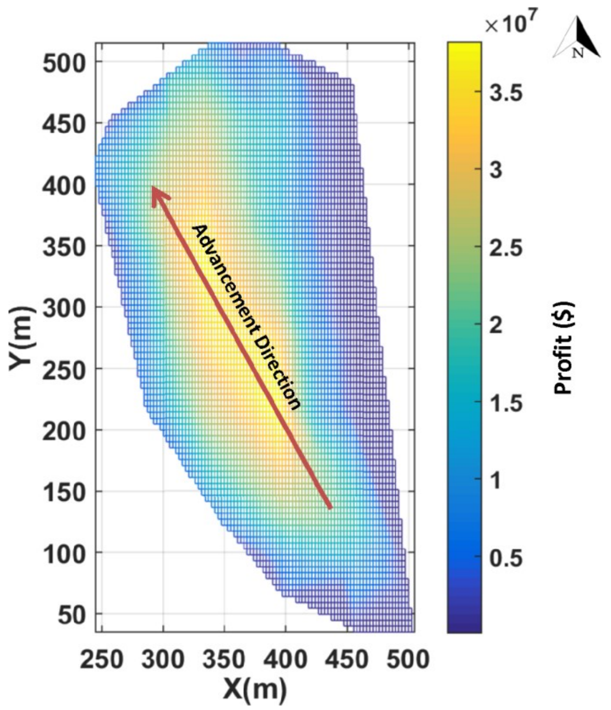


Figure 4.20. Best advancement direction based on the profit at level 39

Table 4.4. Number of variables and constraints for average-simulated block model

Number of big-blocks	Number of constraints	Decision variables		
		Total	Continuous	Binary
66	6,030	2,970	990	1,980

Table 4.5. Scheduling parameters for average-simulated block model

Parameter	Value	Parameter	Value
T	15	$P\,(\$/tonne)$	6,000
$MCL\,(Mt)$	1.2	$SC\,(\$/tonne)$	0.5
$MCU\,(Mt)$	1.875	$MC\,(\$/tonne)$	10
$GL\,(\%)$	1.3	$PC\,(\$/tonne)$	16.1
$GU\,(\%)$	1.6	$\overline{N}_{NBBL,1}$	15
$ExtL\,(Kt)$	90	$\underline{N}_{NBBL,1}$	0
$ExtU\,(Kt)$	350	$\overline{N}_{NBBL,t}$	4
$\mathrm{i}\,(\%)$	10	$\underline{N}_{NBBL,t}$	2
$R\,(\%)$	85	L	100,000,000

The ore production tonnage and average grade in each period for level 39 is shown in Figure 4.21 and Figure 4.22, respectively. As it can be seen, the formulation tried to keep the mining capacity at the upper bound and the material with highest grades was used in the early years of production. The number of active and new opened big-block columns for each period is demonstrated in Figure 4.23 which is within the defined range. Figure 4.24 shows the precedence of extraction at level 39. The results show that all assumed constraints were satisfied.

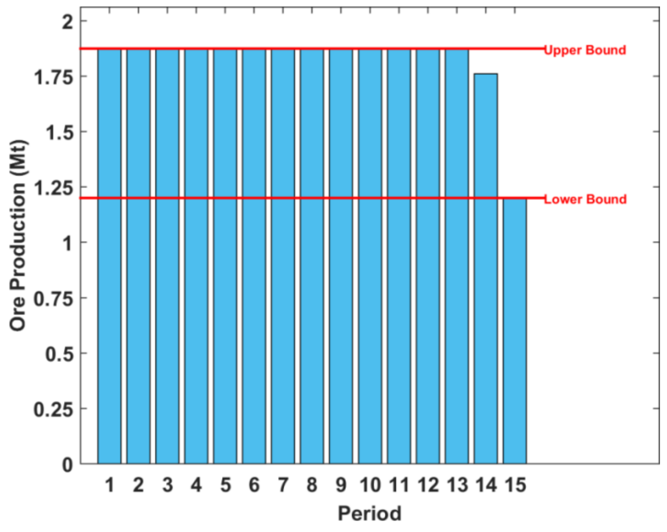


Figure 4.21. Ore production tonnage at level 39

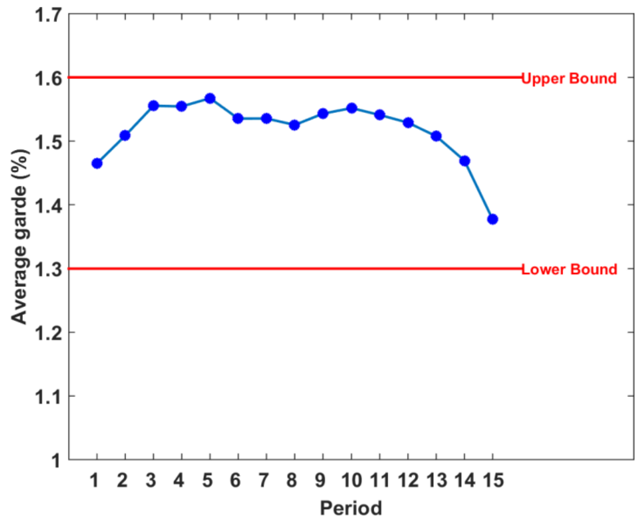


Figure 4.22. Average grade of production at level 39

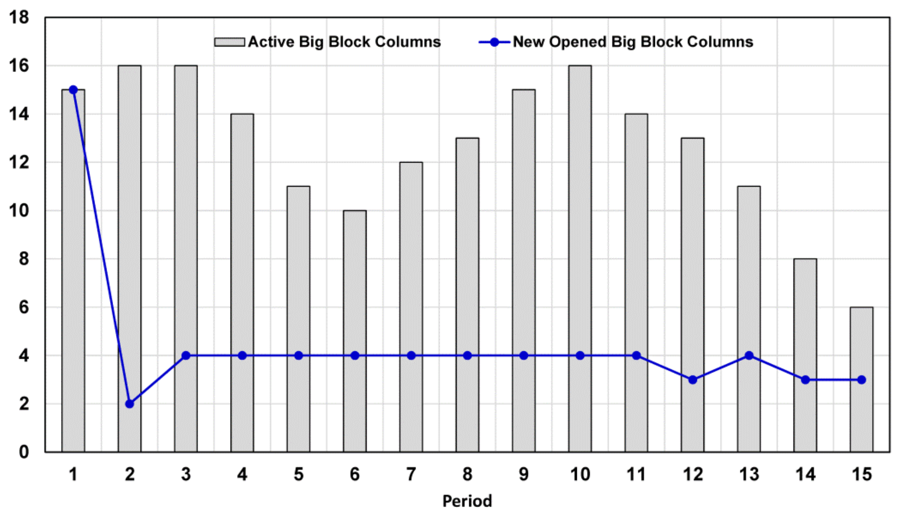


Figure 4.23. Number of active and new big blocks for each period at level 39

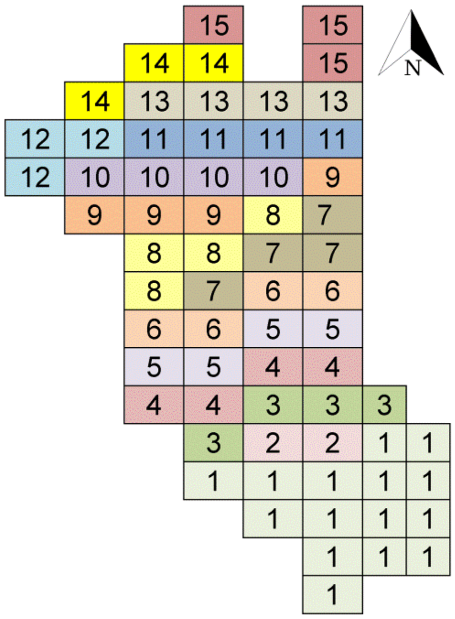


Figure 4.24. Starting extraction period of big-blocks at level 39 (numbers represent the starting period)

4.4.3 Comparison and analysis

In order to evaluate the risks involved in this block-cave mine dataset due to presence of grade uncertainty, the changes in NPV and tonnage should be investigated. Considering the deterministic values for grade, original block model or average-simulated block model result in one NPV or tonnage at the end, which cannot assess the effects of grade uncertainty and inspect the instability of the results. On the other hand, determining the range of NPV or tonnage and its maximum and minimum values considerably helps to understand the worse and the best case scenarios. Figure 4.25 shows the need of using simulation for risk analysis. It is obvious that optimizing one block model (original or average-simulated) will result in a single number. But when a number of block models are optimized, the obtained results show a range associated with the risk of project.

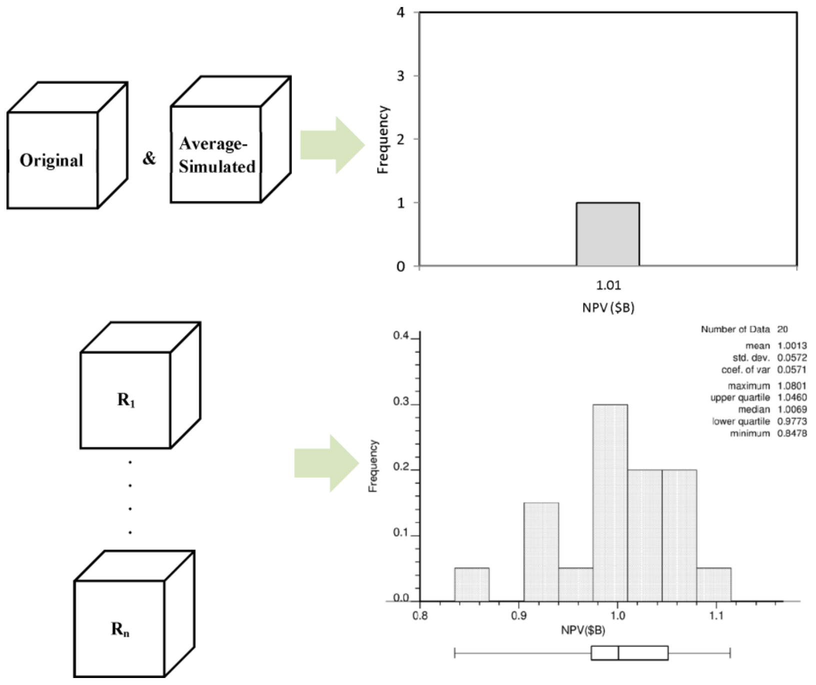


Figure 4.25. Schematic presentation of the NPV analysis

In the following section different scenarios have been studied to obstacle the risks. For all these cases unique scheduling parameters have been implemented to be able to compare the results. Table 4.6 shows the new scheduling parameters that are valid for all the block models.

4.4.3.1 Original and average-simulated block models

The best level for original block model was 38 and according to the new scheduling parameters the NPV was $1.01B and the ore tonnage that can be extracted was 37.45 Mt. The same procedure has been used for average-simulated block model with the best level of 39 and in this case the obtained NPV and ore tonnage were $0.84B and 27.33 Mt, respectively.

Table 4.6. Scheduling parameters for all the block models

Parameter	Value	Parameter	Value
T	15	$P(\$/tonne)$	6,000
$MCL\ (Mt)$	1.2	$SC\ (\$/tonne)$	0.5
$MCU\ (Mt)$	3	$MC\ (\$/tonne)$	10
$GL\ (\%)$	1.3	$PC\ (\$/tonne)$	16.1
$GU\ (\%)$	1.6	$\overline{N}_{NBBL,1}$	27
$ExtL\ (Kt)$	90	$\underline{N}_{NBBL,1}$	0
$ExtU\ (Kt)$	350	$\overline{N}_{NBBL,t}$	5
$i(\%)$	10	$\underline{N}_{NBBL,t}$	2
$R(\%)$	85	L	100,000,000

4.4.3.2 All realizations at their own best levels

In this case, the best level for each realization was determined and maximum NPV for each of them was specified. Figure 4.26 clearly shows the frequency of NPV for all the realizations at their own best levels. As it can be seen from the graph, the NPV changes between \$0.84B and \$1.08B and the mean was \$1B. Additionally, the NPV for original block model was within the lower and upper quartile.

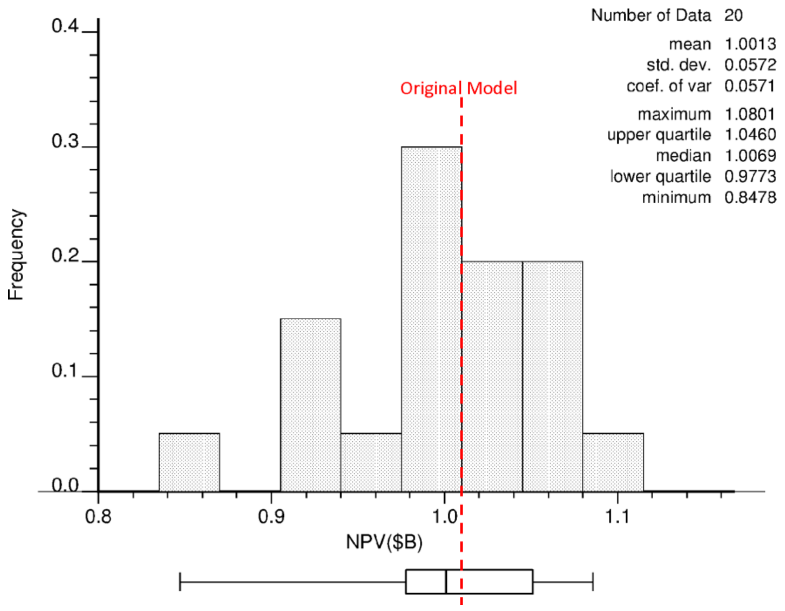


Figure 4.26. The NPV frequency for all the realizations at their own best levels

The same work has been done for the tonnage and Figure 4.27 shows the results. The minimum and maximum ore tonnages that can be extracted were 28.68 Mt and 38.64 Mt, respectively. The original block model tonnage value was within the upper and lower quantile as well.

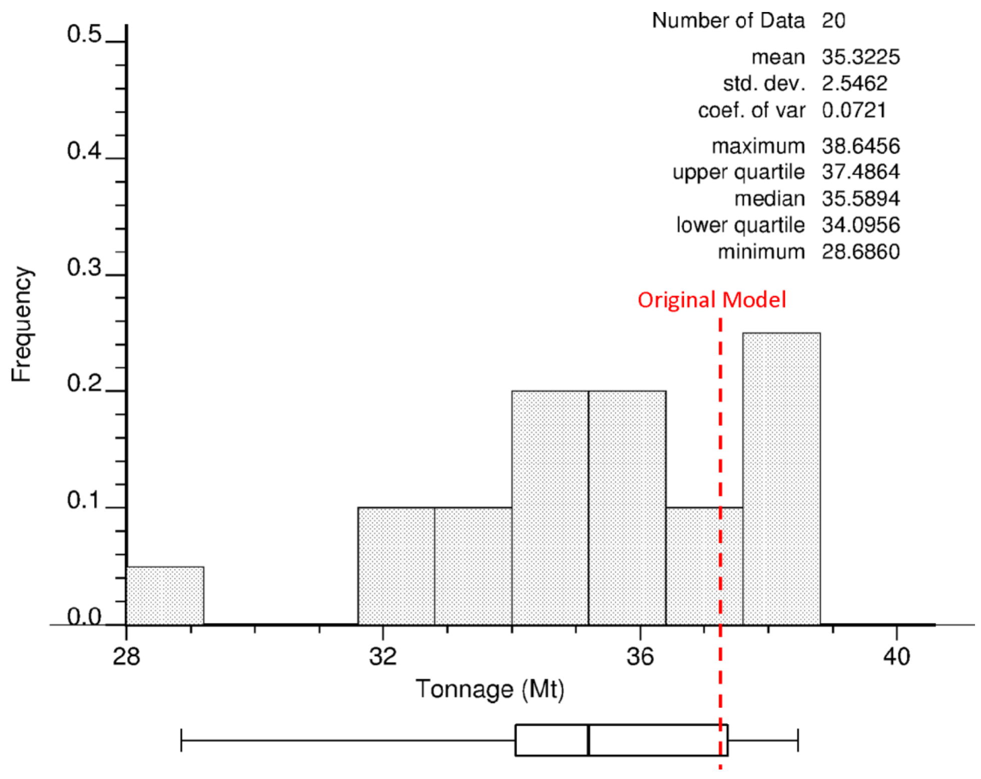


Figure 4.27. The tonnage frequency for all the realizations at their own best levels

4.4.3.3 All realizations at level 39

As described in section 4.3, level 39 was the best level of extraction for 40% of the realizations. In this case the tonnage and NPV changes for all the realizations at level 39 were examined. Figure 4.28 illustrates the frequency of NPV at level 39 for all the realizations. As it can be seen, the NPV changes were much less in compared to the previous scenario and the NPV of original block model was again within the upper and lower quartile. Figure 4.29 shows the tonnage analysis for this case. The ore tonnage changes between 33.1 Mt and 39.6 Mt and for the original block model it stands within this range.

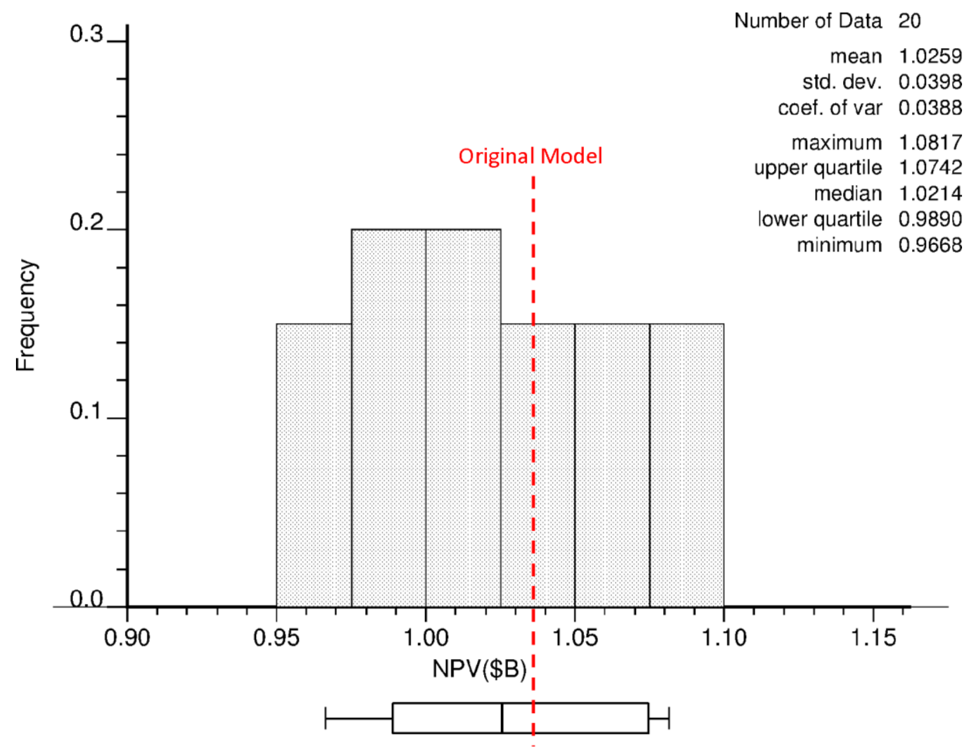


Figure 4.28. The NPV frequency for all the realizations at level 39

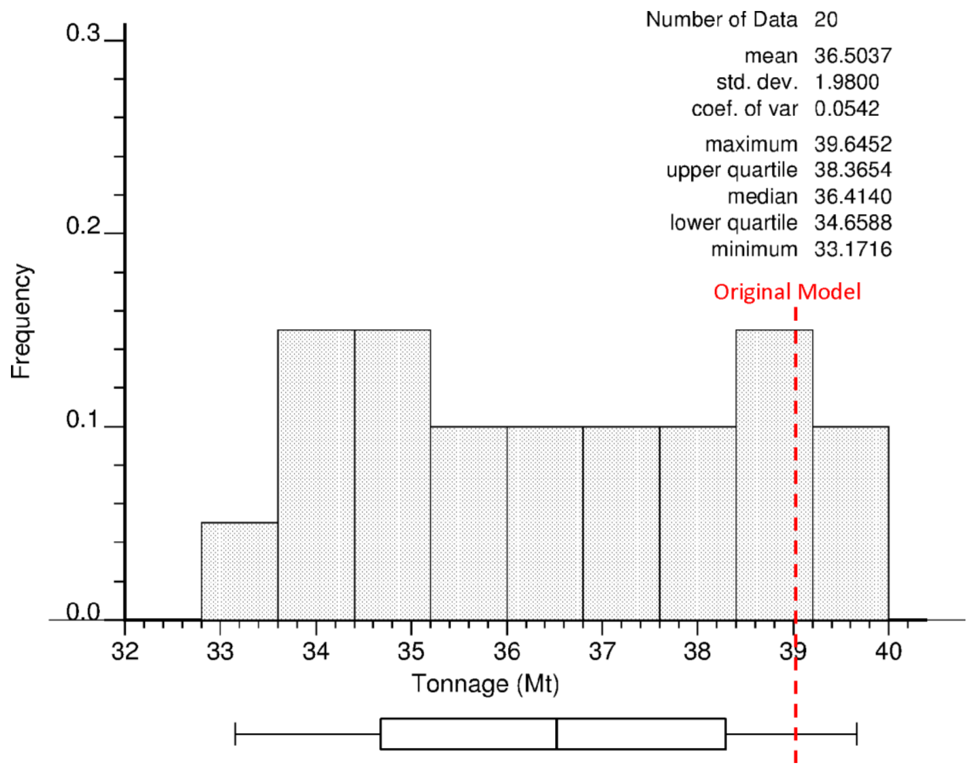


Figure 4.29. The tonnage frequency for all the realizations at level 39

4.4.3.4 Realizations with best level of 39

In this scenario, the evaluation of ore tonnage and NPV was just done on realizations which their best level of extraction was 39. Eight realizations out of 20 had the best level of 39 (40%). Figure 4.30 and Figure 4.31 show the NPV and ore tonnage frequency for just realizations with the best level of 39.

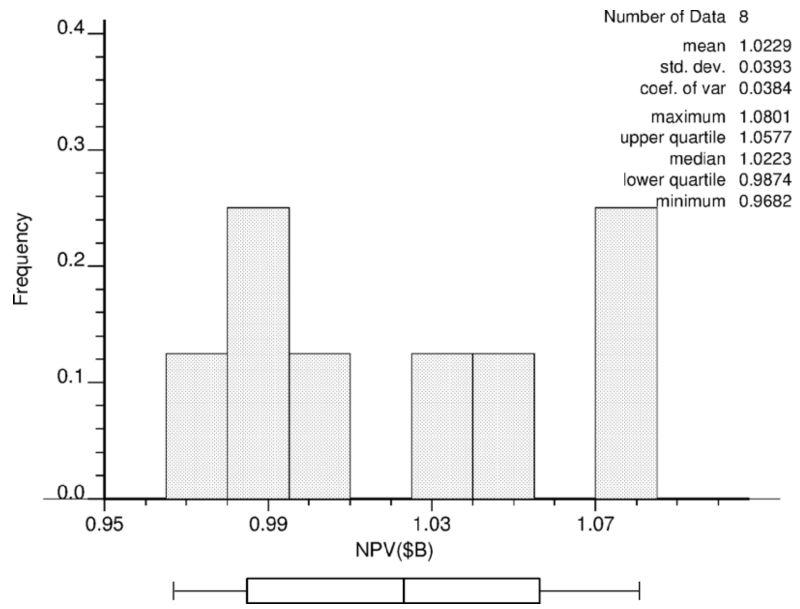


Figure 4.30. The NPV frequency for realizations with the best level of 39

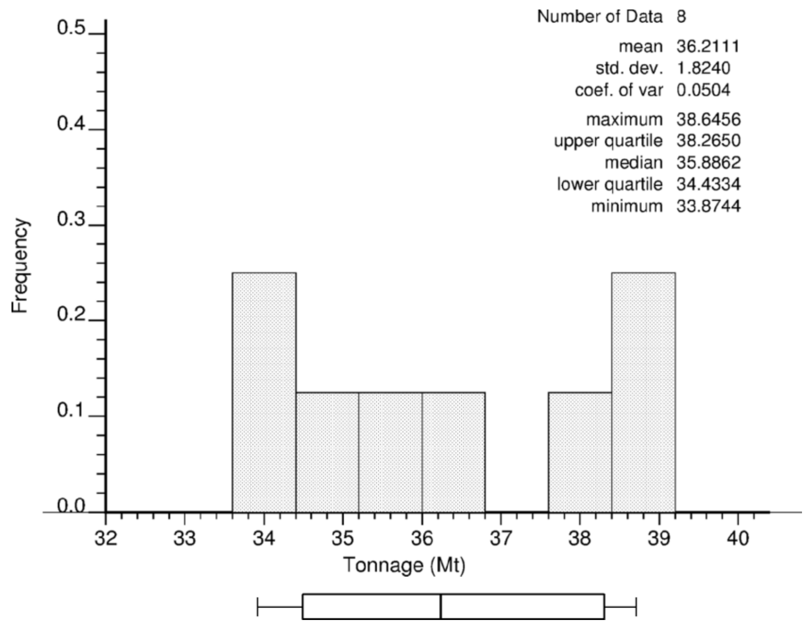


Figure 4.31 The tonnage frequency for realizations with best level of 39

4.4.3.5 NPV comparison at different levels

In Figure 4.32, the NPV for each realization at their own best levels has been demonstrated. Moreover, the NPV of original and average-simulated block models are highlighted. The green line shows the average NPV for each level.

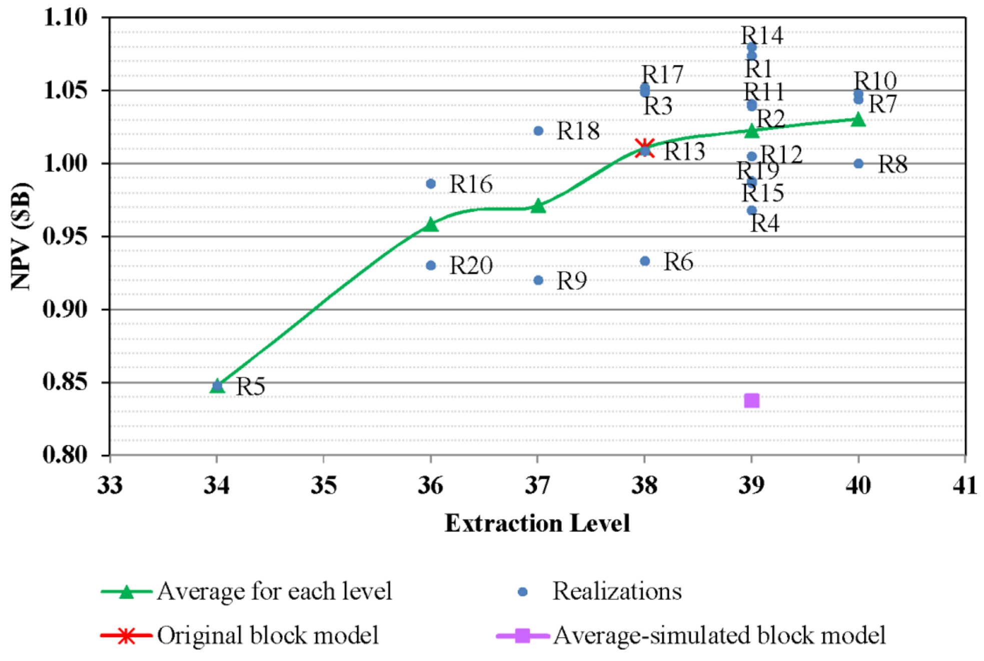


Figure 4.32. NPV comparison for different extraction levels

4.5 Summary and Conclusion

Geological uncertainty has been used in open-pit mining, but is less studied in underground mining, especially in block caving, where it is not so easy to revise production plans after caving has begun (Vargas et al., 2014). This methodology is able to find the best extraction horizon placement under grade uncertainty. Also, it is able to define an optimal production scheduling using mathematical programming and MILP formulation in MATLAB and solving it using IBM/CPLEX.

Two case studies were used in this chapter to verify the proposed geostatistical study and MILP model. To verify the production scheduling algorithm and mathematical formulation, at first, original block model was implemented and the results were discussed, then average-simulated block model which considered grade uncertainty was used.

The average-simulated block model was generated and level 39 with the highest discounted ore profit was selected as the level of extraction. Also for the original block model level 38 as the best level was determined. By utilizing the proposed MILP model, the highest NPV of $726M for 66 big-block columns for average-simulated block model and $925.1M for 90 big-block columns for original block model, at the EPGAP of 1%, were achieved. As the graphs show, the production amount is at its optimal level with highest grade for early periods and the maximum capacity is used during the extraction. Also it is clear from the precedence graph that the big blocks are extracted along the advancement direction trying to extract more big-blocks in first period.

Finally, the risks associated with the grade uncertainty were studied. A unique scheduling parameters was defined which was valid for all the block models. Various scenarios was presented to deal with the problem including: optimizing all the realizations at their own best levels, optimizing all the realizations at level 39 as the best level for all of them and optimizing those realizations that their best level was 39. The range of ore tonnage and NPV in these scenarios were analyzed to understand the best and the worst case. Considering all those scenarios, the NPV varied between $0.85B and $1.081B that means, at the worst case the NPV would be $0.85B and the NPV for the best case would be $1.081B. The same description is justifiable for tonnage. The minimum and maximum ore tonnages were 28.68 Mt and 39.64 Mt, respectively.

CHAPTER 5

SUMMARY, CONCLUSIONS AND RECOMMENDATIONS

Chapter 5 contains the summary and conclusion of this thesis. The contributions of this research are emphasized, as well as recommendation for future work in block-cave production scheduling.

5.1 Summary of Research

Production scheduling should be considered as an important issue due to the fact that mining industry is facing with lower grades and marginal reserves nowadays. Production scheduling should provide a mining sequence that takes into account the physical constraints of the mine as well as defining the tonnages and input grades to the processing plant throughout the mine life. The use of massive mining methods is increasing in major mining companies due to the economic issues of today's mining industry. Among the underground mining methods, block-cave mining could be considered as an appropriate alternative because of its low operation cost and high production rates (Pourrahimian, 2013). Optimal production schedule plays a critical role in underground mining as it is impossible to change the mining method once the cave is initiated. Software packages which use the simulation and heuristic methods generate feasible schedules rather than an optimal global solution. On the other hand, grade uncertainty has profound impact on optimality of the production schedule and it cannot be ignored due to the widely spaced drillholes in block caving (Koushavand, 2014). More studies have been done to address optimization of open pit production scheduling problems but fewer efforts have been made in underground mining. Some major weaknesses of the current production scheduling in block caving are: 1) restriction on solving large-scale problems; 2) considering stochastic variables as deterministic; 3) taking fewer geotechnical constraints into account in real-scale operations; 4) determination of extraction level; 5) trial-and-error to find the advancement direction.

This research is conducted to solve the shortcomings in dealing with large-scale problems, determining the extraction level, grade uncertainty, and integration of fewer geotechnical constraints through developing a mixed-integer linear programming (MILP) model and stochastic sequential simulation.

Two main goals of this research are to: 1) implement and verify stochastic sequential simulation to take grade uncertainty into account; 2) develop, implement, and verify a MILP model to optimize block-cave long-term production scheduling. The objective function is to maximize NPV with respect to technical and operational constraints. These constraints are the draw rate, mining capacity, minimum required mining footprint, grade blending, precedence, number of new big-blocks, continuous extraction, and reserves.

MATLAB (Math Works Inc, 2015) programming platform was used to create the objective function and constraints. IBM/CPLEX (IBM, 2015) which is a solver for large-scale optimization problems, was used in this research. It uses branch-and-bound algorithm to solve the problem.

The best level of extraction was determined for original block model, average-simulated block model and all the realizations according to the maximum discounted profit. The most frequent level from the realizations was determined. According to the best level of average-simulated block model and most frequent level from the realizations, the best level was specified. The actual outline of the ore-body at the best level was specified and then big-blocks were created inside this outline based on the minimum required mining footprint. The MILP formulations were applied on both the original and average-simulated block models containing 90 and 66 big-blocks, respectively, over 15 periods and scheduling parameters were defined. The results showed that all the considered constraints had been satisfied and the MILP model worked properly. After making sure that the MILP model works well for both the models. To solve the two models, the EPGAP of 1% was set. The formulations yielded a NPV of \$925.1M and \$726.5M for the original and average-simulated block models, respectively. Finally, in order to assess the involved risks due to grade uncertainty, the changes in NPV and tonnage have been examined.

Figure 5.1 shows a summary of the workflow for completing case study based on the proposed algorithms and model.

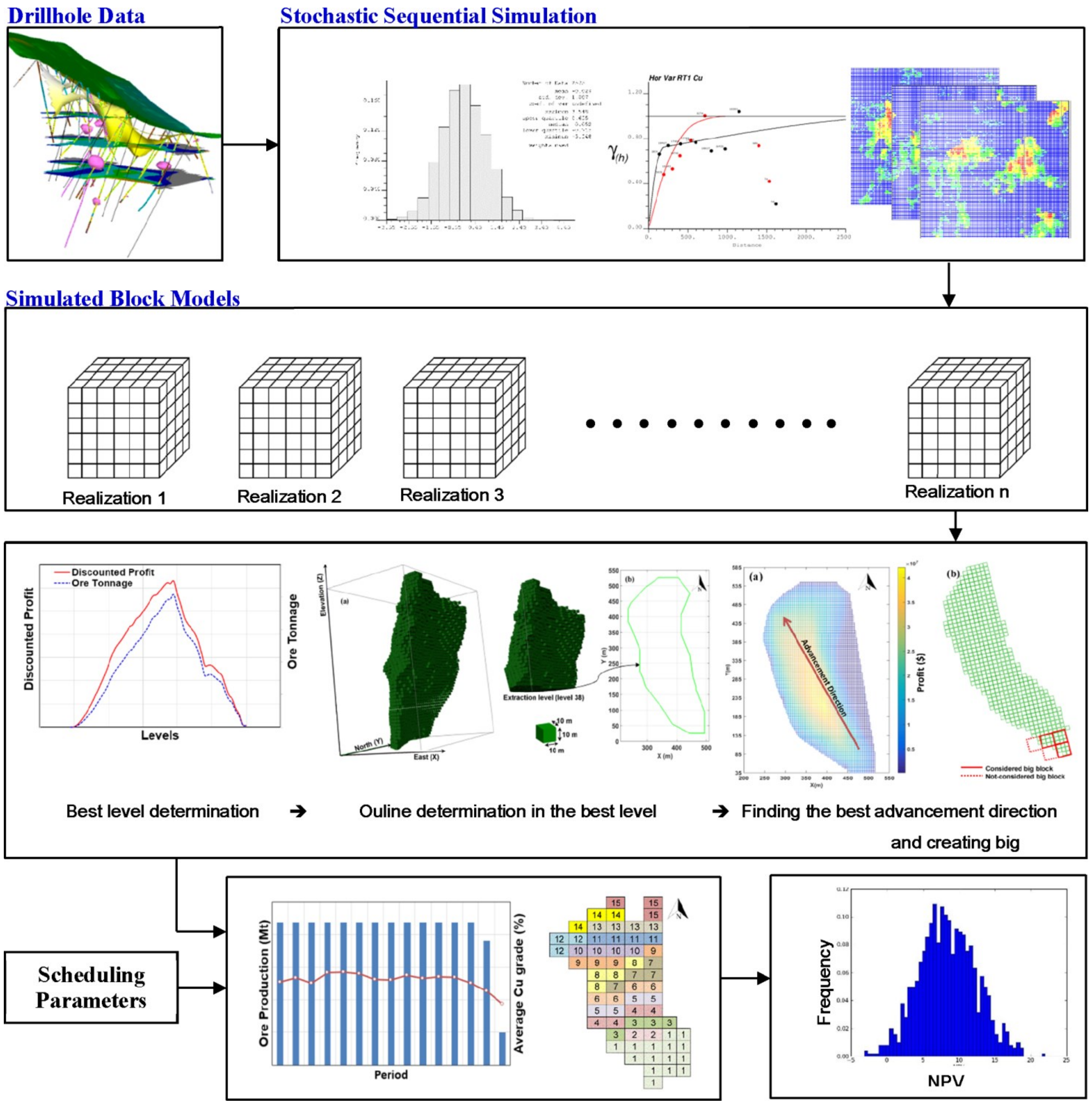


Figure 5.1. Summary of the research methods

5.2 Conclusions

All the research objectives outlined in Chapter 1, have been achieved. The following conclusions were obtained from stochastic sequential simulation and mathematical programming:

1. Implementing stochastic sequential simulation contributes significantly to decrease the effect of grade uncertainty on the optimality of the project.
2. The presented method is able to find the best level of extraction to start the mining operation based on the maximum discounted profit.
3. The proposed MILP model maximizes the NPV of the block-cave mine while enforcing the model to satisfy constraints.
4. The MILP model is able to generate a production schedule for large-scale block-caving operations through creating big-block columns.
5. The proposed methodology is verified in terms of both feasibility and optimality on two case-studies and run with an optimality gap of 1% in which two schedules with maximum NPV were generated.

5.3 Contributions of the Research

This research has implemented stochastic sequential simulation and developed mathematical formulations, which contributes notably to generate an optimal production schedule for the block-cave mine under grade uncertainty. The following constitute the main contributions of this research.

1. Combination of geostatistical simulation with MILP model in the context of block-cave mine planning.
2. Studying the effect of grade uncertainty on block-cave mine production scheduled, and consequently on the NPV of the mining projects.
3. Determination of best level to initiate the extraction based on maximum discounted profit and grade uncertainty.
4. Creation of big-block columns based on the minimum required mining footprint enormously help to decrease the number of variables and enable mine planners to solve large-scale production scheduling problems.
5. Maximizing NPV through the MILP model and subjected to a set of practical constraints including: mining capacity, grade blending, mining precedence, draw rate, number of new big-blocks, and reserves.

6. Introducing a methodology to determine the predecessor big-blocks and its implementation as prototype software with a graphical user interface.
7. Development of a prototype open-source software application with the graphical user interface called Block Cave Footprint Optimizer (BCFO). The prototype software helps transfer knowledge and optimization technology developed in this thesis to practitioners and end-users in the field of block-cave production scheduling.

5.4 Recommendations for Future Research

In spite of the fact that the developed model and implemented algorithms in this thesis have presented new methods and formulations for block-cave production scheduling problems, but there are still some limitations in production scheduling of block-cave mines that should be eliminated through using mathematical programming models. The following recommendations could significantly improve the block-cave mine production scheduling problems:

1. More uncertain attributes other than grade should be added to the optimization problem. It means other economic variables such as cost and price are not deterministic in the future and there is a need to re-optimize the production schedules. To overcome this shortcoming, the MILP model should be extended to take stochastic variables into account during optimization.
2. One of the assumptions in the proposed model was no material mixing between blocks. In future research the dilution should be considered during optimization.
3. There can be other mining parameters other than assumed constraints in this thesis that should be defined in the future works.
4. Generating more number of realizations can be helpful in interpretation and analysis of data.

BIBLIOGRAPHY

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

[2] Alford, C., Brazil, M., and Lee, D. (2007). Optimisation in Underground Mining. in *Handbook Of Operations Research In Natural Resources*, Vol. 99, *International Series In Operations Research amp; Mana*, A. Weintraub, C. Romero, T. Bjørndal, R. Epstein, and J. Miranda, Eds., Springer US, pp. 561-577.

[3] Alonso-Ayuso, A., Carvallo, F., Escudero, L. F., Guignard, M., Pi, J., Puranmalka, R., and Weintraub, A. (2014). Medium range optimization of copper extraction planning under uncertainty in future copper prices. *European Journal of Operational Research*, *233* (3), 711-726.

[4] Asad, M. W. A. and Dimitrakopoulos, R. (2013). Implementing a parametric maximum flow algorithm for optimal open pit mine design under uncertain supply and demand. *Journal of the Operational Research Society*, *64* (2), 185-197.

[5] Askari-Nasab, H. and Awuah-Offei, K. (2009). Open Pit Optimisation using Discounted Economic Block Values. *Transactions of the Institution of Mining and Metallurgy, Section A, Mining Industry*, *118* (1), 1-12.

[6] Askari-Nasab, H., Pourrahimian, Y., Ben-Awuah, E., and Kalantari, S. (2011). Mixed integer linear programming formulations for open pit production scheduling. *Journal of Mining Science, © Springer, New York, NY 10013-1578, United States,*, *47* (3), 338-359.

[7] Bley, A., Boland, N., Fricke, C., and Froyland, G. (2010). A Strengthened Formulation and Cutting Planes for the Open Pit Mine Production Scheduling Problem. *Computers & Operations Research*, *37* (9), 1641-1647.

[8] Boland, N., Dumitrescu, I., Froyland, G., and Gleixner, A. M. (2009). LP-based disaggregation approaches to solving the open pit mining production scheduling problem with block processing selectivity. *Computers and Operations Research*, *36* (4), 1064-1089.

[9] Brannon, C. A., Carlson, G. K., and Casten, T. P. (2011). Block Caving. in *SME mining engineering handbook [electronic resource]* Vol. II, P. Darling, Ed. 3 ed, [Englewood, Colo.] : Society for Mining, Metallurgy, and Exploration, c2011., pp. 1437-1451.

[10] Brazil, M., Lee, D. H., Rubinstein, D. A., Weng, J. F., and Wormald, N. C. (2000). *Network optimization of underground mine design*. in Proceedings of The Australasian Institute of Mining and Metallurgy, pp. 57-65.

[11] Brazil, M., Lee, D. H., Van Leuven, M., Rubinstein, J. H., Thomas, D. A., and Wormald, N. C. (2003). Optimising declines in underground mines. *Mining Technology*, *112* (3), 164-170.

[12] Caccetta, L. and Hill, S. P. (2003). An Application of branch and cut to open pit mine scheduling. *Journal of Global Optimization*, *27* (2), 349-365.

[13] Carpentier, S., Gamache, M., and Dimitrakopoulos, R. (2016). Underground long-term mine production scheduling with integrated geological risk management. *Mining Technology*, *125* (2), 93-102.

[14] Chanda, E. C. K. (1990). An application of integer programming and simulation to production planning for a stratiform ore body. *Mining Science and Technology*, *11* (2), 165-172.

[15] Chanda, E. K. C. and Dagdelen, K. (1995). Optimal blending of mine production using goal programming and interactive graphics systems. *International Journal of Surface Mining, Reclamation and Environment*, *9* (4), 203-208.

[16] Chicoisne, R., Espinoza, D., Goycoolea, M., Moreno, E., and Rubio, E. (2012). A New Algorithm for the Open-Pit Mine Production Scheduling Problem. *Operations Research*, *60* (3), 517-528.

[17] Cullenbine, C., Wood, R., and Newman, A. (2011). A sliding time window heuristic for open pit mine block sequencing. *Optimization Letters*, *5* (3), 365-377.

[18] Dagdelen, K. and Johnson, T. B. (1986). *Optimum open pit mine production scheduling by lagrangian parameterization*. in Proceedings of 19th Application of Computers and Operations Research in the Mineral Industry Proceedings, Society of Mining Engineers of the American Institute of Mining, Metallurgical, and Petroleum Engineers, Inc., Littleton, Colorado, pp. 127-142.

[19] Deutsch, C. V. and Journel, A. G. (1998). *GSLIB : geostatistical software library and user's guide.* Vol. 2, Oxford University Press, New York,369.

[20] Diering, T. (2004). Combining Long Term Scheduling and Daily Draw Control for Block Cave Mines. in *Massmin*. Santiago, Chile, pp. 486-490.

[21] Diering, T. (2004). *Computational considerations for production scheduling of block cave mines*. in Proceedings of MassMin 2004, Santiago, Chile, pp. 135-140.

[22] Diering, T. (2012). Quadratic Programming applications to block cave scheduling and cave management. in *Massmin 2012*. Sudbury, Canada, pp. 1-8.

[23] Diering, T., Richter, O., and Villa, D. (2008). Block cave production scheduling using PCBC. in *MassMin 2008*. Luleå, Sweden., pp. 17.

[24] Dimitrakopoulos, R. (1998). Conditional simulation algorithms for modelling orebody uncertainty in open pit optimisation. *International Journal of Surface Mining, Reclamation and Environment*, *12* (4), 173-179.

[25] Dimitrakopoulos, R. and Ramazan, S. (2008). Stochastic integer programming for optimising long term production schedules of open pit mines: methods, application and value of stochastic solutions. *Mining Technology*, *117* (4), 155-160.

[26] Dowd, P. A. (1994). Risk assessment in reserve estimation and open-pit planning. *Trans. Instn. Min. Metall.*, *103* A148 - A154.

[27] Epstein, R., Goic, M., Weintraub, A., Catalán, J., Santibáñez, P., Urrutia, R., Cancino, R., Gaete, S., Aguayo, A., and Caro, F. (2012). Optimizing Long-Term Production Plans in Underground and Open-Pit Copper Mines. *Operations Research*, *60* (1), 4-17.

[28] Flores, G. (2014). *Future Challenges and Why Cave Mining Must Change*. in Proceedings of Caving 2014, Universidad de Chile, Santiago, Chile.

[29] Gerling, R. and Helms, W. (1986). *A model for long range scheduling of the mining sequence for flat lying orebodies*. in Proceedings of 19 Application of Computers and Operations Research in the Mineral Industry, Society of Mining Engineers of the America Institute of Mining, Metallurgical, and Petroleum Engineers, Inc., New York, pp. 333-342.

[30] Gershon, M. E. (1983). Optimal Mine Production Scheduling: Evaluation of Large Scale Mathematical Programming Approaches. *International Journal of Mining Engineering*, *1* 315-329.

[31] Godoy, M. and Dimitrakopoulos, R. (2003). Managing risk and waste mining in long-term production scheduling of open-pit mines. *SME Ann Meet Exhib*, *316* 43-50.

[32] Goovaerts, P. (1997). *Geostatistics for natural resources evaluation.* Oxford University Press, New York, 409-420 483.

[33] Grieco, N. and Dimitrakopoulos, R. (2007). Managing grade risk stope design optimisation: Probabilistic mathematical programming model and application in sublevel stoping. *IMM Transactions Mining Technology*, *116* (2), 49-57.

[34] Guest, A. R., Van Hout, G. J., and Von Johannides, A. (2000). *An Application of Linear Programming for Block Cave Draw Control*. in Proceedings of MassMin, The Australasian Institute of Mining and Metallurgy, Brisbane, Australia, pp. 461-468.

[35] Hannweg, L. A. and Van Hout, G. J. (2001). *Draw control at Koffiefontein Mine*. in Proceedings of 6th International Symposium on Mine Mechanization and Automation pp. 97-102.

[36] Hanson, B. D. and Selim, A. A. (1975). Probabilistic simulation of underground production systems. *Transactions of the Institution of Mining and Metallurgy*, *258* (3), 19-24.

[37] Holmstrom, K. (2011). *TOMLAB/CPLEX, ver. 11.2. Ver. Pullman.* WA, USA: Tomlab Optimization.

[38] Horst, R. and Hoang, T. (1996). *Global optimization: deterministic approaches.* Springer, New York, 3rd ed,727.

[39] Huang, Y. and Kumar, U. (1994). Optimizing the number of load-haul-dump machines in a Swedish mine by using Queuing theory- A case study. . *International Journal of Surface Mining, Reclamation and Environment*, *8* (3), 171-174.

[40] IBM (2015). IBM ILOG CPLEX 12.6.2. Ver.

[41] Johnson, T. B. (1969). Optimum Open Pit Mine Production Scheduling. in *In A. Weiss (Ed.), A Decade of Digital Computing in the Mineral Industry (pp. 539-562)*. New York: AIME.

[42] Julin, D. E. (1992). Block Caving. in *Mining Engineering Handbook*, Vol. 1, H. L. Hartman, Ed. 2 ed, Littleton, Colorado, SME (Society for Mining, Metallurgy, and Exploration, Inc.), pp. 1815-1836.

[43] Khodayari, F. and Pourrahimian, Y. (2015). Determination of development precedence for drawpoints in block-cave mining. in *5th International Symposium Mineral Resources and Mine Development (AIMS 2015)* Aachen, Germany, pp. 383-391.

[44] Kim, Y. C., Cai, W., and Meyer, W. L. (1988). Comparison of microcomputer-based optimum pit limit design algorithms Algorithms. *AIME Transactions*, *248* 1827-1830.

[45] Kim, Y. C. and Zhao, Y. (1994). Optimum open pit production sequencing - The current state of the art. in *1 st Regional Symposium on the application of Computers in the Mineral Industries (APCOM), Proc.*, pp. 29-39.

[46] Koushavand, B. (2014). Long-Term Mine Planning in Presence of Grade Uncertainty. Ph.D. Thesis, University of Alberta (Canada), Canada, Pages 234.

[47] Koushavand, B. and Askari-Nasab, H. (2009). Transfer of Geological Uncertainty into Mine Planning. in *Mine Planning and Equipment Selection (MPES) CD-ROM*. Banff, Alberta, Canada, pp. 462-476.

[48] Kuchta, M., Newman, A., and Topal, E. (2004). Implementing a Production Schedule at LKAB's Kiruna Mine. *Interfaces*, *Vol. 34, No. 2* 124–134.

[49] Kumral, M. (2012). Production planning of mines: Optimisation of block sequencing and destination. *International Journal of Mining, Reclamation and Environment*, *26* (2), 93-103.

[50] Kumral, M. and Dowd, P. A. (2005). A Simulated Annealing Approach to Mine Production Scheduling. *The Journal of the Operational Research Society*, *56* (8), 922-930.

[51] Lamghari, A. and Dimitrakopoulos, R. (2012). A diversified Tabu search approach for open-pit mine production scheduling problem with metal uncertainty. *European Journal of Operational Research*, *222* (3), 642-652.

[52] Lamghari, A., Dimitrakopoulos, R., and Ferland, J. (2013). A variable neighbourhood decent algorithm for the open-pit mine production scheduling problem with metal uncertainty *Journal of the Operational Research Society 65* (9), 1305-1314.

[53] Laubscher, D. A. (2003). Cave Mining Handbook. De Beers, Johannesburg. pp. 138.

[54] Laubscher, D. H. (1994). Cave mining-the state of the art. *The Journal of The South African Institute of Mining and Metallurgy*, 279-293.

[55] Leite, A. and Dimitrakopoulos, R. (2007). Stochastic optimization model for open-pit mine planning: application and risk analysis at a copper deposit. *IMM Transactions Mining Technology*, *116* (3), 109-118.

[56] Lerchs, H. and Grossmann, I. (1965). Optimum design of open-pit mines. *Canadian Mining Metallurgical Bull*, *58* 17-24.

[57] Maleki, M. and Emery, X. (2015). Joint simulation of grade and rock type in a stratabound copper deposit *Mathematical Geosciences 47* (4), 471-495.

[58] Maxwell, A. S. (1978). Underground production mining at Chingola, with emphasis on a computerized ore-reserve calculation, depletion, and prediction system. in *11th Commonwealth Mining and Metallurgical Congress*, T. I. o. M. a. Metallurgy, Ed. Hong Kong, pp. 507-524.

[59] McLennan, J. A. and Deutsch, C. V. (2004). Conditional Non-Bias of Geostatistical Simulation for Estimation of Recoverable Reserves. *CIM Bulliten*, *97* (No. 1080).

[60] Montiel, L., Dimitrakopoulos, R., and Kawahata, K. (2015). Globally optimising open-pit and underground mining operations under geological uncertainty. *Mining Technology*, *00* (0), 1-13.

[61] Muge, F. H., Santos, N., Vierira, J. L., and Cortez, L. (1992). *Dynamic programming in mine planning and production scheduling*. in Proceedings of 23rd Application of Computers and Operations Research in the Mineral Industry, Society of mining Engineering of the American Institute of Mining, Metallurgical, and Petroleum Engineers, Inc., Littleton, Colorado, USA.

[62] Nehring, M., Topal, E., Kizil, M., and Knights, P. (2012). Integrated short- and medium-term underground mine production scheduling. *Journal of the Southern African Institute of Mining and Metallurgy*, *112* 365-378.

[63] Newman, A. M., Rubio, E., Caro, R., Weintraub, A., and Eurek, K. (2010). A Review of Operations Research in Mine Planning. *Interfaces*, *40* (3), 222-245.

[64] Osanloo, M., Gholamnejad, J., and Karimi, B. (2008). Long-term open pit mine production planning: a review of models and algorithms. *International Journal of Mining, Reclamation and Environment*, *22* (1), 3-35.

[65] Parkinson, A. (2012). Essays on Sequence Optimization in Block Cave Mining and Inventory Policies with Two Delivery Sizes.Thesis, The University Of British Columbia, The University Of British Columbia, Pages 199.

[66] Pourrahimian, Y. (2013). Mathematical programming for sequence optimization in block cave mining. Ph.D. Thesis, University of Alberta (Canada), Ann Arbor, Pages 259.

[67] Pourrahimian, Y. and Askari-Nasab, H. (2014). An application of mathematical programming to determine the best height of draw in block-cave sequence optimization. *Mining Technology (Trans. Inst. Min. Metall. A)*, *123* (3), 162-172.

[68] Pourrahimian, Y., Askari-Nasab, H., and Tannant, D. (2012). Mixed-Integer Linear Programming formulation for block-cave sequence optimisation. *Int. J. Mining and Mineral Engineering*, *4, No. 1* 26-49.

[69] Pourrahimian, Y., Askari-Nasab, H., and Tannant, D. (2013). A multi-step approach for block-cave production scheduling optimization. *International Journal of Mining Science and Technology*, *23* (5), 739-750.

[70] Rahal, D. (2008). Draw Control in Block Caving Using Mixed Integer Linear Programming.Thesis, The University of Queensland, Pages 342.

[71] Rahal, D., Dudley, J., and Hout, G. v. (2008). *Developing an optimised production forecast at Northparkes E48 mine using MILP*. in Proceedings of 5th International Conference and Exhibition on Mass Mining, Luleå Sweden, pp. 227-236.

[72] Rahal, D., Smith, M., Van Hout, G., and Von Johannides, A. (2003). The use of mixed integer linear programming for long-term scheduling in block caving mines. *Application of Computers and Operations Research ill the Minerals Industries, South African Institute of Mining and Metallurgy*, 123-132.

[73] Ramazan, S. (2007). The New Fundamental Tree Algorithm for Production Scheduling of Open Pit Mines. *European Journal of Operational Research*, *177* (2), 1153-1166.

[74] Ramazan, S. and Dimitrakopoulos, R. (2004). Traditional and New MIP Models for Production Scheduling With In-Situ Grade Variability. *International Journal of Surface Mining, Reclamation and Environment*, *18* (2), 85-98.

[75] Ramazan, S. and Dimitrakopoulos, R. (2013). Production scheduling with uncertain supply: a new solution to the open-pit mining problem *Optimization and Engineering*, *14* (2), 361-380.

[76] Ravenscroft, P. (1992). Risk analysis for mine scheduling by conditional simulation. *Canadian Mining and Metallurgical Bulletin*.

[77] Rubio, E. (2002). Long term planning of block caving operations using mathematical programming tools.Thesis, The University of British Columbia, Pages 126.

[78] Rubio, E. and Diering, T. (2004). *block cave production planning using operation research tool.* in Proceedings of Massmin 2004, Gemcom, Santiago, Chile, pp. 141-149.

[79] Russell, F., M. (1987). *Application of a PC-based network analysis program to mine scheduling.* in Proceedings of 20 Internationa Symposium on the Application of Computers and Mathematics in the Mineral Industries, SAIMM, Johannesburg, South Africa, pp. 123-1312.

[80] Sabour, S. A. and Dimitrakopoulos, R. (2011). Incorporating geological and market uncertainties and operational flexibility into open pit mine design. *Journal of Mining Science*, *47* (2), 191-201.

[81] Samanta, B., Bhattacherjee, A., and Ganguli, R. (2005). A genetic algorithms approach for grade control planning in a bauxite deposit. in *Application of Computers and Operations Research in the Mineral Industry*: Taylor & Francis., pp. 337-342.

[82] Sattarvand, J. and Niemann-Delius, C. (2008). Perspective of Metaheuristic Optimization Methods in Open Pit Production Planning. in *Paper presented at the World Mining Congress*. katowice, Poland., pp. 143-154.

[83] Sattarvand, J. and Niemann-Delius, C. (2013). A New Metaheuristic Algorithm for Long-Term Open-Pit Production Planning. *Archives of Mining Sciences*, *58* (1), 107-118.

[84] Sayadi, A. R., Fathianpour, N., and Mousavi, A. A. (2011). Open pit optimization in 3D using a new artificial neural network. *Archives of Mining Sciences*, *56* (3), 389-403.

[85] Sherer, H. E. and Gentry, D. W. (1982). *Use of an interactive dynamic program system as an aid to mine valuation.* . in Proceedings of 17 Application of Computers and Operations Research in the Mineral Industry, Society of Mining Engineers of the Amwrica Institute of Mining, Metallurgical, and Petroleum Engineers, Inc., New York, pp. 161-169.

[86] Smoljanovic, M., Rubio, E., and Morales, N. (2011). *Panel Caving Scheduling Under Precedence Constraints Considering Mining System*. in Proceedings of 35th APCOM Symposium, Wollongong, NSW, Australia, pp. 407-417.

[87] Song, X. (1989). *Caving process simulation and optimal mining sequence at Tong Kuang Yu mine, China*. in Proceedings of pp. 386-392.

[88] Su, Y. L. (1986). *A computer solution to the Queuing problem at a mine-mill interface*. in Proceedings of 19th Application of Computers and Operations Research in the Mineral Industry, Society of mining Engineering of the American Institute of Mining, Metallurgical, and Petroleum Engineers, Inc., Littleton, Colorado. Pennsylvania State University, USA, pp. 394-400.

[89] Tabesh, M. (2015). Aggregation and Mathematical Programming for Long-Term Open Pit Production Planning.Thesis, University of Alberta, Edmonton, Alberta, Pages 319.

[90] Tobie, R. L. and Julin, D. E. (1998). Block Caving: General Description. in *Techniques in Underground Mining - Selections from Underground Mining Methods Handbook*, Society for Mining, Metallurgy, and Exploration (SME).

[91] Topal, E. (2008). Early start and latestart algorithms to improve the solution time for long-term underground mine production scheduling. . *Journal of The South African Institute of Mining and Metallurgy*, *108* (2), 99-107.

[92] Vargas, E., Morales, N., and Emery, X. (2014). *Footprint and economic envelope calculation for block/panel caving mines under geological uncertainty*. in Proceedings of Caving 2014, Santiago, Chile, pp. 449-456.

[93] Weintraub, A., Pereira, M., and Schultz, X. (2008). A Priori and A Posteriori Aggregation Procedures to Reduce Model Size in MIP Mine Planning Models. *Electronic Notes in Discrete Mathematics 30* 297–302.

[94] Winkler, B. M. (1996). *Using MILP to Optimize Period Fix Costs in Complex Mine Sequencing and Scheduling Problems*. in Proceedings of Twenty sixth proceedings of the application of computers and operations research in the minerals industry (APCOM), pp. 441-446.

[95] Wolsey, L. A. (1998). *Integer programming.* J. Wiley, New York,264.

[96] Wooller, R. (1992). Production scheduling system. *Transactions of the Institution of Mining and Metallurgy, Section A, Mining Industry*, *101* A47-A54.

APPENDIX A

BEST LEVEL OF EXTRACTION

After simulation the best level of extraction should be found. Two steps should be followed: (1) import the block model, and (2) find the best level.

```
function Import()
[filename] = uigetfile('.txt');
fileID = fopen(filename);
Data = fscanf(fileID,'%f %f %f %f %f %f %f %f %f %f',[10 Inf])';
InputData.Levels = Data(:,1);
InputData.Rows = Data(:,2);
InputData.Columns = Data(:,3);
InputData.X = Data(:,4);
InputData.Y = Data(:,5);
InputData.Z = Data(:,6);
InputData.Rock = Data(:,7);
InputData.CU = Data(:,8);
InputData.Tonnage = Data(:,9);
InputData.Density = Data(:,10);
save('Results/InputData','InputData');
h = msgbox('All required data was imported.');
end
```

```
function MinHOD21
load('Results/InputData.mat')
% Level data matrix
nLevel = max(InputData.Levels);
LevelData = [(1:nLevel)',zeros(nLevel,6)];
%***************************************************
TonnageTest = zeros(nLevel,1);
for level = 1:nLevel
    % Find amout of ore in each level
    LevelIDs = find(InputData.Levels == level);
    RockData_help = InputData.Rock(LevelIDs);
    TonData_help = InputData.Tonnage(LevelIDs);
    GradeData_help = InputData.CU(LevelIDs);
    % Ore = 12;
    % Waste = 13;
    ID_Ore = find(RockData_help == 12);
    TonOre = TonData_help(ID_Ore);       % vertical vector
    GradeOre = GradeData_help(ID_Ore);  % vertical vector
    Level_Ore = TonOre .* (GradeOre./100);      % vertical vector -
Determines the amount of ore in each level
    % ---------------------------------------------------
    % Economic Data
    % ---------------------------------------------------
    Price = 6000         ;% (US$/t)
    SellingCost = 0.50   ;% (US$/t)
    MineCost = 10          ;% (US$/t)
    PlantCost = 16.1       ;% (US$/t)
    Rec = 0.85             ;% (%)
    LevelData (level,2) = sum(Level_Ore);
    Revenue = Level_Ore .* Rec .* ((Price-
SellingCost)*ones(size(Level_Ore,1),1));
    Cost = TonOre .*
((MineCost+PlantCost)*ones(size(Level_Ore,1),1));
    LevelData (level,3) = sum(Revenue - Cost)/1000000000;
    TonnageTest(level,1) = sum(TonOre);
end
%---------------------Plot the tonnage & profit of each level
Ax = 1:nLevel;
[ax,p1,p2] =
plotyy(Ax,LevelData(:,2),Ax,LevelData(:,3),'semilogy','plot');
ylabel(ax(1),'Ore Tonnage (kt)') % label left y-axis
ylabel(ax(2),'Profit (b$)') % label right y-axis
xlabel(ax(2),'Levels') % label x-axis
p1.LineStyle = '--';
p1.LineWidth = 2;
p2.LineWidth = 2;
%-----------------------MHOD (First Method)-----------------------
ID_Data = zeros(4,nLevel);
for lLoop = 1:nLevel
    stopBar= progressbar(lLoop / nLevel,0);
    if (stopBar) break; end

    LevelIDs = find(InputData.Levels == lLoop);

    RockData_help = InputData.Rock(LevelIDs);
    ID_Ore = find(RockData_help == 12);

    XI_Data = InputData.Rows(LevelIDs);
    YI_Data = InputData.Columns(LevelIDs);

    ID_X = XI_Data(ID_Ore);
    ID_Y = YI_Data(ID_Ore);

    if isempty(ID_X) == 1

        ID_Data(1,lLoop) = 1000000;
        ID_Data(2,lLoop) = 0;
    else
        ID_Data(1,lLoop) = min(ID_X);
        ID_Data(2,lLoop) = max(ID_X);
    end

    if isempty(ID_Y) == 1

        ID_Data(3,lLoop) = 1000000;
        ID_Data(4,lLoop) = 0;
    else
    ID_Data(3,lLoop) = min(ID_Y);
    ID_Data(4,lLoop) = max(ID_Y);
    end
```

```
end
close
MinAll_X_Id = min(ID_Data(1,:));
MinAll_Y_Id = min(ID_Data(3,:));
MaxAll_X_Id = max(ID_Data(2,:));
MaxAll_Y_Id = max(ID_Data(4,:));
%------------------------------------------
%Calculating Economic Values for each block
Revenue_All = InputData.Tonnage .* Rec .* ((InputData.CU)./100)
.*((Price-SellingCost)*ones(size(nLevel,1),1));
Cost_All = InputData.Tonnage .*
((MineCost+PlantCost)*ones(size(nLevel,1),1));
ProfitAll = Revenue_All - Cost_All;
%------------------------------------------
X = cell(nLevel,1);
nRows = max(InputData.Rows);
nColumns = max(InputData.Columns);
Tonnage = cell(nLevel,1);
Profit = cell(nLevel,1);
for iiLoop = 1:nLevel
    X{iiLoop} = zeros(nRows,nColumns);
    LevelInfo = [InputData.Rows(InputData.Levels ==
iiLoop),InputData.Columns(InputData.Levels ==
iiLoop),InputData.Rock(InputData.Levels ==
iiLoop),InputData.Tonnage(InputData.Levels ==
iiLoop),ProfitAll(InputData.Levels == iiLoop)];
    for iLoop = 1:length(LevelInfo)
        X{iiLoop}(LevelInfo(iLoop,1),LevelInfo(iLoop,2)) =
double(LevelInfo(iLoop,3)==12);  %Specifying the ore blocks in
each level
        Tonnage{iiLoop}(LevelInfo(iLoop,1),LevelInfo(iLoop,2)) =
X{iiLoop}(LevelInfo(iLoop,1),LevelInfo(iLoop,2)).*(LevelInfo(iLoop
,4));  %Put the relative tonnage of each ore block in each level
        Profit{iiLoop}(LevelInfo(iLoop,1),LevelInfo(iLoop,2)) =
X{iiLoop}(LevelInfo(iLoop,1),LevelInfo(iLoop,2)).*(LevelInfo(iLoop
,5));   %Put the relative profit of each ore block in each level
    end
end
MHOD.TopLevel = cell(nLevel,1);
MHOD.TopTonnage = cell(nLevel,1);
MHOD.TopProfit = cell(nLevel,1);
for j = 1:nLevel
    MHOD.TopLevel{j} = zeros(nRows,nColumns);
    MHOD.TopTonnage{j} = zeros(nRows,nColumns);
    MHOD.TopProfit{j} = zeros(nRows,nColumns);
    for jLoop = j:-1:1
         MHOD.TopLevel{j} = MHOD.TopLevel{j} + (X{j}.*X{jLoop});
% Calculating the number of ore blocks above each ore block in
every level including the ore blocks in that level itself.
         MHOD.TopTonnage{j} = MHOD.TopTonnage{j} +
(X{j}.*Tonnage{jLoop});   %Calculating the total tonnage of ore
blocks above each block in every level.
         MHOD.TopProfit{j} = MHOD.TopProfit{j} +
(X{j}.*Profit{jLoop});      %Calculating the total Proft of ore
blocks above each block in every level.
    end
end
```

```
%Calculating the Total Tonnage and Profit of ore above each level
MHOD.Total_Ore_Profit_each_Level = zeros(nLevel,1);
MHOD.Total_Ore_Tonnage_each_Level = zeros(nLevel,1);
for t = 1:nLevel
    MHOD.Total_Ore_Profit_each_Level(t) =
sum(sum(MHOD.TopProfit{t}));
    MHOD.Total_Ore_Tonnage_each_Level(t) =
sum(sum(MHOD.TopTonnage{t}));
    MHOD.Test = sum(MHOD.TopTonnage{t});
end
save('Results/MHOD','MHOD')
%-------------------------Plot the tonnage & profit of each level
figure
X_Axis = 1:nLevel;
[ax,p3,p4] =
plotyy(X_Axis,MHOD.Total_Ore_Tonnage_each_Level(:,1),X_Axis,MHOD.T
otal_Ore_Profit_each_Level(:,1),'semilogy','plot');
ylabel(ax(1),'Ore Tonnage (kt)') % label left y-axis
ylabel(ax(2),'Ore Profit (b$)') % label right y-axis
xlabel(ax(2),'Levels') % label x-axis
p3.LineStyle = '--';
p3.LineWidth = 2;
p4.LineWidth = 2;

%-----------------------------MHOD (Second Method)--------------
%Considering the extraction rate & discounted rate in the ore
profit calculation
prompt1 = 'What is the blocks height? ';
BLsHeight = input(prompt1);
prompt2 = 'What is Extraction Rate? ';
Ex = input(prompt2);
%------------%Calculating the NPV for each ore Block of each level
iRate = 0.1;
B1.NPV = cell(nLevel,1);
for lLoop = nLevel:-1:1
    B1.NPV{lLoop} = zeros(nRows,nColumns);
    for rLoop = 1:nRows
        for cLoop = 1:nColumns
            if X{lLoop}(rLoop,cLoop) == 1
                for Level = lLoop:-1:1
                    B1.NPV{lLoop}(rLoop,cLoop) =
B1.NPV{lLoop}(rLoop,cLoop)+(Profit{Level}(rLoop,cLoop)*X{Level}(rL
oop,cLoop)/(1+iRate)^((lLoop-Level)*BLsHeight/Ex));
                end
            end
        end
    end
end
%-----------------------------------------------------------------
%Calculating the total NPV for the ore blocks of each level
considering
%just the ore blocks above
%-----------------------------------------------------------------
B1.NPV_Total = zeros(nLevel,1);
for Level = 1:nLevel
    B1.NPV_Total(Level) = sum(sum(B1.NPV{Level}));
end
```

```
%--------------------------------------------------
%Plotting discounted profit & tonnage in one figure
%--------------------------------------------------
figure
X_Axis = 1:nLevel;
[ax,p3,p4] =
plotyy(X_Axis,MHOD.Total_Ore_Tonnage_each_Level(:,1),X_Axis,
B1.NPV_Total(:,1),'semilogy','plot');
ylabel(ax(1),'Ore Tonnage (kt)') % label left y-axis
ylabel(ax(2),'Ore Discounted Profit (b$)') % label right y-axis
xlabel(ax(2),'Levels') % label x-axis
p3.LineStyle = '--';
p3.LineWidth = 2;
p4.LineWidth = 2;
save('Results/B1','B1')
```

APPENDIX B

PRODUCTION SCHEDULING

After finding the best level of extraction the following steps should be done to run the MILP model:

1. Import block model.
2. Find the maximum and minimum of the coordinates.
3. Determine the coordinates of the ore blocks in a given level and plot the ore boundary for that specific level.
4. Create big-blocks and calculate the profit, tonnage and average grade of each big-block.
5. Plot the boundary of created big-blocks.
6. Determine the coordinates, indices, grade, profit, and tonnage of big-blocks with more than seven small blocks in them.
7. Determine the scheduling parameters.
8. Create the objective function.
9. Create the binary constraints.
10. Create the mining capacity constraint.
11. Create grade blending constraint.
12. Create drawrate constraint.
13. Create reserve constraint.
14. For precedence constraint the following steps should be applied:
    - 14.1. Find the adjacent big-blocks for each big-block.
    - 14.2. Determine the mining direction and find the predecessor big- blocks for each big-block.
    - 14.3. Create the precedence constraint.

15. Create the number of new big-blocks constraint.

16. Run the model.

```
function Import()
[filename] = uigetfile('.txt');
fileID = fopen(filename);
Data = fscanf(fileID,'%f %f %f %f %f %f %f %f %f %f',[10 Inf])';
InputData.Levels = Data(:,1);
InputData.Rows = Data(:,2);
InputData.Columns = Data(:,3);
InputData.X = Data(:,4);
InputData.Y = Data(:,5);
InputData.Z = Data(:,6);
InputData.Rock = Data(:,7);
InputData.CU = Data(:,8);
InputData.Tonnage = Data(:,9);
InputData.Density = Data(:,10);
save('Results/InputData','InputData');
h = msgbox('All required data was imported.');
end
```

```
function  MaxMin()
load('Results\InputData.mat');
InputData.ux= max(InputData.X);
InputData.lx= min(InputData.X);
InputData.uy= max(InputData.Y);
InputData.ly= min(InputData.Y);
InputData.uz= max(InputData.Z);
InputData.lz= min(InputData.Z);
save('Results\InputData','InputData');
h = msgbox('Operation Completed');
end
```

```
function Plot_PlanView
load('Results/InputData.mat')
prompt = 'What is the production level? ';
noLevel = input(prompt);
% find the size of the matrix
nRow = max(InputData.Rows);
nCol = max(InputData.Columns);
nLevel = max(InputData.Levels);
%join row, col, rocktype
LRCRT=[InputData.Levels,InputData.Rows,InputData.Columns,InputData
.Rock,InputData.X,InputData.Y];
% create the main matrix
MainMatrix = zeros(nRow,nCol);
X = zeros(nRow,nCol);
Y = zeros(nRow,nCol);
i_index = zeros(nRow,nCol);
j_index = zeros(nRow,nCol);
for level = 1:nLevel
    Data = LRCRT(((nRow*nCol*(level-1)+1)):(level*nRow*nCol),:);
    for iLoop = 1:(nRow*nCol)
            if Data(iLoop,4) == 12 && Data(iLoop,1) <= noLevel
                MainMatrix(Data(iLoop,2),Data(iLoop,3)) = 1;
                i_index(Data(iLoop,2),Data(iLoop,3)) =
Data(iLoop,2);
                j_index(Data(iLoop,2),Data(iLoop,3)) =
Data(iLoop,3);
                X(Data(iLoop,2),Data(iLoop,3)) = Data(iLoop,5);
                Y(Data(iLoop,2),Data(iLoop,3)) = Data(iLoop,6);
            end
        end
end
InputPlot.Level = noLevel;
InputPlot.i_plot = i_index(find(i_index));
InputPlot.j_plot = j_index(find(i_index));
InputPlot.X_plot = X(find(X));
InputPlot.Y_plot = Y(find(Y));
%----------------------------- >> Plots darwpoints
 figure
 hold on
  k = boundary(InputPlot.X_plot,InputPlot.Y_plot);
  % Find Boundary Data (I, J, K, X, Y)
  BoundaryData = zeros (numel(k),4);
  BoundaryData(:,1) = InputPlot.i_plot(k);
  BoundaryData(:,2) = InputPlot.j_plot(k);
  BoundaryData(:,3) = noLevel*ones(numel(k),1);
  BoundaryData(:,4) = InputPlot.X_plot(k);
  BoundaryData(:,5) = InputPlot.Y_plot(k);
  InputPlot.BoundaryData = BoundaryData;
  plot(InputPlot.X_plot(k),...
       InputPlot.Y_plot(k),...
                  'Linewidth',2,...
                  'linestyle', '-',...
                  'color','g');
xlabel('X (m)','fontsize',11,'fontweight','bold');
ylabel('Y (m)','fontsize',11,'fontweight','bold');
axis equal %square
% find Min X, Y for this level
MinX_L = min(BoundaryData(:,4));
MaxX_L = max(BoundaryData(:,4));
MinY_L = min(BoundaryData(:,5));
MaxY_L = max(BoundaryData(:,5));
ax = gca;
ax.XGrid = 'on';
ax.YGrid = 'on';
ax.XTick = [MinX_L-5:10:MaxX_L+5];
ax.YTick = [MinY_L-5:10:MaxY_L+5];
save('Results\InputPlot','InputPlot');
```

```
function OutputPlot = ClusteringBlocksAll(InputPlot)
load('Results/InputPlot.mat')
prompt = 'What is the production level? ';
noLevel = input(prompt);
OreBlocks = [InputPlot.X_plot,InputPlot.Y_plot];
MinX = min(InputPlot.X_plot);
MinY = min(InputPlot.Y_plot);
MaxX = max(InputPlot.X_plot);
MaxY = max(InputPlot.Y_plot);
OutputPlot.id9 = [];
counter9 = 1;
```

```
for yloop = (MinY):30:(MaxY)
    for xloop = (MinX):30:(MaxX)
        x1 = xloop;
        x2 = x1+30;
        y1 = yloop;
        y2 = y1+30;
        OutputPlot.id9(OreBlocks(:,1) >= x1 & OreBlocks(:,1) <= x2
& OreBlocks(:,2) >= y1 & OreBlocks(:,2) <= y2) = counter9;
            counter9 = counter9 + 1;
    end
end
OutputPlot.squares=[InputPlot.X_plot,InputPlot.Y_plot,OutputPlot.i
d9'];
%*************************************************
Data_id = OutputPlot.squares(:,3);
Min_id = min(Data_id);
Max_id = max(Data_id);
Data_X = OutputPlot.squares(:,1);
Data_Y = OutputPlot.squares(:,2);
Unique_Ids = unique(OutputPlot.squares(:,3));
noBigBls = numel(Unique_Ids);
XY_MinMax = zeros(noBigBls,4);
noXs_Bl = zeros(noBigBls,1);
noYs_Bl = zeros(noBigBls,1);
CenterData = zeros(noBigBls,3);
for BlLoop = 1: noBigBls
    Xs_Bl = Data_X(Data_id==Unique_Ids(BlLoop,1));
    Ys_Bl = Data_Y(Data_id==Unique_Ids(BlLoop,1));

    noXs_Bl(BlLoop,1) = numel(Xs_Bl);
    noYs_Bl(BlLoop,1) = numel(Ys_Bl);
    CenterData(BlLoop,1) = min(Xs_Bl)+10;
    CenterData(BlLoop,2) = min(Ys_Bl)+10;
    CenterData(BlLoop,3) = BlLoop;

    XY_MinMax(BlLoop,1) = min(Xs_Bl);
    XY_MinMax(BlLoop,2) = max(Xs_Bl);

    XY_MinMax(BlLoop,3) = min(Ys_Bl);
    XY_MinMax(BlLoop,4) = max(Ys_Bl);
end
OutputPlot.XY_MinMax = XY_MinMax;
OutputPlot.CenterData = CenterData;

load('Results\InputPlot.mat');
load('InputData.mat');
Total_X = InputData.X;
Total_Y = InputData.Y;
Total_Levels = InputData.Levels;
Total_Rock = InputData.Rock;
Total_CU = InputData.CU;
Total_Tonnage = InputData.Tonnage;
Total_Density = InputData.Density;
nLevel = max(InputData.Levels);
clear InputData
Price = 6000         ;% (US$/t)
```

```
SellingCost = 0.50   ;% (US$/t)
MineCost = 10        ;% (US$/t)
PlantCost = 16.1     ;% (US$/t)
Rec = 0.85           ;% (%)
Revenue_All = Total_Tonnage .* Rec .* ((Total_CU)./100) .*((Price-
SellingCost)*ones(size(nLevel,1),1));
Cost_All = Total_Tonnage .*
((MineCost+PlantCost)*ones(size(nLevel,1),1));
ProfitAll = Revenue_All - Cost_All;
for BigBL =1:noBigBls
     TempData_ind = find(Total_X > XY_MinMax(BigBL,1)-0.5 &
Total_X <XY_MinMax(BigBL,2)+0.5 & Total_Y > XY_MinMax(BigBL,3)-0.5
& Total_Y <XY_MinMax(BigBL,4)+0.5  );
     Temp_Data
=[Total_Levels(TempData_ind),Total_Rock(TempData_ind),Total_CU(Tem
pData_ind),ProfitAll(TempData_ind),Total_Tonnage(TempData_ind),Tot
al_Density(TempData_ind)];
     Ton = 0;
     bl = 0;
     P = 0;
     SigmaGrade = 0;
     SigmaDensity = 0;
     for kLoop = 1:numel(TempData_ind)
         if Temp_Data(kLoop,1) <= noLevel && Temp_Data(kLoop,2)~=
11
         bl = bl + 1;
         Ton = Ton + Temp_Data(kLoop,5);
         P = P + Temp_Data(kLoop,4);
         SigmaGrade = SigmaGrade + Temp_Data(kLoop,3);
         SigmaDensity = SigmaDensity + Temp_Data(kLoop,6);
         end
     end
     BigBLsData.allTon (BigBL,1) = Ton;
     %-----------------------------------------------------
     BigBLsData.Profit(BigBL,1) = P;
     BigBLsData.CU (BigBL,1) = SigmaGrade / (bl);
     BigBLsData.Density (BigBL,1) = SigmaDensity / (bl);
     %------------------------------------------------------
end
     OutputPlot.BigBLsData =
[BigBLsData.allTon,BigBLsData.Profit,BigBLsData.CU,BigBLsData.Dens
ity];
%---------------------------- >> Plot
blk_i = InputPlot.X_plot;
blk_j = InputPlot.Y_plot;
g = OutputPlot.id9';
mark0 = num2str(g);
mark = cellstr(mark0);
figure('units','normalized','outerposition',[0 0 1 1]);
F = gscatter(blk_i,blk_j,g,'','s',15);
title('Block Model');
xlabel('X coordinate');
ylabel('Y coordinate');
text(blk_i,blk_j,mark,'FontSiz',7);
save('Results\OutputPlot','OutputPlot');
end
```

```
function Plot_PlanView_BigBL()
load('Results/InputData.mat')
prompt = 'What is the production level? ';
noLevel = input(prompt);
% find the size of the matrix
nRow = max(InputData.Rows);
nCol = max(InputData.Columns);
nLevel = max(InputData.Levels);
%join row, col, rocktype
LRCRT =
[InputData.Levels,InputData.Rows,InputData.Columns,InputData.Rock,
InputData.X,InputData.Y];
% create the main matrix
MainMatrix = zeros(nRow,nCol);
X = zeros(nRow,nCol);
Y = zeros(nRow,nCol);
i_index = zeros(nRow,nCol);
j_index = zeros(nRow,nCol);
for level = 1:nLevel
    Data = LRCRT(((nRow*nCol*(level-1)+1)):(level*nRow*nCol),:);
    for iLoop = 1:(nRow*nCol)
            if Data(iLoop,4) == 12 && Data(iLoop,1) <= noLevel
                MainMatrix(Data(iLoop,2),Data(iLoop,3)) = 1;
                i_index(Data(iLoop,2),Data(iLoop,3)) =
Data(iLoop,2);
                j_index(Data(iLoop,2),Data(iLoop,3)) =
Data(iLoop,3);
                X(Data(iLoop,2),Data(iLoop,3)) = Data(iLoop,5);
                Y(Data(iLoop,2),Data(iLoop,3)) = Data(iLoop,6);
            end
    end
end
PLPV.i_plot = i_index(find(i_index));
PLPV.j_plot = j_index(find(j_index));
PLPV.X_plot = X(find(X));
PLPV.Y_plot = Y(find(Y));
save('Results/PLPV','PLPV')
%*****************************************************************
MinX = min(PLPV.X_plot);
MinY = min(PLPV.Y_plot);
MaxX = max(PLPV.X_plot);
MaxY = max(PLPV.Y_plot);
OutputPlot.id9 = [];
counter9 = 1;
for yloop = (MinY):30:(MaxY)
    for xloop = (MinX):30:(MaxX)
        x1 = xloop;
        x2 = x1+30;
        y1 = yloop;
        y2 = y1+30;
        OutputPlot.id9(PLPV.X_plot >= x1 & PLPV.X_plot <= x2 &
PLPV.Y_plot >= y1 & PLPV.Y_plot <= y2) = counter9;
            counter9 = counter9 + 1;
    end
end
OutputPlot.squares = [PLPV.X_plot,PLPV.Y_plot,OutputPlot.id9'];
% How many Big Block
```

```
Data_id = OutputPlot.squares(:,3);
Min_id = min(Data_id);
Max_id = max(Data_id);
Data_X = OutputPlot.squares(:,1);
Data_Y = OutputPlot.squares(:,2);
Unique_Ids = unique(OutputPlot.squares(:,3));
noBigBls = numel(Unique_Ids);
CornerData = zeros(noBigBls,4);
CenterData = zeros(noBigBls,3);
% find the left bottom corner of each Big Bl
for BlLoop = 1: noBigBls
    % which blocks equal to id
    %BL_id9 = find(Data_id==BlLoop);
    Xs_Bl = Data_X(find(Data_id==Unique_Ids(BlLoop,1)));
    Ys_Bl = Data_Y(find(Data_id==Unique_Ids(BlLoop,1)));
    if numel (Xs_Bl) >= 7
    CornerData(BlLoop,1) = min(Xs_Bl)-5;
    CornerData(BlLoop,2) = min(Ys_Bl)-5;
    CornerData(BlLoop,3) = 1000;
    CornerData(BlLoop,4) = BlLoop;
    %***********************************
    Xs_Bl = Data_X(Data_id==Unique_Ids(BlLoop,1));
    Ys_Bl = Data_Y(Data_id==Unique_Ids(BlLoop,1));
    noXs_Bl(BlLoop,1) = numel(Xs_Bl);
    noYs_Bl(BlLoop,1) = numel(Ys_Bl);
    CenterData(BlLoop,1) = min(Xs_Bl)+10;
    CenterData(BlLoop,2) = min(Ys_Bl)+10;
    CenterData(BlLoop,3) = BlLoop;
    %***********************************
    end
end
PLPV.CornerData = CornerData;
PLPV.CenterData = CenterData;
 figure
 for Rect = 1: noBigBls
     if CornerData(Rect,3) == 1000
     Xrec = CornerData(Rect,1);
     Yrec = CornerData(Rect,2);
     rectangle('Position',[Xrec,Yrec,30,30]);
     end
 end
%*****************************************************************
  k = boundary(PLPV.X_plot,PLPV.Y_plot);
  % Find Boundary Data (I, J, K, X, Y)
  BoundaryData = zeros (numel(k),4);
  BoundaryData(:,1) = PLPV.i_plot(k);
  BoundaryData(:,2) = PLPV.j_plot(k);
  BoundaryData(:,3) = noLevel*ones(numel(k),1);
  BoundaryData(:,4) = PLPV.X_plot(k);
  BoundaryData(:,5) = PLPV.Y_plot(k);

  PLPV.BoundaryData = BoundaryData;
  plot(PLPV.X_plot(k)-5,...
       PLPV.Y_plot(k)-5,...
                  'Linewidth',2,...
                  'linestyle', '-',...
                  'color','g');
```

```
xlabel('X (m)','fontsize',11,'fontweight','bold');
ylabel('Y (m)','fontsize',11,'fontweight','bold');
axis equal %square
% find Min X, Y for this level
MinX_L = min(BoundaryData(:,4));
MaxX_L = max(BoundaryData(:,4));
MinY_L = min(BoundaryData(:,5));
MaxY_L = max(BoundaryData(:,5));
ax = gca;
ax.XGrid = 'on';
ax.YGrid = 'on';
ax.XTick = (MinX_L-5:100:MaxX_L+5);
ax.YTick = (MinY_L-5:50:MaxY_L+5);
save('Results\PLPV','PLPV');
x = CenterData (:,1);
y = CenterData (:,2);
ID = CenterData (:,3);
for jLoop=1:numel(x)
                 line(x(jLoop),...
                 y(jLoop),...
                 'linestyle', 'none',...
                 'marker', 'o',...
                 'MarkerSize',2,...
                 'MarkerEdgeColor',[0 0 0],...
                 'MarkerFaceColor','w');
             text(x(jLoop),y(jLoop),num2str(ID(jLoop)),...
                  'FontName','Times New Roman',...
                  'Color','k','FontSize',12,'fontweight','bold');
 end
load OutputPlot.mat
X_CenterBL = OutputPlot.CenterData(:,1);
Y_CenterBL = OutputPlot.CenterData(:,2);
BBLProfit = OutputPlot.BigBLsData(:,2);

MinX = min(X_CenterBL);
MaxX = max(X_CenterBL);

MinY = min(Y_CenterBL);
MaxY = max(Y_CenterBL);

u = linspace(MaxX,MinX,100);
v = linspace(MaxY,MinY,100);
[X,Y] = meshgrid(u,v);
Z = griddata(X_CenterBL,Y_CenterBL,BBLProfit, X, Y);
%plotting based on the Big Blocks Profit
mesh(X,Y,Z);
xlabel('X(m)');
ylabel('Y(m)');
zlabel('BBLProfit (M$)');
%axis tight
axis equal %square
% find Min X, Y for this level
MinX_L = min(BoundaryData(:,4));
MaxX_L = max(BoundaryData(:,4));
MinY_L = min(BoundaryData(:,5));
MaxY_L = max(BoundaryData(:,5));
```

```
ax = gca;
ax.XGrid = 'on';
ax.YGrid = 'on';
ax.XTick = (MinX_L-5:10:MaxX_L+5);
ax.YTick = (MinY_L-5:10:MaxY_L+5);
%shading interp
colorbar
view(0,90)
```

```
function ModelInputAll()
prompt = 'What is the production level? ';
noLevel = input(prompt);
load('Results\PLPV.mat')
% get Id of candidate blocks
ID_selected_BLs = find (PLPV.CornerData(:,4));
P = PLPV.CenterData(:,1);
M = PLPV.CenterData(:,2);
X_selected_BLs = P(find (PLPV.CenterData(:,1)));
Y_selected_BLs = M(find (PLPV.CenterData(:,2)));
load('Results\OutputPlot.mat')
TonTemp = OutputPlot.BigBLsData(:,1);
ModelTon = TonTemp(ID_selected_BLs);
DensityTemp = OutputPlot.BigBLsData(:,4);
ModelDensity = DensityTemp(ID_selected_BLs);
GradeTemp = OutputPlot.BigBLsData(:,3);
ModelGrade = GradeTemp(ID_selected_BLs);
ProfitTemp = OutputPlot.BigBLsData(:,2);
ModelProfit = ProfitTemp(ID_selected_BLs);
Index = (1:numel(ID_selected_BLs))';
ModelInputNotInclude =
[ID_selected_BLs,X_selected_BLs,Y_selected_BLs , ModelTon,
ModelGrade, ModelProfit,Index, ModelDensity];
ModelInputInclude =
[OutputPlot.CenterData(:,3),OutputPlot.CenterData(:,1),OutputPlot.
CenterData(:,2),TonTemp,GradeTemp,ProfitTemp];
name = strcat('Level_',num2str(noLevel));
if exist('Model\MainModelInput.mat') == 2
    load('Model\MainModelInput.mat')
    MainModelInput.NotInclude.(name) = ModelInputNotInclude;
    MainModelInput.Include.(name) = ModelInputInclude;
    save('Model\MainModelInput', 'MainModelInput');
else
    MainModelInput.NotInclude.(name) = ModelInputNotInclude;
    MainModelInput.Include.(name) = ModelInputInclude;
    save('Model\MainModelInput', 'MainModelInput');
end
NoBigBlInclude.(name) = numel(MainModelInput.Include.(name)(:,1));
NoBigBlNotInclude.(name) =
numel(MainModelInput.NotInclude.(name)(:,1));

NoBigBlDif.(name) = NoBigBlInclude.(name) -
NoBigBlNotInclude.(name);

SumTonnageInclude.(name) =
sum(MainModelInput.Include.(name)(:,4));
```

```
SumTonnageNotInclude.(name) =
sum(MainModelInput.NotInclude.(name)(:,4));

SumTonnageDif.(name) = SumTonnageInclude.(name) -
SumTonnageNotInclude.(name);
MainModelInput.Total.(name) =
[NoBigBlInclude.(name),SumTonnageInclude.(name),NoBigBlNotInclude.
(name),SumTonnageNotInclude.(name),NoBigBlDif.(name),SumTonnageDif
.(name)];
save('Model\MainModelInput', 'MainModelInput');
end
```

```
function varargout = SchedulingParameters(varargin)
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn',
@SchedulingParameters_OpeningFcn, ...
                   'gui_OutputFcn',
@SchedulingParameters_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
function SchedulingParameters_OpeningFcn(hObject, eventdata,
handles, varargin)
% Choose default command line output for Scheduling Parameters.
% If the user does not enter the value these default will be used.
handles.output = hObject;
num=0;                                 % default value=0
handles.edit_ND = num;                 % Number of Big Blocks
handles.edit_Periods = num;            % Number of Periods
handles.edit_i = num;                  % Discount rate
handles.edit_Mue = num;                % number related to
uncertainty
handles.edit_M = num;                  % big enough number for
Binary cons.
handles.edit_Cost = num;               % Mining cost
handles.edit_LowerCapacity = num;      % Lower Mining Capacity
handles.edit_UpperCapacity = num;      % Upper Mining Capacity
handles.edit_LowerGrade = num;         % Lower Grade
handles.edit_UpperGrade = num;         % Upper Grade
handles.edit_LowerNnew = num;          % Lower Nnew
handles.edit_UpperNnew = num;          % Upper Nnew
handles.edit_NactL = num;              % Minimum Number of Active
Drawpoints
handles.edit_NactU = num;              % Maximum Number of Active
Drawpoints
```

```
handles.edit_Workingdays = 350;       % # of working days during a
year
handles.edit_LowerPRC = num;          % Lower Draw rate
handles.edit_UpperPRC = num;          % Upper Draw rate
handles.edit_P1_dep = num;            % PRC curve 1st point
depletion(P1)
handles.edit_P2_dep = num;            % PRC curve 2nd point
depletion(P2)
handles.edit_P3_dep = num;            % PRC curve 3rd point
depletion(P3)
handles.edit_P1_DR = num;             % PRC curve 1st point draw
rate(P1)
handles.edit_P2_DR = num;             % PRC curve 2nd point draw
rate(P2)
handles.edit_P3_DR = num;             % PRC curve 3rd point draw
rate(P3)
%=========================================PLOT HISTOGRAMS
load('MODEL/MainModelInput.mat')
DataTemp = MainModelInput.NotInclude.Level_39;
colormap cool
%======= Grade Histogram
axes(handles.axes_GradeHist)
hist(handles.axes_GradeHist,DataTemp(:,5));
xlabel('Grade(%)','fontsize',9,'fontweight','bold');
grid on
datacursormode on
%======= Tonnage Histogram
axes(handles.axes_TonnageHist)
hist(handles.axes_TonnageHist,DataTemp(:,4));
xlabel('Tonnes','fontsize',9,'fontweight','bold');
grid on
datacursormode on
%======= May we need in future
 axes(handles.axes_TonnageBarChart)
 %hist(handles.axes_TonnageBarChart,DPtonnage);
X_DR = DataTemp(:,2);
Y_DR = DataTemp(:,3);
ND =size(X_DR);
hold on
     %------------------------------ >> Plots darwpoints
 for jLoop=1:ND
                line(X_DR(jLoop),...
                 Y_DR(jLoop),...
                 'linestyle', 'none',...
                 'marker', 'o',...
                 'MarkerSize',5,...
                 'MarkerEdgeColor',[0 0 0],...
                 'MarkerFaceColor','g');
text(X_DR(jLoop)+2.5,Y_DR(jLoop),num2str(jLoop),...
                  'FontName','Times New Roman',...
                  'Color','b','FontSize',4,'fontweight','bold');
end
datacursormode on
% Tonnage & Grade ============================
%== Total tonnage
str1=sprintf('%g',sum(DataTemp(:,4))/1000000);
set(handles.text_TG1,'string',str1);
```

```
%== Min grade
str2=sprintf('%g',min(DataTemp(:,5)));
set(handles.text_TG2,'string',str2);
%== Max grade
str3=sprintf('%g',max(DataTemp(:,5)));
set(handles.text_TG3,'string',str3);

%== Weighted grade
str4=sprintf('%g',sum(DataTemp(:,4) .*
DataTemp(:,5))/sum(DataTemp(:,4)));
set(handles.text_TG4,'string',str4);

%== Min tonnage
str5=sprintf('%g',min(DataTemp(:,4)));
set(handles.text_TG5,'string',str5);

%== Max tonnage
str6=sprintf('%g',max(DataTemp(:,4)));
set(handles.text_TG6,'string',str6);

%== Number of DPs
str7=sprintf('%g',numel(DataTemp(:,4)));
set(handles.DP,'string',str7);

% %== Mining Capacity upper bound
 str8=sprintf('%g',sum(DataTemp(:,4))*1.2);
 set(handles.Mcap_Up,'string',str8);

% Update handles structure
guidata(hObject, handles);

% --- Outputs from this function are returned to the command line.
function varargout = SchedulingParameters_OutputFcn(hObject,
eventdata, handles)
% Get default command line output from handles structure
varargout{1} = handles.output;
**************************************>> Number of Drawpoints
function edit_ND_Callback(hObject, eventdata, handles)

num = str2double(get(hObject,'String'));
if isnan(num)
    num = 0;
    set(hObject,'String',num);
    beep
    errordlg('Input must be a number', 'Error')
end
handles.edit_ND = num;
guidata(hObject,handles)

function edit_ND_CreateFcn(hObject, eventdata, handles)

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
```

```
    set(hObject,'BackgroundColor','white');
end
%*******************************************>> Number of Periods
function edit_Periods_Callback(hObject, eventdata, handles)
num = str2double(get(hObject,'String'));
if isnan(num)
    num = 0;
    set(hObject,'String',num);
    beep
    errordlg('Input must be a number', 'Error')
end
handles.edit_Periods = num;
guidata(hObject,handles)
function edit_Periods_CreateFcn(hObject, eventdata, handles)

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
%*****************************************>> Discount rate
function edit_i_Callback(hObject, eventdata, handles)

num = str2double(get(hObject,'String'));
if isnan(num)
    num = 0;
    set(hObject,'String',num);
    beep
    errordlg('Input must be a number', 'Error')
end
handles.edit_i = num;
guidata(hObject,handles)

function edit_i_CreateFcn(hObject, eventdata, handles)

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
%**********************************************>> Mue
function edit_Mue_Callback(hObject, eventdata, handles)

num = str2double(get(hObject,'String'));
if isnan(num)
    num = 0;
    set(hObject,'String',num);
    beep
    errordlg('Input must be a number', 'Error')
end
handles.edit_Mue = num;
guidata(hObject,handles)

function edit_Mue_CreateFcn(hObject, eventdata, handles)
```

```
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
%****************************************************>> M
function edit_M_Callback(hObject, eventdata, handles)

num = str2double(get(hObject,'String'));
if isnan(num)
    num = 0;
    set(hObject,'String',num);
    beep
    errordlg('Input must be a number', 'Error')
end
handles.edit_M = num;
guidata(hObject,handles)

% --- Executes during object creation, after setting all
properties.
function edit_M_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
%*************************************************>> Mining Cost
function edit_Cost_Callback(hObject, eventdata, handles)

num = str2double(get(hObject,'String'));
if isnan(num)
    num = 0;
    set(hObject,'String',num);
    beep
    errordlg('Input must be a number', 'Error')
end
handles.edit_Cost = num;
guidata(hObject,handles)

function edit_Cost_CreateFcn(hObject, eventdata, handles)

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
%***************************>> Number of Active Drawpoints
% Lower bound
function edit_NactL_Callback(hObject, eventdata, handles)
num = str2double(get(hObject,'String'));
if isnan(num)
    num = 0;
    set(hObject,'String',num);
    beep
    errordlg('Input must be a number', 'Error')
end
handles.edit_NactL = num;
guidata(hObject,handles)

function edit_NactL_CreateFcn(hObject, eventdata, handles)

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% Upper bound
function edit_NactU_Callback(hObject, eventdata, handles)

num = str2double(get(hObject,'String'));
if isnan(num)
    num = 0;
    set(hObject,'String',num);
    beep
    errordlg('Input must be a number', 'Error')
end
handles.edit_NactU = num;
guidata(hObject,handles)

function edit_NactU_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
%******************************>> Lower Mining Capacity

function edit_LowerCapacity_Callback(hObject, eventdata, handles)

num = str2double(get(hObject,'String'));
if isnan(num)
    num = 0;
    set(hObject,'String',num);
    beep
    errordlg('Input must be a number', 'Error')
end
handles.edit_LowerCapacity = num;
guidata(hObject,handles)

function edit_LowerCapacity_CreateFcn(hObject, eventdata, handles)

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
%**************************************>> Upper Mining Capacity

function edit_UpperCapacity_Callback(hObject, eventdata, handles)

num = str2double(get(hObject,'String'));
if isnan(num)
    num = 0;
    set(hObject,'String',num);
    beep
    errordlg('Input must be a number', 'Error')
end
handles.edit_UpperCapacity = num;
guidata(hObject,handles)

function edit_UpperCapacity_CreateFcn(hObject, eventdata, handles)

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
%***********************************>> Lower grade

function edit_LowerGrade_Callback(hObject, eventdata, handles)

num = str2double(get(hObject,'String'));
if isnan(num)
    num = 0;
    set(hObject,'String',num);
    beep
    errordlg('Input must be a number', 'Error')
end
handles.edit_LowerGrade = num;
guidata(hObject,handles)

function edit_LowerGrade_CreateFcn(hObject, eventdata, handles)

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
%*************************************************>> Upper grade

function edit_UpperGrade_Callback(hObject, eventdata, handles)
num = str2double(get(hObject,'String'));
if isnan(num)
    num = 0;
    set(hObject,'String',num);
    beep
    errordlg('Input must be a number', 'Error')
end
handles.edit_UpperGrade = num;
```

```
guidata(hObject,handles)

function edit_UpperGrade_CreateFcn(hObject, eventdata, handles)

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
%***************************************************>> Lower Nnew

function edit_LowerNnew_Callback(hObject, eventdata, handles)

num = str2double(get(hObject,'String'));
if isnan(num)
    num = 0;
    set(hObject,'String',num);
    beep
    errordlg('Input must be a number', 'Error')
end
handles.edit_LowerNnew = num;
guidata(hObject,handles)

function edit_LowerNnew_CreateFcn(hObject, eventdata, handles)

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
%***************************************************>> Upper Nnew

function edit_UpperNnew_Callback(hObject, eventdata, handles)

num = str2double(get(hObject,'String'));
if isnan(num)
    num = 0;
    set(hObject,'String',num);
    beep
    errordlg('Input must be a number', 'Error')
end
handles.edit_UpperNnew = num;
guidata(hObject,handles)
function edit_UpperNnew_CreateFcn(hObject, eventdata, handles)

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
%*************************************>> Working days and radius
```

```
function edit_Workingdays_Callback(hObject, eventdata, handles)
num = str2double(get(hObject,'String'));
if isnan(num)
    num = 0;
    set(hObject,'String',num);
    beep
    errordlg('Input must be a number', 'Error')
end
handles.edit_Workingdays = num;

guidata(hObject,handles)

function edit_Workingdays_CreateFcn(hObject, eventdata, handles)

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
%**************************************************>> Lower PRC

function edit_LowerPRC_Callback(hObject, eventdata, handles)

num = str2double(get(hObject,'String'));
if isnan(num)
    num = 0;
    set(hObject,'String',num);
%     beep
%     errordlg('Input must be a number', 'Error')
end
handles.edit_LowerPRC = num;
guidata(hObject,handles)

function edit_LowerPRC_CreateFcn(hObject, eventdata, handles)

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
%****************************************************>> Upper PRC

function edit_UpperPRC_Callback(hObject, eventdata, handles)

num = str2double(get(hObject,'String'));
if isnan(num)
    num = 0;
    set(hObject,'String',num);
%     beep
%     errordlg('Input must be a number', 'Error')
end
handles.edit_UpperPRC = num;
guidata(hObject,handles)
```

```
function edit_UpperPRC_CreateFcn(hObject, eventdata, handles)

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
%*******************************************>> PRC points

function edit_P1_dep_Callback(hObject, eventdata, handles)
num = str2double(get(hObject,'String'));
if isnan(num)
    num = 0;
    set(hObject,'String',num);
%     beep
%     errordlg('Input must be a number', 'Error')
end
handles.edit_P1_dep = num;
guidata(hObject,handles)

function edit_P1_dep_CreateFcn(hObject, eventdata, handles)

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit_P2_dep_Callback(hObject, eventdata, handles)
num = str2double(get(hObject,'String'));
if isnan(num)
    num = 0;
    set(hObject,'String',num);
%     beep
%     errordlg('Input must be a number', 'Error')
end
handles.edit_P2_dep = num;
guidata(hObject,handles)

function edit_P2_dep_CreateFcn(hObject, eventdata, handles)

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit_P3_dep_Callback(hObject, eventdata, handles)
num = str2double(get(hObject,'String'));
if isnan(num)
    num = 0;
    set(hObject,'String',num);
%     beep
%     errordlg('Input must be a number', 'Error')
end
handles.edit_P3_dep = num;
guidata(hObject,handles)

function edit_P3_dep_CreateFcn(hObject, eventdata, handles)

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit_P1_DR_Callback(hObject, eventdata, handles)
num = str2double(get(hObject,'String'));
if isnan(num)
    num = 0;
    set(hObject,'String',num);
%     beep
%     errordlg('Input must be a number', 'Error')
end
handles.edit_P1_DR = num;
guidata(hObject,handles)

function edit_P1_DR_CreateFcn(hObject, eventdata, handles)

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit_P2_DR_Callback(hObject, eventdata, handles)
num = str2double(get(hObject,'String'));
if isnan(num)
    num = 0;
    set(hObject,'String',num);
%     beep
%     errordlg('Input must be a number', 'Error')
end
handles.edit_P2_DR = num;
guidata(hObject,handles)
function edit_P2_DR_CreateFcn(hObject, eventdata, handles)

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
function edit_P3_DR_Callback(hObject, eventdata, handles)
num = str2double(get(hObject,'String'));
if isnan(num)
    num = 0;
    set(hObject,'String',num);
%     beep
%     errordlg('Input must be a number', 'Error')
end
handles.edit_P3_DR = num;
guidata(hObject,handles)

function edit_P3_DR_CreateFcn(hObject, eventdata, handles)

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% ================>>>>  PLOT PRC

function PRC_plot_Callback(hObject, eventdata, handles)

x=[handles.edit_P1_dep,...
   handles.edit_P2_dep,...
   handles.edit_P3_dep];

y=[handles.edit_P1_DR,...
   handles.edit_P2_DR,...
   handles.edit_P3_DR];

axes(handles.PRC_axes)
 plot(x,y,'--rs','LineWidth',2,...
                 'MarkerEdgeColor','k',...
                 'MarkerFaceColor','g',...
                 'MarkerSize',5);
 set(handles.PRC_axes,'XMinorTick','on')
 axis([0 110 0 max(y)+20]);
 xlabel('Depletion(%)');
 ylabel('Draw Rate (tonne/Period)');
 grid on
%**********************************************>> Save

function pushbutton_Save_Callback(hObject, eventdata, handles)
 if exist('MODEL/CPLEX.mat')== 2
   load('MODEL/CPLEX.mat');
else
end
             SchParam = struct('ND',0,...
                               'T',0,...
                               'i_rate',0,...
                               'Cost',0.0,...
                               'Mue',0.0,...
                               'McapLower',0.0,...
```

```
                                'McapUpper',0.0,...
                                'GradeLower',0.0,...
                                'GradeUpper',0.0,...
                                'NnewLower',0.0,...
                                'NnewUpper',0.0,...
                                'NactL',0.0,...
                                'NactU',0.0,...
                                'WorkingDays',0.0,...
                                'PRCLower',0.0,...
                                'PRCUpper',0.0,...
                                'PRCcurve',0.0);

     CPLEX.SchParam.nBigBl=handles.edit_ND;
     CPLEX.SchParam.T=handles.edit_Periods;
     CPLEX.SchParam.i_rate=handles.edit_i;
     CPLEX.SchParam.Mue = handles.edit_Mue;
     CPLEX.SchParam.L = handles.edit_M;
     CPLEX.SchParam.Cost=handles.edit_Cost;
     CPLEX.SchParam.McapLower=handles.edit_LowerCapacity;
     CPLEX.SchParam.McapUpper=handles.edit_UpperCapacity;
     CPLEX.SchParam.GradeLower=handles.edit_LowerGrade;
     CPLEX.SchParam.GradeUpper=handles.edit_UpperGrade;
     CPLEX.SchParam.NnewLower=handles.edit_LowerNnew;
     CPLEX.SchParam.NnewUpper=handles.edit_UpperNnew;
     CPLEX.SchParam.NactL=handles.edit_NactL;
     CPLEX.SchParam.NactU=handles.edit_NactU;
     CPLEX.SchParam.WorkingDays=handles.edit_Workingdays;
     CPLEX.SchParam.PRCLower=handles.edit_LowerPRC;
     CPLEX.SchParam.PRCUpper=handles.edit_UpperPRC;

CPLEX.SchParam.PRCcurve=[handles.edit_P1_dep,handles.edit_P2_dep,h
andles.edit_P3_dep;...

handles.edit_P1_DR,handles.edit_P2_DR,handles.edit_P3_DR ];
            save('MODEL/CPLEX','CPLEX');
                close
            clear all

%*********************************************************>> Cancel
function pushbutton_Cancel_Callback(hObject, eventdata, handles)
close
%
END==================================================================

-------------------------------------------------------------------

function ObjFuncGeneralAllAddingBandY()
load('MODEL\MainModelInput.mat')
load('MODEL\CPLEX.mat')
prompt = 'What are the production level? ';
InData = input(prompt);
noLevel = InData(1,1);
T = CPLEX.SchParam.T ;
r = CPLEX.SchParam.i_rate/100;
name = strcat('Level_',num2str(noLevel));
Data4Cons = cell(1,10);
% 1st col: level number
% 2nd col: T
% 3rd col: Tonnage (vertical vector)
% 4th col: Grade (vertical vector)
% 5th col: Profit (vertival vector)
% 6th col: Mining capacity lower bound
% 7th col: Mining capacity upper bound
% 8th col: Grade lower bound
% 9th col: Grade upper bound
% 10th col: DR upper bound
if isfieldRecursive(MainModelInput,'NotInclude',name) == 1
%Data = strcat('MainModelInput.','NotInclude.',name);
Data = MainModelInput.NotInclude.Level_39;
% Prepare data for constraint creation
Data4Cons{1,1} = noLevel;
Data4Cons{1,2} = T;
Data4Cons{1,3} = (Data(:,4));
Data4Cons{1,4} = (Data(:,5));
Data4Cons{1,5} = (Data(:,6));
save('Data4Cons','Data4Cons')
Profit = (Data(:,6))';
noBigBl = CPLEX.SchParam.nBigBl;
f = zeros(1,(3*T*noBigBl));
for t = 1:T
    f(1,(t-1)*noBigBl+1:t*noBigBl) = -Profit(1,:)./(1+r)^t;
%creating coefficient matrix for the objective function
end
fnorm = norm(f);
f = f/fnorm;
CPLEX.Inputs.ObjectiveFunction.f = f;
CPLEX.Inputs.Parameters.fnorm = fnorm;
save('MODEL/CPLEX','CPLEX');
      createMode.Interpreter='tex';
      createMode.WindowStyle='modal';
      msgbox('\bfObjective functuion was created
successfully.','saha Model - V1.0 ','help',createMode);
else
    errordlg('There is no data for the selected level','Data
Error');
end
-------------------------------------------------------------------
function Binary_Constraints_AddingY
load('MODEL\MainModelInput.mat')
if exist('MODEL/CPLEX.mat')== 2
   load('MODEL/CPLEX.mat');
else
end
prompt = 'What are the production level? ';
InData = input(prompt);
noLevel = InData(1,1);
name = strcat('Level_',num2str(noLevel));
T = CPLEX.SchParam.T;    %number of periods
if isfieldRecursive(MainModelInput,'NotInclude',name) == 1
Data = MainModelInput.NotInclude.Level_39;
N = CPLEX.SchParam.nBigBl;  %number of Blocks

Aineq_Binary_2_1 = sparse(zeros(N*T,N*T));
Aineq_Binary_2_2 = sparse(zeros(N*T,N*T));
```

```
Aineq_Binary_2_3 = sparse(zeros(N*T,N*T)); %Y part
Aineq_Binary_3 = zeros(N*(T-1),N*T);

A1_2 = repmat(1,N*T,1);

Aineq_Binary_2_1(1:N*T,1:N*T) = diag(A1_2); %the first part of the
second binary equation, creating matrix for the x-b<=0 creating
the coefficient for x
Aineq_Binary_2_2(1:N*T,1:N*T) = diag(-A1_2); %the second part of
the second binary equation, creating matrix for the x-b<=0
creating the coefficient for b

%creating the coefficient matrix for the third constraint
for t = 1:N*(T-1)
    Aineq_Binary_3(t,t) = 1;
    Aineq_Binary_3(t,t+N) = -1;
end
%concatenation
Aineq_Binary_2 =
[Aineq_Binary_2_1,Aineq_Binary_2_2,Aineq_Binary_2_3]; % x-b<=0
pp = sparse(zeros(N*(T-1),N*T));
ppp = sparse(zeros(N*(T-1),N*T));
Aineq_Binary_3 = [pp,Aineq_Binary_3,ppp];

bineq_Binary_2 = sparse(zeros(N*T,1)); %right hand side matrix for
the second equation x-b<=0
bineq_Binary_3 = sparse(zeros(N*(T-1),1)); %right hand side matrix
for the third equation b(n,t)-b(n,t+1)<=0
Aineq_Binary = [Aineq_Binary_2;Aineq_Binary_3];
bineq_Binary = [bineq_Binary_2;bineq_Binary_3];
CPLEX.Inputs.Constraints.Aineq_Binary = Aineq_Binary;
CPLEX.Inputs.Constraints.bineq_Binary = bineq_Binary;
save('MODEL/CPLEX','CPLEX');
      createMode.Interpreter='tex';
      createMode.WindowStyle='modal';
      msgbox('\bfBinary constraints were created
successfully.','saha Model - V1.0 ','help',createMode);
else
    errordlg('There is no data for the selected level','Data
Error');
end
end
------------------------------------------------------------------
function MCCGeneralBinary_AddingY()
load('MODEL\MainModelInput.mat')
load('Data4Cons.mat')
if exist('MODEL/CPLEX.mat')== 2
   load('MODEL/CPLEX.mat');
else
end
N = CPLEX.SchParam.nBigBl;
T = CPLEX.SchParam.T;
Tonnage = (Data4Cons{1,3})';
Data4Cons{1,6} = CPLEX.SchParam.McapLower;
Data4Cons{1,7} = CPLEX.SchParam.McapUpper;
BL = Data4Cons{1,6};
```

```
BU = Data4Cons{1,7};
save('Data4Cons','Data4Cons')
% Coefficient matrix
A_Mcap_Uhelp = zeros(T,3*T*N);
A_Mcap_Lhelp = zeros(T,3*T*N);
for t = 1:T
%Upper
A_Mcap_Uhelp(t,(t-1)*N+1:t*N) = Tonnage;
%Lower
A_Mcap_Lhelp(t,(t-1)*N+1:t*N) = (-1)*Tonnage;
end
A_Mcap_U = A_Mcap_Uhelp ./norm(A_Mcap_Uhelp);
A_Mcap_L = A_Mcap_Lhelp ./ norm(A_Mcap_Lhelp);
%Boundaries
BU_Mcap = (BU*ones(T,1))./norm(A_Mcap_Uhelp);
BL_Mcap = (-BL*ones(T,1))./ norm(A_Mcap_Lhelp);
load('MODEL/CPLEX.mat')
CPLEX.Inputs.Cons.AMcap = [A_Mcap_U;A_Mcap_L] ;
CPLEX.Inputs.Cons.BMcap = [BU_Mcap;BL_Mcap];
save('MODEL/CPLEX','CPLEX');
clear all
      createMode.Interpreter='tex';
      createMode.WindowStyle='modal';
      msgbox('\bfMining capacity constraint was created
successfully.','Saha Model - V1.0 ','help',createMode);
```

```
function GCBinary_AddingY()
load('MODEL\MainModelInput.mat')
load('Data4Cons.mat')
if exist('MODEL/CPLEX.mat')== 2
   load('MODEL/CPLEX.mat');
else
end
N = CPLEX.SchParam.nBigBl;
T = CPLEX.SchParam.T;
Tonnage = (Data4Cons{1,3})';
Grade = (Data4Cons{1,4})';
BL = CPLEX.SchParam.GradeLower;
BU = CPLEX.SchParam.GradeUpper;
Data4Cons{1,8} = CPLEX.SchParam.GradeLower;
Data4Cons{1,9} = CPLEX.SchParam.GradeUpper;
save('Data4Cons','Data4Cons')
% Coefficient matrix
A_GU_help = zeros(T,3*T*N);
A_GL_help = zeros(T,3*T*N);
%Upper *****************************************
GupVector = BU * ones(1,N);
% Gbl-Gup
GblGup = Grade - GupVector;
% (Gbl-Gup)*tonnage
TempUpper = GblGup .* Tonnage;
%Lower *****************************************
GlVector = BL * ones(1,N);
% Gbl-Gup
GblGl = GlVector - Grade;
% (Gl-Gbl)*tonnage
TempLower = GblGl .* Tonnage;
```

```
for t = 1:T
%Upper
A_GU_help(t,(t-1)*N+1:t*N) = TempUpper;
%Lower
A_GL_help(t,(t-1)*N+1:t*N) = TempLower;
end
% join A_Gcap_U and A_Gcap_L
AGNOTnormalized = [A_GU_help;A_GL_help];
% Norm of Each Line (NEL)
NEL_AGNOTnormalized = sqrt(sum(abs(AGNOTnormalized).^2,2));
                A_Grade=[];
                ChunkSize=300;
                Chunks = ceil(size(AGNOTnormalized,2)/ChunkSize);
                for iLoop=1:Chunks
                        range = ((iLoop-1) * ChunkSize + 1):
(min((iLoop * ChunkSize),size(AGNOTnormalized,2)));
                        Norm_matrix_help =
repmat(NEL_AGNOTnormalized,1,size(range,2));
                        tempL =
AGNOTnormalized(:,range)./Norm_matrix_help;
                        A_Grade = [A_Grade,tempL];
                end
                close
%Boundaries
BU_Grade = zeros(T,1);
BL_Grade = zeros(T,1);
load('MODEL/CPLEX.mat')
CPLEX.Inputs.Cons.AGrade = A_Grade;
CPLEX.Inputs.Cons.BGrade = [BU_Grade;BL_Grade];
save('MODEL/CPLEX','CPLEX');
clear all
      createMode.Interpreter='tex';
      createMode.WindowStyle='modal';
      msgbox('\bfGrade constraint was created successfully.','Saha
Model - V1.0 ','help',createMode);
```

```
function DrawRateBinary_AddingY2()
load('MODEL\MainModelInput.mat')
load('Data4Cons.mat')
if exist('MODEL/CPLEX.mat')== 2
   load('MODEL/CPLEX.mat');
else
end
N = CPLEX.SchParam.nBigBl;
T = CPLEX.SchParam.T;
Tonnage = (Data4Cons{1,3})';
%Grade = (Data4Cons{1,4})';
L = CPLEX.SchParam.L;
% Drawrate Upper bound
DR_U = CPLEX.SchParam.PRCUpper;
Data4Cons{1,10} = DR_U;
save('Data4Cons','Data4Cons')
% Drawrate Lower bound
DR_L = CPLEX.SchParam.PRCLower;
% Coefficient matrix : size for the matrix is [N*T, N*T] we have
to write
% this constraint for each drawpoint
```

```
A_DR_help = zeros(T*N,3*T*N);
%>>>>>>>>>>>>>>>>>>>>>>>>
A_DR_help2 = zeros(T*N,3*T*N);
%>>>>>>>>>>>>>>>>>>>>>>>
A_DR_help3 = zeros(T*N,3*T*N);
counter =0;
for t = 1:T
    for BL = 1:N
        A_DR_help(BL+counter,BL+counter) = Tonnage(1,BL);
%x*Ton<=DR_U (x)
        %>>>>>
        A_DR_help2(BL+counter,BL+counter) = -Tonnage(1,BL);
%B*DR_L-xTon-LY<=0 (x)
        A_DR_help2(BL+counter,BL+counter+(N*T)) = DR_L ; %B*DR_L-
xTon-LY<=0 (B)
        A_DR_help2(BL+counter,BL+counter+2*(N*T)) = -L; %B*DR_L-
xTon-LY<=0 (Y)
        %>>>>>
        A_DR_help3(BL+counter,BL+counter+2*(N*T)) = 1; %Y-
sigma(x)<=0 (Y)
        for a = 1:t
            A_DR_help3(BL+counter,BL+(a-1)*N) = -1; %Y-sigma(x)<=0
(x)
        end
    end
    counter = counter+N;

    stopBar= progressbar(BL / N ,0);
    if (stopBar) break; end
close
end
% The A_DR_help matrix has not been normalized, so it has to be
normalized.
% Norm of each line (NEL)
NEL_A_DR_help = sqrt(sum(abs(A_DR_help).^2,2));
%>>>>>>>>>>>
NEL_A_DR_help2 = sqrt(sum(abs(A_DR_help2).^2,2));
%>>>>>>>>>>>>>>>
NEL_A_DR_help3 = sqrt(sum(abs(A_DR_help3).^2,2));
% Sometimes we have to deal with a big size problems, to solve
this kind of problems it is better we use the ChunkSize method.
ADRhelp_Normalized = [];
%>>>>>>>>
ADRhelp2_Normalized = [];
%>>>>>>>
ADRhelp3_Normalized = [];
ChunkSize=200;
Chunks = ceil(size(A_DR_help,2)/ChunkSize);
   for iLoop = 1:Chunks
        range = ((iLoop-1) * ChunkSize + 1): (min((iLoop *
ChunkSize),size(A_DR_help,2)));
        Norm_matrix_help = repmat(NEL_A_DR_help,1,size(range,2));
        tempU = A_DR_help(:,range)./Norm_matrix_help;

        ADRhelp_Normalized = [ADRhelp_Normalized,tempU];
```

```
          stopBar= progressbar(iLoop / Chunks,0);
          if (stopBar) break; end
   end
   close

   A_DR1 = ADRhelp_Normalized;
% Upper bound
DR_Ub1 = (DR_U*ones(T*N,1))./NEL_A_DR_help;
%>>>>>>>>>>>>>>>>>
Chunks = ceil(size(A_DR_help2,2)/ChunkSize);
   for iLoop = 1:Chunks
        range = ((iLoop-1) * ChunkSize + 1): (min((iLoop *
ChunkSize),size(A_DR_help2,2)));
        Norm_matrix_help2 =
repmat(NEL_A_DR_help2,1,size(range,2));
        tempU = A_DR_help2(:,range)./Norm_matrix_help2;
        ADRhelp2_Normalized = [ADRhelp2_Normalized,tempU];
          stopBar= progressbar(iLoop / Chunks,0);
          if (stopBar) break; end
   end
   close
   A_DR2 = ADRhelp2_Normalized;
% Upper bound
DR_Ub2 = zeros(T*N,1);
%>>>>>>>>>>>>>>
Chunks = ceil(size(A_DR_help3,2)/ChunkSize);
   for iLoop = 1:Chunks
        range = ((iLoop-1) * ChunkSize + 1): (min((iLoop *
ChunkSize),size(A_DR_help3,2)));
        Norm_matrix_help3 =
repmat(NEL_A_DR_help3,1,size(range,2));
        tempU = A_DR_help3(:,range)./Norm_matrix_help3;
        ADRhelp3_Normalized = [ADRhelp3_Normalized,tempU];
          stopBar= progressbar(iLoop / Chunks,0);
          if (stopBar) break; end
   end
   close
   A_DR3 = ADRhelp3_Normalized;
% Upper bound
DR_Ub3 = zeros(T*N,1);
%>>>>>>>>>>>>>>
A_DR = [A_DR1;A_DR2;A_DR3];
DR_Ub = [DR_Ub1;DR_Ub2;DR_Ub3];
load('MODEL/CPLEX.mat')
CPLEX.Inputs.Cons.ADR = A_DR;
CPLEX.Inputs.Cons.BDR = DR_Ub;
save('MODEL/CPLEX','CPLEX');
clear all
      createMode.Interpreter='tex';
      createMode.WindowStyle='modal';
      msgbox('\bfDrawrate constraint was created
successfully.','Saha Model - V1.0 ','help',createMode);
```

```
function ReserveBinary_AddingY()

% total ore material within each block must be extracted.
% The main coefficien matrix contains (N*T)columns.
% Let say we have 5 blocks and 3 periods, so matrix A has 15
columns.
% First 5 columns indicate to the period 1.
% Second 5 columns indicate to the period 2.
% Third 5 columns indicate to the period 3.
% Dimension of matrix A for this constraint is N*(N*T). We have T
periods
% so for each period we have N*N matrix.
%
%       <-Period1-><-Period2-><-Period3->
%      | 1 0 0 0 0 |1 0 0 0 0 | 1 0 0 0 0|
%      | 0 1 0 0 0 |0 1 0 0 0 | 0 1 0 0 0|
%   A= | 0 0 1 0 0 |0 0 1 0 0 | 0 0 1 0 0|
%      | 0 0 0 1 0 |0 0 0 1 0 | 0 0 0 1 0|
%      | 0 0 0 0 1 |0 0 0 0 1 | 0 0 0 0 1|
%      <---  A1 --->
% It can be seen clearly, if we just make a 5*5  matrix with 1's
on the diagonal and 0's elsewhere, then we can construct matrix A
using matrix A1.
load('Data4Cons.mat')
if exist('MODEL/CPLEX.mat')== 2
   load('MODEL/CPLEX.mat');
else
end
N = CPLEX.SchParam.nBigBl;
T = CPLEX.SchParam.T;
A1 = eye(N,N);
ARes = repmat(A1,1,T);
pp = zeros(N,2*N*T);
ARes = [ARes,pp];
BRes_U = ones(N,1);
load('MODEL/CPLEX.mat')
CPLEX.Inputs.Cons.ARes = ARes;
CPLEX.Inputs.Cons.BRes = BRes_U;
save('MODEL/CPLEX','CPLEX');
clear all
      createMode.Interpreter='tex';
      createMode.WindowStyle='modal';
      msgbox('\bfReserve constraint was created
successfully.','Saha Model - V1.0 ','help',createMode);
------------------------------------------------------------------
function  Adj_Block_Indices_BigBLs
prompt = 'What is the production level? ';
noLevel = input(prompt);
name = strcat('Level_',num2str(noLevel));
load('MODEL\MainModelInput.mat');
load('MODEL\CPLEX.mat');
OreBlocks =
[MainModelInput.NotInclude.Level_39(:,2),MainModelInput.NotInclude
.Level_39(:,3)];
X_BLs_Vvector = MainModelInput.NotInclude.Level_39(:,2);
Y_BLs_Vvector = MainModelInput.NotInclude.Level_39(:,3);
X_BLs_Hvector = MainModelInput.NotInclude.Level_39(:,2)';
Y_BLs_Hvector = MainModelInput.NotInclude.Level_39(:,3)';
NBLs = CPLEX.SchParam.nBigBl;        % # of Blocks
%====================calculating distance between Blocks
```

```
%======= Create dX^2
X_BLS_1 = repmat(X_BLs_Hvector,NBLs,1);
X_BLS_2 = repmat(X_BLs_Vvector,1,NBLs);
dX = (X_BLS_1)-(X_BLS_2);
dX_square = dX.^2;
%========================== calculating distance between Blocks
%========Create dY^2
Y_BLS_1 = repmat(Y_BLs_Hvector,NBLs,1);
Y_BLS_2 = repmat(Y_BLs_Vvector,1,NBLs);
dY = (Y_BLS_1)-(Y_BLS_2);
dY_square = dY.^2;
%===================================  DISTANCE^2=dX^2+dY^2
Distance_square = dX_square+dY_square;
%========================================= Distance between Blocks
Distance_BLs = sqrt(Distance_square);
InData.Distance_BLs = Distance_BLs;
save('Results/InData','InData');
load 'InData.mat';
Distance_BLs = InData.Distance_BLs;
maxdist = 43;
BLs_x =  MainModelInput.NotInclude.Level_39(:,2) ;
N = CPLEX.SchParam.nBigBl;
indices = [];
indices = cell(N,3);
for i = 1:N
    temp = find(Distance_BLs(i,:)<=maxdist & Distance_BLs(i,:)>0);
    indices{i,1} = temp;
    indices{i,2} = X_BLs_Hvector(temp);
    indices{i,3} = Y_BLs_Hvector(temp);
end
InData.indices = indices;
save('Results/Indata','InData');
end
```

```
function MiningDirection
load('Results/InData','InData')
load('MODEL/CPLEX.mat')
prompt1 = 'What is the coordinates of point1? ';
Point1 = input(prompt1);
prompt2 = 'What is the coordinates of point2? ';
Point2 = input(prompt2);
%calculating the slope of the mining direction
dify = Point2(1,2)-Point1(1,2);
difx = Point2(1,1)-Point1(1,1);
SlopeMD = dify/difx;
%calculating the slope of the perpendicular line
SlopePL = -1/SlopeMD;
prompt = 'What is the production level? ';
noLevel = input(prompt);
name = strcat('Level_',num2str(noLevel));
load('MODEL\MainModelInput.mat')
X = MainModelInput.NotInclude.Level_39(:,2);
Y = MainModelInput.NotInclude.Level_39(:,3);
Xnew = 200;
N = CPLEX.SchParam.nBigBl;
AdjecentIndices = InData.indices(:,1);
X_Adjecent = InData.indices(:,2);
```

```
Y_Adjecent = InData.indices(:,3);
for i = 1:N
for adjecent = 1:numel(AdjecentIndices(i,1))
    Ynew(i,1) = SlopePL.*(Xnew-X(i,1))+Y(i,1);
    d = cell(2,adjecent);
    d{1,:} = (X_Adjecent{i,1} - X(i,1)).*(Ynew(i,1) - Y(i,1))-
(Y_Adjecent{i,1} - Y(i,1)).*(Xnew - X(i,1));
    d{2,:} = AdjecentIndices{i,1};
    Data1 = d{1,:};
    Data2 = d{2,:};
    B{i,adjecent} = Data2(find(Data1<0)); %Adjacent below mining
direction
end
end
InData.B = B;
save('Results/InData','InData');
end
```

```
function Precedence_AddingY
if exist('MODEL/CPLEX.mat')== 2
    load('MODEL/CPLEX.mat');
else
end
prompt = 'What are the production level? ';
InData = input(prompt);
noLevel = InData(1,1);
T = CPLEX.SchParam.T;
name = strcat('Level_',num2str(noLevel));
load('MODEL\MainModelInput.mat')
N = CPLEX.SchParam.nBigBl;
load('Results/InData','InData');
AdjecentsBelowMiningDirection = InData.B;
Precedence1 = zeros(N*T,N*T);
Precedence2 = zeros(N*T,N*T);
Precedence3 = zeros(N*T,N*T); %Because of Adding Y
for jLoop = 1:T
    for iLoop = 1:N
        n(iLoop,1) =
numel(AdjecentsBelowMiningDirection{iLoop,1});
        Precedence1((jLoop-1)*N+iLoop,(jLoop-1)*N+iLoop) =
n(iLoop,1);
        for kLoop =
1:numel(AdjecentsBelowMiningDirection{iLoop,1});
            Precedence1((jLoop-1)*N+iLoop,(jLoop-
1)*N+AdjecentsBelowMiningDirection{iLoop,1}(1,kLoop)) = -1;
        end
    end
end
APrecedence_Binary = [Precedence2,Precedence1,Precedence3];
bPrecedence_Binary = sparse(zeros(N*T,1));
CPLEX.Inputs.Constraints.APrecedence_Binary = APrecedence_Binary;
CPLEX.Inputs.Constraints.bPrecedence_Binary = bPrecedence_Binary;
save('MODEL/CPLEX','CPLEX');
createMode.Interpreter='tex';
createMode.WindowStyle='modal';
```

```
msgbox('\bfPrecedence constraints was created successfully.','saha
Model - V1.0 ','help',createMode);
```

```
function NumberOfNewBBLs
if exist('MODEL/CPLEX.mat')== 2
   load('MODEL/CPLEX.mat');
else
end
N = CPLEX.SchParam.nBigBl;
T = CPLEX.SchParam.T;
prompt1 = 'What is maximum number of new big blocks in period 1?';
NoNewBBLsPeriod1_U = input(prompt1);
prompt2 = 'What is maximum number of new big blocks after
period1?';
NoNewBBLsPeriod2On_U = input(prompt2);
prompt3 = 'What is minimum number of new big blocks in period 1?';
NoNewBBLsPeriod1_L = input(prompt3);

prompt4 = 'What is minimum number of new big blocks after
period1?';
NoNewBBLsPeriod2On_L = input(prompt4);
NoNewBBLsHelp = zeros(T,3*N*T);
NoNewBBLsHelp1 = zeros(T,3*N*T);
counter = 0;
for t = 1:T
    for BL = 1:N
        if t == 1
            NoNewBBLsHelp(t,(N*T)+BL) = 1;
            NoNewBBLsHelp1(t,(N*T)+BL) = -1;
        else
           NoNewBBLsHelp(t,BL+counter+(T*N)) = 1;
           NoNewBBLsHelp(t,(t-2)*N+BL+(N*T)) = -1;
           NoNewBBLsHelp1(t,BL+counter+(T*N)) = -1;
           NoNewBBLsHelp1(t,(t-2)*N+BL+(N*T)) = 1;
        end
    end
    counter = counter+N;
end
A_NoNewBBLs = [NoNewBBLsHelp;NoNewBBLsHelp1];

BFirstPeriod_U = NoNewBBLsPeriod1_U;
BSecondPeriodOn_U = NoNewBBLsPeriod2On_U*ones((T-1),1);
B_NoNewBBLs_U = [BFirstPeriod_U;BSecondPeriodOn_U];

BFirstPeriod_L = NoNewBBLsPeriod2On_L;
BSecondPeriodOn_L = -NoNewBBLsPeriod2On_L*ones((T-1),1);
B_NoNewBBLs_L = [BFirstPeriod_L;BSecondPeriodOn_L];
% Norm of each line (NEL)
NEL_A = sqrt(sum(abs(A_NoNewBBLs).^2,2));
Ahelp_Normalized = [];
ChunkSize=200;
Chunks = ceil(size(A_NoNewBBLs,2)/ChunkSize);
   for iLoop = 1:Chunks
        range = ((iLoop-1) * ChunkSize + 1): (min((iLoop *
ChunkSize),size(A_NoNewBBLs,2)));
        Norm_matrix_help = repmat(NEL_A,1,size(range,2));
        tempU = A_NoNewBBLs(:,range)./Norm_matrix_help;
        Ahelp_Normalized = [Ahelp_Normalized,tempU];
          stopBar= progressbar(iLoop / Chunks,0);
          if (stopBar) break; end
   end
   close
   A_New = Ahelp_Normalized;
B_New = [B_NoNewBBLs_U;B_NoNewBBLs_L];
B_New_Nomarlized = B_New./NEL_A;
load('MODEL/CPLEX.mat')
CPLEX.Inputs.Cons.ANoNewBBLs = A_New;
CPLEX.Inputs.Cons.BNoNewBBLs = B_New_Nomarlized;
save('MODEL/CPLEX','CPLEX');
createMode.Interpreter='tex';
      createMode.WindowStyle='modal';
      msgbox('\bfNumber of New BBLs constraint was created
successfully.','Saha Model - V1.0 ','help',createMode);
end
-------------------------------------------------------------------
function RunModelBinaryWithEPGAP()
load('MODEL\CPLEX.mat')
load('Data4Cons.mat')
T = CPLEX.SchParam.T;
N = CPLEX.SchParam.nBigBl;
ton = Data4Cons{1,3}'; % Horizontal vector
grade = Data4Cons{1,4}'; % Horizontal vector
f = CPLEX.Inputs.ObjectiveFunction.f;
fnorm = CPLEX.Inputs.Parameters.fnorm;
A1 = CPLEX.Inputs.Cons.AMcap;
B1 = CPLEX.Inputs.Cons.BMcap;
A2 = CPLEX.Inputs.Cons.ADR;
B2 = CPLEX.Inputs.Cons.BDR;
A3 = CPLEX.Inputs.Cons.AGrade;
B3 = CPLEX.Inputs.Cons.BGrade;
A4 = CPLEX.Inputs.Constraints.Aineq_Binary;
B4 = CPLEX.Inputs.Constraints.bineq_Binary;
A5 = CPLEX.Inputs.Constraints.APrecedence_Binary;
B5 = CPLEX.Inputs.Constraints.bPrecedence_Binary;
A6 = CPLEX.Inputs.Cons.ANoNewBBLs;
B6 = CPLEX.Inputs.Cons.BNoNewBBLs;
Aineq = [A1;A2;A3;A4;A5;A6];
bineq = [B1;B2;B3;B4;B5;B6];
Aineq = full(Aineq);
bineq = full(bineq);
Aeq = [CPLEX.Inputs.Cons.ARes];
beq = [CPLEX.Inputs.Cons.BRes];
% X vector size
xSize = size(Aineq,2);
lb = zeros(xSize,1);
ub = ones(xSize,1);
ctype_1 = [char('C'*ones(1,N*T))]; %Defining type of the
variables, 1:N variables are Continues and N+1:2N are Binary
ctype_2 = [char('B'*ones(1,N*T))];
ctype_3 = [char('B'*ones(1,N*T))];
ctype = [ctype_1,ctype_2,ctype_3];
```

```
addpath('C:\Program
Files\IBM\ILOG\CPLEX_Enterprise_Server126\CPLEX_Studio\cplex\matla
b\x64_win64');
addpath('C:\Program
Files\IBM\ILOG\CPLEX_Enterprise_Server126\CPLEX_Studio\cplex\examp
les\src\matlab');
%*****************************************************************
% Model Information
A = [Aineq;Aeq];
[m,n] = size(A);
fprintf('Block cave Scheduling Problem. Variables %d Constraints
%d\n',n,m);
%*****************************************************************
%The CPLEX run function
%[x,fval,exitflag,output] =
cplexmilp(f,Aineq,bineq,Aeq,beq,[],[],[],lb,ub,ctype,[],options)
%solution
%[x,fval,exitflag,output] =
cplexlp(f,Aineq,bineq,Aeq,beq,lb,ub,[],options) %solution
options = cplexoptimset;
   options.Display = 'on';
   options.parameter2009 = 0.01;
[x,fval,exitflag,output] =
cplexmilp(f,Aineq,bineq,Aeq,beq,[],[],[],lb,ub,ctype,[],options);
%solution
output
Valu = -fval*fnorm
%**************************************************
%saving the results
CPLEX.Results.out1 = Cplex();
CPLEX.Results.out2 = CpxInfo(Cplex);
   fprintf ('\nSolution status = %s \n',
output.cplexstatusstring);
   fprintf ('Solution value = %f \n', fval);
CPLEX.Outputs.x = x;
CPLEX.Outputs.fval = fval*fnorm;
CPLEX.Outputs.exitflag = exitflag;
CPLEX.Outputs.output = output;
%preparing the results(x) based on the periods,
%the first N decision variables(x1:xN) are for the first period
and so on.
for iloop = 1:T
    y(iloop,:) = x((iloop-1)*N+1:N*iloop,1);
end
CPLEX.Outputs.y = y;
BSolution = x((T*N)+1:(2*T*N));
for jloop = 1:T
    BPeriod(jloop,:) = BSolution((jloop-1)*N+1:N*jloop,1);
end
CPLEX.Outputs.BPeriod = BPeriod;
YSolution = x((2*T*N)+1:(3*T*N));
for jloop = 1:T
    YPeriod(jloop,:) = YSolution((jloop-1)*N+1:N*jloop,1);
end
CPLEX.Outputs.YPeriod = YPeriod;

%---------------Plot Total Production per Period------------------
```

```
if exist('HelpPlot.mat') == 2
    load('HelpPlot.mat')
end
%***************************************** Tonnage
TonTemp = repmat(ton,T,1);
BLPeriodicProduction = TonTemp .* y; %total production per each
period(tonne), it will be used to plot the "production per period"
graph
CPLEX.Outputs.BLPeriodicProduction = BLPeriodicProduction;
HelpPlot.BLPeriodicProduction = BLPeriodicProduction;
fprintf('Initial tonnage before optimization = %d
tonne\n',sum(ton));
fprintf('Extracted tonnege after optimization = %d
tonne\n',sum(sum(BLPeriodicProduction)));

%--------------------Plot Average Grade Per Period----------------
%***************************************** Grade
GrTemp = repmat(grade,T,1);
% grade * tonnage
GrTon = GrTemp.* BLPeriodicProduction;
Sigma_GrTon = sum(GrTon,2);
SigmaTon = sum(BLPeriodicProduction,2);
AvgGradeperiod = Sigma_GrTon ./ SigmaTon;
HelpPlot.AvgGradeperiod =AvgGradeperiod;
%----------Plot Number of Active and New Big Blocks in each Period
%************Calculating Number of Active Blocks in each period
y = CPLEX.Outputs.y;
ActiveBl = zeros(T,N);
for iLoop = 1:N
ActiveBl(:,iLoop) = (y(:,iLoop)>0);
end
NoActiveBlPeriod = sum(ActiveBl,2);

%***********Calculating Number of New Blocks in each period
BinaryVariablesPeriod = CPLEX.Outputs.BPeriod;
OpeningPeriod = zeros(T-1,N);
for iLoop = 1:T-1
    OpeningPeriod(iLoop,:) = BinaryVariablesPeriod(iLoop+1,:)-
BinaryVariablesPeriod(iLoop,:);
end
OpeningPeriodWithFirstPeriod =
[BinaryVariablesPeriod(1,:);OpeningPeriod];
NoNewBlPeriod = sum(OpeningPeriodWithFirstPeriod,2);
TwoDataforPlot = [NoActiveBlPeriod,NoNewBlPeriod];
HelpPlot.TwoDataforPlot = TwoDataforPlot;
%---------------------Plot Opening Period for each Big Block
BinaryVariablesPeriod = CPLEX.Outputs.BPeriod;
OpeningPeriod = zeros(T-1,N);
for iLoop = 1:T-1
    OpeningPeriod(iLoop,:) = BinaryVariablesPeriod(iLoop+1,:)-
BinaryVariablesPeriod(iLoop,:);
end
OpeningPeriodWithFirstPeriod =
[BinaryVariablesPeriod(1,:);OpeningPeriod];
NoOpeningPeriod = zeros(1,N);
for jLoop = 1:N
```

```
    Temp = find(OpeningPeriodWithFirstPeriod(:,jLoop));
    NoOpeningPeriod(1,jLoop) = Temp(1,1);
end
save('OpeningPeriodWithFirstPeriod','OpeningPeriodWithFirstPeriod'
)
HelpPlot.NoOpeningPeriod = NoOpeningPeriod;
CPLEX.Outputs.HelpPlot = HelpPlot;
save('MODEL/CPLEX','CPLEX');
Test_variables
      createMode.Interpreter='tex';
      createMode.WindowStyle='modal';
      msgbox('\bfThe problem was solved.','Saha Model - V1.0
','help',createMode);
end
```