# Stopes Layout and Production Scheduling Optimization in Sublevel Stoping Mining

by

Zeinab Basiri

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

*The economics of today's mining industry is such that the major mining companies are increasing the use of massive underground mining methods. Follow to this attraction, mine planning in underground mining and its optimization have been considered more seriously in recent years. Two aspects of mine planning optimization in underground mining are stopes layout and production scheduling optimization. Few algorithms have been presented to optimize the stopes layout and stope production scheduling for underground mining; however, those are not able to provide an optimum solution. Most of the presented algorithms are heuristic so they cannot guarantee to achieve the optimum solution.*

*The objective of this study is to present the mathematical formulations to find the optimum stopes layout and production scheduling in sublevel stoping method. The stopes layout and production scheduling optimization are applied based on the total blocks in the block model altogether ( LOT and SOT methods)and based on the separated levels of block model (LOL and SOL methods) to see the impact of leveling in the stopes layout and production scheduling optimization which has not been clarified in the previous researches.*

*The proposed methodologies maximize the economic value (EV) to determine the stopes layout. The presented production scheduling algorithm maximizes the NPV over the life of mine by using the Binary Integer Programming (BIP) while honoring the constraints such as only one-time mining, stopes adjacency, the connection between mining the stopes and activation of levels, concurrent active levels, and the delay between activation of the levels. The methods have been applied to a block model to check the application of the model in real size problems. Achieved EV by LOT method is 4% higher than LOL method; however, the running time of LOT method is 418*

*times more than LOL method. Also, from the accessibility of production levels to all the stopes during mining point of view, LOL presented the practical stops layout. The NPV of SOT method is 22% higher than the NPV of SOL method; however, the running time of SOT method is 3.4 times more than SOL method.*

*The main contributions of this research are determining the mathematical models with the optimum solution to create the optimal stopes layout and production scheduling by two methods and comparing the results of the methods. Considering the practicality of the models by defining the set of practical constraints suitable for sublevel stoping and paying attention to the constraints that were not considered in the previous works include concurrent active levels and the delay between activation of the levels are other important contributions.*

*This Thesis is Proudly Dedicated To:*

*My husband,* ***Ahmad***
*I am truly thankful for having you in my life.*

*&*

*My son,* ***Ryan***
*I love you to the moon and back. You do not know how precious you are to me.*

# ACKNOWLEDGMENT

First and foremost, I would like to express my sincere appreciation to my supervisor, *Dr. Yashar Pourrahimian,* for his invaluable support throughout my M.Sc. study and research. His friendship and professional relationship are very important to me. I am deeply grateful for his patience, enthusiasm, motivation and his immense knowledge.

I would like to thank my colleagues *Ali Yaghini, Ali Moradi, Firouz Khodayari, Roberto Noriega, Shahrokh Paravarzar* and especially *Dr. Mohammad Tabesh* for sharing their knowledge. Furthermore, I am thankful to my other colleagues for making the office a friendly environment to pursue our purposes. I am grateful for the insight and encouragement from all my friends in Edmonton. They supported me greatly and were always willing to help me.

Finally, I must express my very profound gratitude to my family: My beloved husband, *Ahmad* who stands beside me and provides and me with unfailing support and continuous encouragement throughout my years of study. My beautiful three-year-old son, *Ryan* whose smile and hug provide me unending inspiration and comfort me for my guilt from being away from him. My devoted parents, my father *Majid* and especially my mother *Mahvash* whose love, guidance and sacrifices are with me in whatever I pursue. They mean a lot to me. My wonderful brothers, *Mohammad, Mahdi* and *Masih* whom I cannot imagine the world without them. They support me whenever I need them. This accomplishment would not have been possible without you all. Thank you!

Zeinab Basiri
September 2018

# TABLE OF CONTENTS

# LIST OF TABLES

# LIST OF FIGURES

# LIST OF ABBREVIATIONS

| | |
|---|---|
| DEV | Discounted Economic Value |
| EPGAP | Relative MIP Gap Tolerance |
| EV | Economic Value |
| LOL | Layout optimization based on levels |
| LOT | Layout optimization based on total blocks |
| BIP | Binary Integer Programming |
| NPV | Net Present Value |
| SLS | Sublevel Stopping method in underground mining |
| SOL | Production scheduling optimization based on levels |
| SOT | Production scheduling optimization based on total stopes |

# LIST OF NOMENCLATURES

**Indices**

| | |
|---|---|
| $l \in \{1,...,L\}$ | Index for levels in LOL method |
| $l* \in \{1,...,L*\}$ | Index for selected levels as the output of LOL method |
| $s \in \{1,...,S\}$ | Index for stopes in LOT and LOL methods |
| $s* \in \{1,...,S*\}$ | Index for selected stopes as the output of LOL method |
| $s^*_{l*} \in \{1,...,S^*_{l*}\}$ | Index for selected stopes $s*$ in selected level $l*$ as the output of LOL method |
| $t \in \{1,...,T\}$ | Index for scheduling period in SOT method |
| $t_{l*} \in \{1,...,T_{l*}\}$ | Index for scheduling period in SOL method |

**Sets**

| | |
|---|---|
| $A_{s*}$ | Set of all stopes adjacency contains all adjacency for every selected stopes in SOT method |
| $A^{s*}_{l*}$ | Set of all stopes adjacency which contains all adjacency for every selected stopes of selected levels in SOL method |
| $B^{s*}_{l*}$ | Set of the stopes $s*$ in level $l*$ in SOT method |
| $Ol$ | Set of all levels overlaps contains all overlaps for each levels in LOL method |
| $Os$ | Set of all stopes overlaps contains all overlaps for each stope in LOT and LOL methods |

***Decision variables***

| | |
|---|---|
| $x_l \in \{0,1\}$ | Binary variable controlling the selection of levels in LOL method |

| | |
|---|---|
| $x_s \in \{0,1\}$ | Binary variable controlling the selection of stopes in LOT and LOL methods |
| $x_{s*}^{t} \in \{0,1\}$ | Binary variable controlling mining of stope s*in period t in SOT method |
| $x_{s*,l*}^{t_{l*}} \in \{0,1\}$ | Binary variable controlling mining of stope $s*$ of level $l*$ in period t in SOL method |
| $y^{t}_{l*} \in \{0,1\}$ | Binary variable controlling the activation of level $l*$ in period t in SOT method |

***Parameters***

| | |
|---|---|
| $BEV$ | Block economic value |
| $c_p$ | Cost of processing |
| $c_m$ | Cost of mining |
| $c_s$ | Cost of selling |
| $Cu_t$ | Upper bound of mining capacity in period $t$ in SOT method |
| $Cl_t$ | Lower bound of mining capacity in period $t$ in SOT method |
| $Cl_{l*}^{t}$ | Lower bound of mining capacity in period $t_{l*}$ for level $l*$ |
| $Cu_{l*}^{t}$ | Upper bound of mining capacity in period $t_{l*}$ for level $l*$ |
| $D$ | Required delay between activation of levels $l*$ |
| $EV_l$ | Economic value of level $l$ |
| $EV_s$ | Economic value of stope $s$ |
| $EV_{s*}$ | Economic value of stope $s*$ in SOT method |

| | |
|---|---|
| $EV_{l*}^{s*}$ | Economic value of stope $s*$ of level $l*$ in SOL method |
| $g$ | Block average grade |
| $G_{s*}$ | Average grade of stope $s*$ in SOT method |
| $G_{l*}^{s*}$ | Average grade of stope $s*$ of level $l*$ in SOL method |
| $Gl_t$ | Lower bound of acceptable average grade in period $t$ in SOT method |
| $Gu_t$ | Upper bound of acceptable average grade in period $t$ in SOT method |
| $Gu_{l*}^{t}$ | Upper bound of acceptable average grade in period $t_{l*}$ for level $l*$ |
| $Gl_{l*}^{t}$ | Lower bound of acceptable average grade in period $t_{l*}$ for level $l*$ |
| $i$ | Discount rate |
| $L$ | Maximum number of levels in LOL method |
| $L*$ | Maximum number of levels in SOT method |
| $Lenz$ | The dimension of stope in the Z direction |
| $M_l$ | Life of mine |
| $Mcl$ | Maximum number of concurrent active levels |
| $N_{l*}^{s*}$ | Number of selected stopes $s*$ in level $l*$ |
| $Nl$ | Total number of levels |
| $NPV$ | Net present value |
| $Ns$ | Total number of positive stopes |
| $Op$ | Number of operation days |

| | |
|---|---|
| $p$ | Metal price |
| $Pl$ | Number of possible levels |
| $Pr$ | Production rate |
| $r$ | Recovery |
| $\mathrm{R_t}$ | Metal processing recovery in period $t$ in SOT method |
| $R_{l*}^{t}$ | Metal processing recovery in period $t_{l*}$ for level $l*$ in SOL method |
| $S$ | Maximum number of positive stopes |
| $S*$ | Maximum number of stopes selected by LOL method |
| $S_{l*}^{*}$ | Maximum number selected stope $s*$ in selected level $l*$ |
| $t$ | Period of mining |
| $T$ | Maximum number of scheduling periods in SOT method |
| $T_{l*}$ | Maximum number of scheduling periods in level $l*$ |
| $Ton$ | Expected tonnage of mining |
| $ton_{bl}$ | Tonnage of the block |
| $Ton_{s*}$ | Tonnage of stope $s*$ |
| $Ton_{l*}^{s*}$ | Tonnage of stope $s*$ in each level $l*$ |
| $Tz$ | Number of blocks in direction Z of the block model |

# CHAPTER 1: INTRODUCTION

*This chapter is an overall overview of the research. It includes the background of the study; the problem statement; the objectives of the research, scope and limitations; the research methodology; and the contributions of the investigation.*

## 1.1. Background

The economics of today's mining industry is such that the major mining companies are increasing the use of massive underground mining methods. It is a step change for the industry, move from the traditional open-pit mining to the underground mining. Because of incentive to the underground mining, the mine planning in underground mining and its optimization have been considered more seriously in recent years. Compare to the open-pit mine, limited techniques and algorithms are available for underground mining because of the complexity and less flexibility due to geotechnical and operational constraints in underground mining. In fact, to model the problems in underground mining, more constraints and variables are required. The number of variables at mix integer programming (MIP) may exceed hundred thousand in the planning of underground mining (Little et al., 2011). Even, this complexity can be higher in sublevel stoping mining method because of some condition such as the alignment of the extraction level, and non-concurrent production (Copland et al., 2016).

Based on Topal (2008) the optimization of three aspects of the underground mine planning is the center of the attention. These aspects are stopes layout, production scheduling and infrastructure. The stopes layout optimization determines the dimension and the locations of the stopes. Production scheduling is defined as the sequential order of the preparation, extraction and backfilling of the stopes. The goal of stopes layout and production scheduling optimization is profit maximization. In fact, stopes layout and production scheduling optimization are tools to maximize the profitability of mining over mine life while the operational and geotechnical constraints are met.

The existing algorithms for underground stope optimization are divided into two sets level-based and field-based. Level-based algorithms of stope optimization implement the optimization on the different levels or panels of the block model; however, field-based stope optimization algorithms are applied on the block model before dividing into levels or panels (Sotoudeh et al., 2017).

Since different mining methods obtain the different geotechnical constraints, it is not reasonable to define a general purpose optimization algorithm suited for all underground mining methods (Bai et al., 2012). As a result, in this study in contrast with the majority of previous researches, the optimization of stopes layout and production scheduling specifically in sublevel stoping method, which is suitable for the wide vein-type steeply dipping deposits, is investigated.

The focus of this study is on two aspects of three aspects of optimization in underground mine planning presented by Topal (2008). Create the optimization models to find the optimum stopes layout and optimum production scheduling in sublevel stoping are the purposes of this study. Besides, the impact of applying the level-based and field-based algorithms is evaluated.

## 1.2. Statement of the Problem

The desire for the underground mining and subsequently the demand for the reliable underground planning and optimization approaches have been grown. However, not adequate number of practical and accurate optimization approaches especially for stopes layout and production scheduling have been caught. In the current mining industry, the designing of stopes layout is still limited to the manual techniques with narrow computer aides that causes to not having the optimal solution (Little, 2012). As Nehring (2011) mentioned, the production scheduling is also largely dependent on the manual techniques with a limited computer used. Similar to the designing the stopes layout the manual techniques cannot guarantee the optimality of the solution, as they are not able to satisfy the complexities of all objectives. As a result, determining the practical models to design the stopes layout and production scheduling is necessary.

Several algorithms have been presented to optimize the stopes layout and stopes production scheduling for underground mining in the last four decades although those fail to provide a comprehensive solution. Most of the algorithms are heuristic so they cannot guarantee the optimality. Besides, since many simplifications were used in order to provide the algorithms, they are not appropriate to find the solution for the real mining case (Nikbin et al., 2018).

The goal of designing optimum stopes layout is to choose the best combination of blocks of the block model to be extracted. By determining the best dimension and the location of the stopes that contain numbers of blocks, the achieved profit will be maximized. Finding the stopes production scheduling is the response to questions that which stope and when that stope can be extracted. The primary goal of production scheduling is determining a sequential order to mine the stopes to provide the maximum net present value (Manchuk, 2007). Stopes layout and production scheduling are subjected to a variety of operational and geotechnical constraints which makes the planning optimization even more difficult. The defined constraints enforce the selection of the blocks to be mined, the number of stopes includes their size and their location, the production

target, mining precedence, number of mined stopes in each period and continuous mining. As a result, considering the practical constraints plays an essential role.

The selection of the stope and defining the sequential order of mining the stopes can be done in each level separately or can be applied to the whole block models as one set. The difference between these two methods has not been clarified in the previous researches. However, the goal of both methods is selection the stopes to have the highest economic value and to define the timing of the mining the stopes to achieve the highest net present value during the life of the mine. The result will be more invaluable if the optimum solution can be determined by applying the mathematical programming.

Generally, the following research questions drive this dissertation.

> *Can stopes layout optimization framework of sublevel stoping be established that will result in the maximum economic value for the mining operation while meeting the geotechnical constraints, such as stopes overlap?*
>
> *May an optimum production scheduling specifically for sublevel stoping be defined to achieve the maximum NPV during life of the mine while honoring the operational and geotechnical constraints include mining capacity, grade blending, only one-time mining, stopes adjacency, connection between mining the stopes and activation of levels, concurrent active levels and the delay between activation of the levels?*
>
> *What are the differences between the result of stopes layout and production scheduling optimization based on the total block model and based on the levels? What are the advantages and disadvantages of each method?*

In this research, finding the optimum stopes layout and optimum production scheduling based on two methods of field-based and level-based are studied, and results are compared.

### 1.3. Summary of Literature Review

Sublevel stoping which also is referred to as long-hole stoping or blast-hole stoping is a vertical large-scale underground method. This method is a proper underground mining technique for wide vein-type deposits with stable host rock and competent steeply dipping ore-bodies (Haycocks et al., 1992; Lawrence, 1998). After finishing the development of declines, shafts, raises, orepasses

and production levels, a raise slot or a winze is operated into one corner of the stope from one sublevel to the next sublevel, and drawpoints and funnels are provided. After completing the infrastructures and constructing the slot, the drilling, blasting and extraction in the stope can be started. Extracted ore in drawpoints is transported to the crusher or the surface. After extracting the ore within the stope, stope is backfilled by a mixture of mill tailings and cement (Hartman, 1992; Haycocks et al., 1992; Nehring, 2011).

The goal of designing stopes in sublevel stoping is achieving the highest profit by defining the best location, size and number of stopes within an ore-body, while the geotechnical stability concerns are met. The stope size depends on the size and shape of the ore-body (Nehring, 2011). The main consideration with planning and scheduling of sublevel stoping method is geotechnical nature of the ore-body such as faults and principal stress directions. Beside the geotechnical nature of the ore-body, other parameters related to the ore-body including shape, continuity and grade distribution of the ore-body are important parameters in designing and scheduling of sublevel stoping method. Also, some other factors such as the filling types should be considered (Mann, 1998).

Generally, the all presented 2D or 3D algorithms to define the optimal stopes layout classify to two groups of mathematical and heuristic. Mathematical proof supports mathematical algorithms; however, the heuristic algorithms are based on constraints and limitations to find an approximate solution (Sandanayake, 2014). In order to see the optimum stopes layout, geotechnical and operational and economic considerations such as characteristics of the ore-body, accessing to stopes, mining equipment size, pillar size must be considered (Bai et al., 2012). The existing algorithms for underground stope optimization are divided into two sets of level-based and field-based (Sotoudeh et al., 2017).

Majority of the previous works focus on the stopes layout optimization or production scheduling. Few studies consider both, in some of those studies the stopes layout and production scheduling are applied simultaneously, and in others, stopes layout is defined first, and then the production scheduling is executed (Little, 2012).

The significant limitations of the current stopes layout and production scheduling optimization in sublevel stoping reviewed in Chapter 2 are:

- Not any algorithms consider the level-based and field-based optimization with the same data set to be able to compare the results.
- Most of the algorithms are applicable for all mining methods although different mining methods obtain the different geotechnical constraints.
- Only a few numbers of researches are presented the mathematical algorithm with the optimum solution.
- Not all the defined constraints are practical for the case of sublevel stoping method. Also, none of the studies covers all the required constraints for the real situation, for instance, concurrent active levels, and the delay between activation of the levels constraints are ignored in those algorithms.
- Not all the algorithms are able to solve the large-scale problems.

### 1.4. Objective of the Study

One of the objectives of this study is presenting a mathematical model to optimize stopes layout in sublevel stoping method in which the objective function is to maximize economic value. Another objective is generating a mathematical formulation to optimize a production scheduling in sublevel stoping method in which the objective function is to maximize net present value (NPV). In both cases, the related technical and operational constraints are respected. These objectives contain two elements: (i) generating a binary integer mathematical programming model, (ii) verification of the model by real data set.

The proposed methodologies generate an optimal solution with meeting constraints such as stopes overlap for stopes layout optimization and only one-time mining, stopes adjacency, the connection between mining the stopes and activation of levels, concurrent active levels, and the delay between activation of the levels constraints for production scheduling optimization. Stopes layout and production scheduling optimization are applied based on the total block model and based on the levels to see the impact of leveling. Figure 1.1 and Figure 1.2 indicate the better understanding of stopes layout optimization. Figure 1.3 and Figure 1.4 show the purpose of production scheduling

optimization in this study. Figure 1.1 demonstrates a block model and the result of the optimization to find the stopes layout based on the total blocks of the block model.

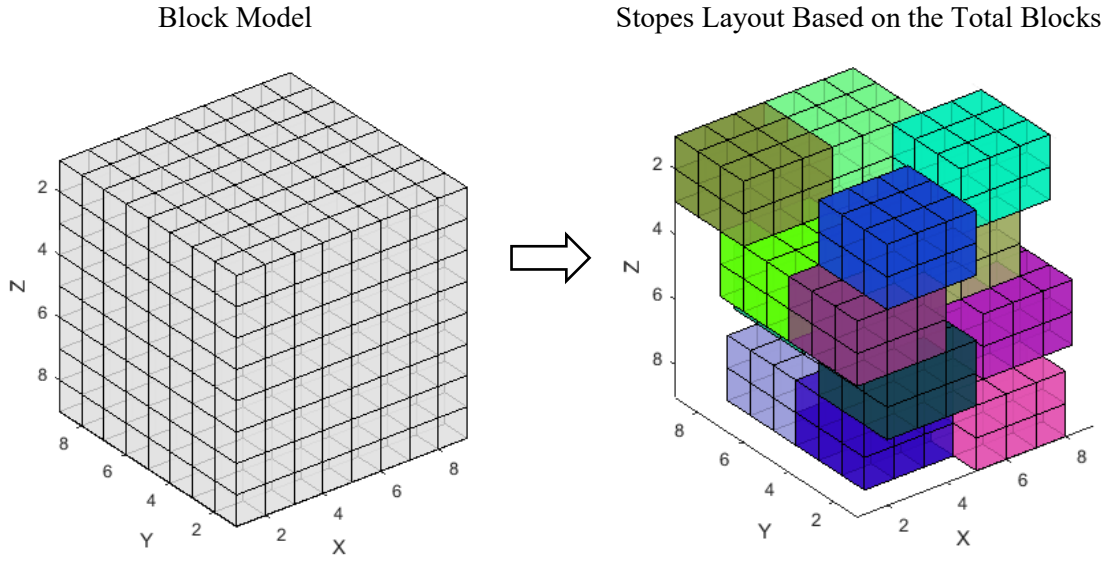



Figure 1.1. The block model and stopes layout based on the total blocks

Figure 1.2 illustrates the block model and the achieved stopes layout based on the applying the optimization in each level independently. The different colors indicate the selected stopes at the different levels.

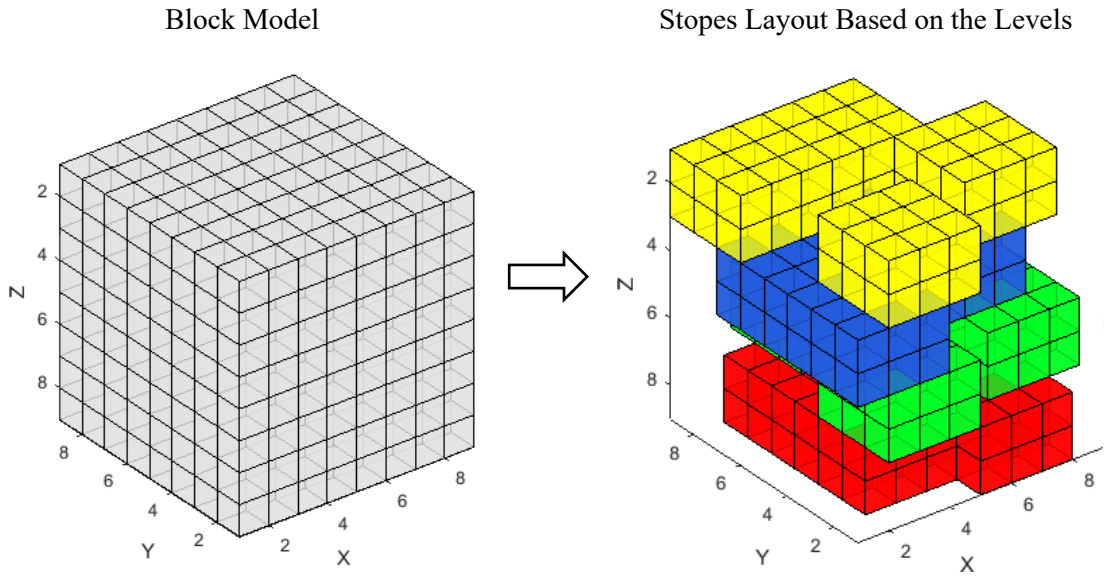



Figure 1.2. The block model and stopes layout based on the levels

Figure 1.3 demonstrates the sequential order of mining the stopes while all stopes are considered. Each color indicates a period, and stopes with the same color are mined at the same period. Figure 1.4 shows the result of production scheduling optimization based on the levels. In fact, optimization is applied to the stopes that are at the same level. It means after mining the stopes at a level, mining in the next level will be started.

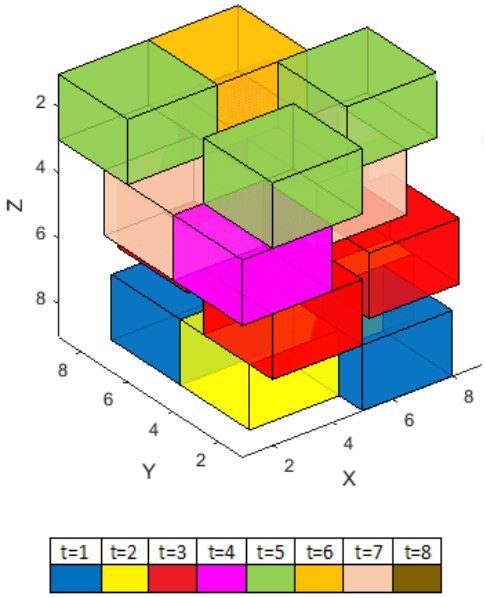



Figure 1.3. Production scheduling based on the total stopes

Generally, the objectives of this study are:

- Maximizing the economic value due to mining the selected stopes with taking into account the effects of geotechnical constraints to find the optimum stopes layout.
- Maximizing the net present value in order to find the optimum stopes production scheduling.
- Finding the best combination of stopes and best sequential order of mining the stopes to start the mining operation based on the practicality of the solution in the real mining situation.
- Developing computer codes to implement those mathematical formulations.

- Comparing the output of the optimizations based on the total block model and based on the levels to see the impact of leveling in stopes layout and production scheduling optimization.
- Assessing the results of all algorithms to ensure the feasibility and optimality of the solution by using a real data set.

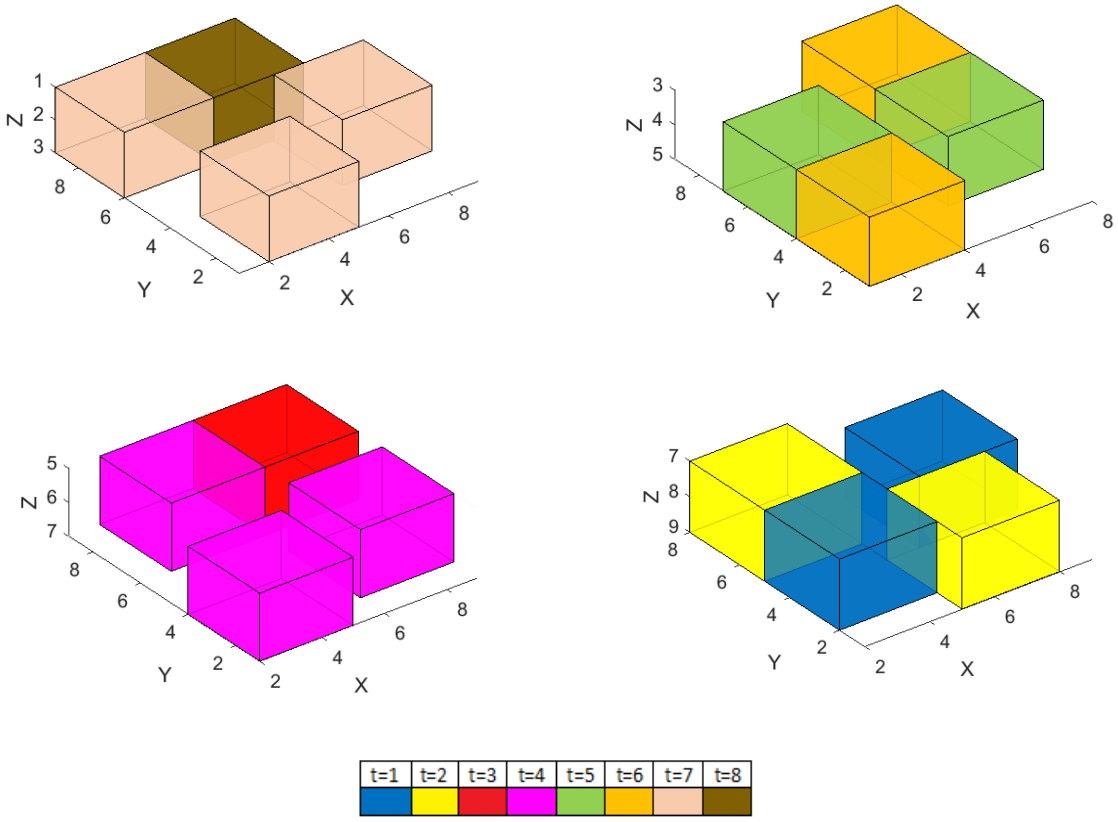



Figure 1.4. Production scheduling based on the levels

## 1.5. Scope and Limitations of the study

This research addresses the development, implementation and verification of a BIP model to create an optimal stopes layout and production schedule for sublevel mining with honoring of operational and geotechnical constraints. The final model's objective is to maximize EV in stopes layout and NPV in production scheduling optimization. As it mentioned, in this research, finding the optimum stopes layout and optimum production scheduling based on two methods of field-based and level-based are studied. Each method has its defined assumptions.

Assumptions of stopes layout optimization based on the total blocks are:

- The numerical data such as tonnage, grade, coordinates, and economic value are applied to identify ore-body attributes in each block.
- In order to create stope, no partial block is considered. In other words, a stope consists of the number of complete blocks.
- The present value of stopes is considered to find the optimum solution. In fact, the factor of time is not considered.

Besides the assumption of stopes layout optimization based on the total blocks method, the following assumptions are applied for stopes layout optimization based on the levels:

- In order to create a level, no partial stope is considered. In fact, levels must create only at top or bottom of the stopes, not middle.
- A level covers of all possible stopes with the same base elevation, not some of those stopes.

Assumptions of production scheduling optimization based on the total stopes are:

- No partial stope is considered in production scheduling. In other words, a stope must thoroughly be mined or not be mined in a period.
- When the model selects a stope in a period, it includes the preparation, extraction and backfilling of that stope during that period.
- The market fluctuations during the life of the mine are not considered.
- There is no material mixing between stopes and within a stope during mine operation.

Besides the assumption of production scheduling optimization based on the total stopes method, the following assumptions are placed for production scheduling optimization based on the levels:

- Only based on the proportion of material tonnage in each level out of total material tonnage, the proportion of mine life is assigned to each level and other factors are not considered.

- Production scheduling is done at each level independently, and the connection between levels is not considered.
- Mining capacity and required average grade in each period in SOL method are assumed same as SOT method other factors are not considered.

There are other common limitations in all presented methods in this research. Same as all previous researches, these algorithms are based on static orebody models, as a result, reach the optimality is not guaranteed (Nhleko et al., 2018). In all presented algorithms in this study, the pillars between stopes are not defined; however, adding those to the model is not a complicated procedure. In order to apply any changes of input parameters such as desirable stope size, the problem should be resolved.

### 1.6. Research Methodology

The primary motivation of this research is the improvement of stopes layout and production scheduling optimization in sublevel stoping mining method by finding the optimum solution in the presence of practical constraints. In addition, evaluation of the impact of leveling in stopes layout and production scheduling is another important purpose of this research. This research contains two main optimization models; (i) stopes layout optimization and (ii) production scheduling optimization. Each of those is divided into two categories: optimization based on the total blocks of the block model and optimization based on the different levels.

The steps of the stopes layout optimization based on total blocks are as followed:

1. Generate the block economic model based on the economic parameters.
2. Create the initial stopes that contain all possibilities of stopes, assigned the economic value to each of those according to the economic value of blocks, and finally select the positive value possible stopes.
3. Assess the overlaps between all the positive possible stopes to determine the positive stopes.
4. Discover the best combination of positive stopes with no overlap and the highest economic value. This combination is referred to as the optimum stopes layout.

The steps of the stopes layout optimization based on the levels are as followed:

1. Generate the block economic model based on the economic parameters.
2. Define possible levels along the Z direction of the block model that can be as the elevation base for creating the possible stopes.
3. Create the initial possible stopes, assigned the economic value to each of those, and select the positive value possible stopes in each level independently.
4. Assess the overlaps between all the positive possible stopes to determine the positive stopes in each level independently.
5. Find the best positive stopes combination without overlap and the highest economic value in each level independently.
6. Discover the optimum set of levels by comparing the economic value of levels that is the summation of the economic value of selected positive stopes in that level. This set of selected positive stopes in the chosen levels makes the optimum stopes layout.

The steps of the production scheduling optimization based on the total stopes are as followed:

1. Define mine life according to the summation of tonnage of the selected stopes which are selected by optimum stopes layout optimization based on the levels method and production capacity of the mining operation.
2. Determine the economic value of each selected stope in different periods.
3. Define adjacent stopes according to their blocks distances to limit mining those stopes concurrently during any periods.
4. Define the relationship between levels that includes the reasonable delay between activation of subsequent levels and the maximum number of active levels at the same time in order to have the practical production scheduling.

5. Determine other considerations include minimum and the maximum of the capacity of the mining operation, minimum and maximum of the required average grade and desirable direction of the mining between levels.
6. Discover the best timing of mining the stopes scheduling with the highest NPV by considering all the limitations.

The steps of the production scheduling optimization based on the levels are as followed:

1. Define the first level based on the desired direction of mining.
2. Define the total number of the period for each level based on the summation of the tonnage of the selected stopes, which are selected by optimum stopes layout optimization based on the levels method, in that level and mining operation capacity assigned for each level.
3. Determine the economic value of each stope in different periods for each level.
4. Define adjacent stopes in each level independently according to their blocks distances to limit mining the adjacent stopes in a level concurrently.
5. Determine considerations include minimum and the maximum of the capacity of the mining operation in each period, minimum and maximum of the required average grade in each period and the sequence of the activation of the levels.
6. Discover the best timing of mining the stopes with the highest NPV by considering all the limitations in each level independently.

After following all mentioned steps and defining the scheduling parameters, objective functions and constraints in MATLAB (MathWorks Inc, 2017) are created. Then, models are solved by using IBM ILOG CPLEX Optimization Studio (IBM, 2017). Finally, a gap tolerance (EPGAP) is utilized as an optimization termination criterion.

## 1.7. Scientific Contributions and Industrial Significance of the Research

The main scientific contribution of this research is to develop, assess and implement mathematical programming models in the context of sublevel mining to find the optimum stopes layout and production scheduling. The summary of the contributions are:

- Developing a mathematical model to generate the optimal stopes layout in sublevel stoping based on the total block model, as well as based on the levels which maximize the economic value while satisfying the constraints.
- Developing a mathematical model to generate the optimal production schedule in sublevel stoping based on the total blocks of the model, as well as based on the levels which maximize the NPV by considering the variety of constraints.
- Allowing the mining industry to compare the result of applying the stopes layout and production scheduling optimization based on the total blocks of the block model or based on the defined levels in order to decide about using the better method for different situations.
- Applying the introduced models on the real size industrial applications as it has been examined on a real sublevel stoping mining case.
- Paying attention to the practicality by defining the set of practical constraints including stopes overlap, mining capacity, grade blending, only one-time mining, stopes adjacency, the connection between mining the stopes and activation of levels, concurrent active levels, and the delay between activation of the levels constraints.
- Considering some constraints which were ignored in the previous works such as the number of concurrent active levels and the delay between activation of the levels.

## 1.8. Organization of Thesis

Chapter 1 of this thesis is an introduction to the study. It concludes an overall explanation of the background of the research followed by the statement of the problem, objectives, scopes and limitations of the study, the proposed methodology and the contribution of the study.

Chapter 2 contains a literature review of the sublevel stoping mining method. Provide a review of previous researches and generated algorithms to find the optimum stopes layout in underground mining related to sublevel stoping, and their applications, methodologies, capabilities, restrictions and their similarities and contrasts are discussed. In addition, the investigations about production scheduling optimization related to sublevel stoping method and their methodologies, objective functions and constraints of the models are analyzed.

Chapter 3 describes the steps of two methods of stopes layout optimization. The first method called LOT method considers total blocks of the block model altogether to create the possible stopes and applies the optimization based on those stopes and second method called LOL, the block model is separated to the levels, and possible stopes are created for each level separately. Besides, two methods of SOT and SOL of stope production scheduling optimization are explained. SOT considers total selected stopes as one set and employs the optimization process on those stopes and SOL method applies the optimization process for each selected level independently. Developing the BIP model for stopes layout and production scheduling optimization are the next sections. Finally, the implementation of the BIP formulation and the creation of the matrices for objective function and constraints are highlighted.

Chapter 4 provides the implementation of the proposed model and steps in Chapter 3 on a case study of sublevel stoping method. It explains how various components of the BIP model can be set up in MATLAB (MathWorks Inc, 2017) and how IBM ILOG CPLEX Optimization Studio (IBM, 2017) is utilized to solve a large-scale BIP problem. Then, the comparisons between the achieved solutions by LOT and LOL methods are studied to assess the methods. Besides, the result of SOT and SOL methods are compared.

Chapter 5 includes the summary, the contribution of the research and suggestions for future work.

# CHAPTER 2: LITERATURE REVIEW

*Chapter 2 provides a review of stopes layout and production scheduling algorithms related to sublevel stoping mining method (SLS) in the mining industry. It contains a brief description of the sublevel stoping mining method. Previous researches and generated algorithms to find the optimum stopes layout in underground mining specifically sublevel stoping are reviewed, and their applications, methodologies, capabilities, restrictions and their similarities and contrasts are discussed. In addition, the investigations about production scheduling optimization related to sublevel stoping method and their methodologies, objective functions and constraints of the models are analyzed. The chapter contains the remarks and summary.*

## 2.1. Sublevel Stoping

Sublevel stoping which also is referred to as long-hole stoping or blast-hole stoping is a vertical large-scale underground method. This method is a proper underground mining technique for wide vein-type deposits with stable host rock and competent steeply dipping ore-bodies where the angle of the footwall is higher than the angle of repose of the blasted ore. In this method, the ore is blasted from different levels and discharges to the lower level of elevation (Haycocks et al., 1992; Lawrence, 1998).

The advantages of the sublevel stoping method are high production rates per man per shift, high capability of automation potential, high personnel safety standards, high ore recovery and low operation costs. Also in the case of the competent ore-body and stable host rock, minimum stope support is needed. Also, different types of the sublevel stoping method exist to use for the different ore-body characteristics (Howard et al., 2002). However, due to the high amount of required development for generating the production levels, other accesses and the infrastructures, the required cost and times before stope production is usually high. Also in the case of caving in a stope due to the improper designing of stope support, a wide range of damaging to the surrounding stopes and infrastructure can occur (Lawrence, 1998).

Figure 2.1 shows sublevel stoping mining method. This figure indicates the accesses and mining functions in the different stopes. The access to the ore-body is through a decline or a shaft or combination of both. Considering technical factors include depth, shape and size of the deposit, the geology of the ore-body and host rock and desired production rate and economic factors at the same time effects on the selection of accessing tool to the ore-body. In overall, a decline is used for the ore-bodies close to the surface, and a shaft is utilized for deeper ore-bodies. Both decline and shaft can be used for haulage purposes. The dimension of decline relays on equipment size, traffic flow and geotechnical conditions. Usually, the width and height of the decline are between four to six meters. A shaft usually is utilized for transferring supplies, personnel, mined ore and ventilation. In addition, the dimensions of a shaft are defined according to the main purpose of its usage and the geotechnical conditions. For a haulage shaft, the range of its diameter is between six to seven meters (Hartman, 1992; Mann, 1998).

Due to decreasing the chance of geotechnical failures, all main accesses and infrastructures should be established in the footwall or the ore-body (Hartman, 1992; Haycocks et al., 1992).

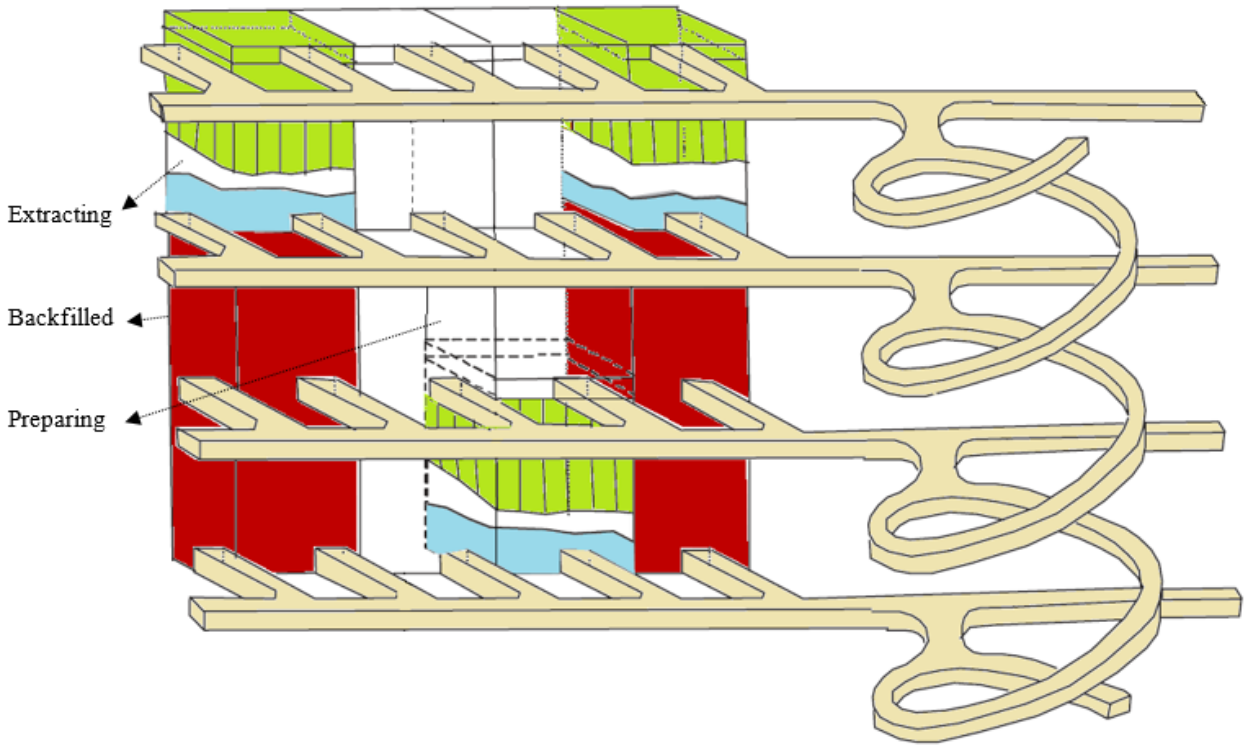



Figure 2.1. Sublevel stoping and mining functions (after Villaecusa, 2014)

After providing the primary accesses, raises, which are steeply inclined to connect different parts of the mine, and orepasses, which are used for transporting blasted ore and ventilation purposes, are built. The inclination of a raise varies between 55º and 90º and the cross-section of a raise varies between four and six square meters. Then after finishing the development of all accesses to the stoping areas, production levels are constructed. The vertical location of production levels is based on the sublevels placement. Usually, an ore-body is divided into stopes with the height of 45 to 120 meters. Afterwards, a raise slot or a winze is operated into one corner of the stope from one sublevel to next sublevel, and drawpoints and funnels are provided. In order to decrease the preparation cost and time, the number of sublevels in a stope is minimized, and the height between sublevels is maximized. Sublevels are driven into the stope every 10 to 55 meters (Hartman, 1992; Haycocks et al., 1992; Atlas Copco, 2000; Nehring, 2011).

Constructing a slot to provide room for drilling and blasting within the stope is the next step. Finally, based on the drilling pattern, the drill holes are generated on the sublevel to extract the ore in the stope (Hartman, 1992). Extracted ore in drawpoints is transported by Load Haul Dump (LHD). Then, trucks, orepass, conveyor, bin or skip can be used to transfer the blasted ore to the

crusher or the surface (Nehring, 2011). After extracting the ore within the stope, stope is backfilled by a mixture of mill tailings and cement (Hartman, 1992).

Pillars are used in the stopes to implement different functions. In order to protect the level above the stope crown pillars, to separate the adjacent stopes rib pillars and to collect the blasted ore in drowpoints sill pillars are applied (Little, 2012).

Sublevel stoping designing includes developing of the accesses, drawpoints and stopes and considering the ground support, dilution, backfilling, hauling, ventilation (Nehring, 2011). The main point of designing stopes is achieving the highest profit by defining the best location, size and number of stopes within an ore-body, while the geotechnical stability concerns are met. The stope size depends on the size and shape of the ore-body. Generally, to limit the dilution in the outer regions of the ore-body, the shape of outer stopes follows the shape of the ore-body (Hartman, 1992). In the case of thicker ore-bodies with multiple stopes along and across the strike, inner stopes have regular shape. Also, the profitability of a stope may effect on the stope shape (Copland et al., 2016). Usually, square or rectangular are selected as stope shape at the inner part of the ore-body (Nehring, 2011).

The main consideration with planning and scheduling of sublevel stoping method is geotechnical nature of the ore-body such as faults and principal stress directions. Rockmass behavior is not controllable; however, the sequence of extracting the stopes can play an essential role in managing the rockmass behavior (Villaecusa, 2003). Beside the geotechnical nature of the ore-body, other parameters related to the ore-body including shape, continuity and grade distribution of the ore-body are important parameters in designing and scheduling of sublevel stoping method. Also, some other factors such as the filling types should be considered (Mann, 1998; Nehring, 2011).

## 2.2. Stopes Layout Optimization

Optimization techniques for defining the stopes layout back to the more than forty years ago. Riddle (1977) presented the first algorithm, called “Dynamic Programming Algorithm” to find optimum stopes layout in block-caving mining method. This method solves the 3D problems by using 2D north-south sections and east-west sections. Although the presented algorithm can optimize the sections, it fails to find the optimum stope in three dimensions because it does not consider all necessary constraints simultaneously. Dynamic Programming Algorithm assumes a

2D section of blocks with rows and columns. At the first step, the profit achieved by mining the block for all blocks in the first level of drawpoint is calculated. Then, the first column is eliminated and the profit achieved by mining the blocks in the second level of drawpoint is calculated, and this process continues for other columns as well. Result in the last column is equal to the cumulative net value of blocks. Then the calculations for other rows are done, and at the final step, maximum profit is determined. This methodology is simple to use, however, it not suitable for 3D designing.

Deraisme et al. (1984) used the Downstream Geostatistical Approach to determine optimal stope. This model is a 2D sectional numerical model. Mathematical morphology helps this model to consider the stope geometry constraints. This approach is recommended when because of the underground mining constraints restrictions; the linear and nonlinear geostatistics are not able to estimate the mineable reserves. Generally, the Downstream Geostatistical Approach is based on a combination of conditional simulation with underground mining simulation to compare selectivity, productivity, and profitability in cut-and-fill and block caving methods. The approach steps include the constructing a numerical model of the deposit as the first step and then defining the outlines of the mineable ore.

Cheimanoff et al. (1989) described a heuristic approach with binary-tree division technique, called “Octree Division Approach”, to move from geological resources to mineable reserves based on the mining constraints and provides a 3D solution to find optimum stope. In fact, this model is based on the removing the non-desired mining blocks to define minimum stope size. This model covers two main constraints. First, the geometric constraints which are based on the ore-body geotechnical behaviour as well as mining equipment. Second, the economic constraints which are based on the cut-off grade and the mining costs such as accesses cost and services cost. The algorithm determines the reserves based on these two constraints. Then, the reserves are divided into small volumes. These small volumes will be eliminated from the block model if they do not meet the constraints. Since this algorithm does not control the amount of the waste in the final mine layout, it cannot guarantee to reach the optimum stopes layout.

Ovanic et al. (1995) developed one-dimensional “Branch and Bound Technique” to optimize outline of the stope based on the optimizing of starting and ending points of mining locations within each row of the blocks. To find the optimum starting and ending points, they used two

piecewise linear cumulative distribution functions for each row. In addition, they considered a mixed integer approach, called "Type-Two Special Ordered Sets", to optimize stope boundary. Two separate "Type-Two Special Ordered Sets" are defined as stope boundaries (starting and ending points) variable. The objective function in each row is determined as the difference between the cumulative values of blocks in starting and ending points of each row. In addition, the constraints are based on the geometric limitations, which impact on minimum and a maximum size of stopes. In contrast with previous algorithms, having regular or uniform the shapes blocks is not required in their algorithm. In other words, shape and size of the blocks do not effect on the optimization because the block cumulative value function is developed in this model. Using this model is beneficial in the case of existing the geological interpretations in the block model.

Alford (1996) described a heuristic model called “Floating Stope Algorithm”, which is similar to the "Moving Cone" method in open-pit optimization, to set up the optimal stope boundary. This algorithm applies to use in all underground mining method. Regarding the definition of "Floating Stope" term, this technique is based on moving a floating stope shape with minimum stope dimension, through blocks to locate the stope position. The procedure of floating the stope shape can be based on the best grade stope shapes or based on the possible stope positions. Furthermore, the main constraint is the geometry of the stope. Inner and outer envelopes are the model’s outputs, and the solution is located between these two envelopes which are defined by users. It means finding the solution relays on the users’ experience that can be led to the error. This algorithm is the based algorithm on the Datamine (Datamine International, 1981) software .

The presented algorithm by Alford (1996) improved by Cawrse (2001) which called multiple pass floating stope process (MPFSP). MPFSP works as the extension of the “Floating Stope Algorithm” to generate more envelopes by applying a multiple-optimization process. As a result, extra information about to selection of the best stope layouts can be provided for the mine-planning engineer. This algorithm exams the stoping feasibility and identifies the possible areas for the development of stopes. Nevertheless; the shortcomings of the “Floating Stope Algorithm” are not completely covered by this algorithm and cannot generate optimality of stope layouts (Cawrse, 2007). Improving the “Floating Stope Algorithm” was continued by the Alford Mining Systems (AMS) and Australian Minerals Industry Research Association (AMIRA). Their algorithm is able to optimize the location and shape of the stope in order to maximize the economic value within the stope boundaries. A set of wireframes that can be used for further mine design and scheduling

is the output of this algorithm. Maptek (Maptek Co, 1981) and Deswik (Deswik Mining Consultants Pty Ltd, 2007) software use this algorithm (Bootsma et al., 2014).

Ataee-pour (2000) and Ataee-pour (2005) presented a heuristic algorithm and called it "Maximum Value Neighborhood" (MVN). This algorithm works on the economic block model of an ore-body to provide a 3D analysis of optimization of the stope boundaries. He defined the neighborhood concept based on the number of mining blocks equivalent to minimum stope size. The MVN algorithm is applied to all underground mining method. MVN algorithm locates the best neighborhood of a block to find the best combination of blocks to create the maximum profit, while certain mining and geotechnical constraints are considered. FORTRAN programming language is applied to develop the algorithm. The stages of the algorithm start by generating the block economic value. Then, the sets of possible neighborhoods are determined for each block, and the feasibility of each neighborhood (If the neighborhood elements are located inside the block model or not) is evaluated. Afterwards, the economic value of each neighborhood is calculated, the maximum value neighborhood is determined and by adding this stope to the stope boundaries, the economic value is updated. This procedure is continued until all positive blocks are assessed. MVN algorithm failed to determine the optimal stopes layout. However, it guarantees the optimum value neighborhood for each block. The problem with this algorithm is how to combine these optimum neighborhoods value to create the optimum layout. Also, selecting the different starting locations for executing the algorithm leads to the different result. According to Nhleko et al. (2018), "Maximum Value Neighborhood" algorithm is based algorithm in the MinSight (Mintec Inc., 1970) software.

Topal et al. (2010) proposed a heuristic algorithm to find optimum stopes layout in the case of single as well as variable stope sizes in three-dimensions. Their proposed methodology can be used in all underground mining method. It consisted of three basic elements which are block converter, stope boundary optimizer, and stope visualizer. Block converter is created to convert a block model with multiple block sizes into a block model with only one size of blocks with new values. Stope boundary optimizer element uses a range of all the possible stope sizes, ore price, mining and processing costs, backfill as well as the fixed stope start-up costs as inputs and the optimum stope boundaries and layout for ore-body are the outputs. The stope boundary optimizer starts from the smallest available stope size on every possible location, and it continues to evaluate all the possible stopes and their profits. At first, the information of all positive possible stopes is

listed in a table. Then, the stope with the highest value is selected from this table and is labelled. All neighboured stopes with overlap with this stope are cut off from the table. This procedure continues until there are not any stopes in the table. The eliminating the neighboured stopes with overlap without any analysis develops a risk of removing the combinations with higher values. The stope visualizer element is a program to create the three-dimensional view of the final stopes layout. Their algorithm works based on two assumptions. Firstly, all stopes have a fixed start-up time, and the production and backfilling time have a linear relation with the stope volume. Secondly, the calculation of NPV is based on the mining of single stope at a given time.

Sens (2011) suggested the combinatorial optimization algorithm suitable for sublevel stope mining to optimize the stope boundaries, infrastructures and ore hauling systems. He used Ant Colony logic for designing infrastructure and Hill Climbing algorithm for ore hauling system optimization. He used three different criteria in stope boundary optimization. Firstly, the strategy based on highest profit per stope which shows the maximum possible overall profit. Secondly, the optimization based on the highest profit per square meter leads to the higher Net Present Value. Thirdly, the optimization based on the stope profit per mining time which demonstrates the mine design with optimised NPV. The overall procedure of the presented algorithm to optimize the stopes layout is similar to the presented procedure by Topal et al. (2010).

Bai et al. (2012) suggested a new 3D method using “Network Flow Algorithm” to design stopes layout. This model is based on a cylindrical coordinate. They believed that there was not a general-purpose optimization algorithm suited for all underground mining methods because of the differences between geotechnical constraints of different mining methods. As a result, they introduced an optimization algorithm that was suitable only for sublevel stoping method. Their optimization algorithm includes two main objective functions. The first one is the stope optimizer that consists of stope optimizing based on the specified raise location and height. The second one is finding the best raise location and height. In addition to the footwall and hanging wall slope, the stope width and height are played the constraints roles. Besides, the maximum distance of a block from the raise and required horizontal width of the block at that distance play the controlling role for the cylindrical system of coordinates. In order to consider the constraints, they have defined the arc in the graph in the cylindrical system and then after finding the overall optimal stope they have converted the solution to the Cartesian system. Since their algorithm is based on the cylindrical coordinate system with vertical raise, this algorithm is not acceptable in the case of

sub-vertical or sub-horizontal deposits that need inclined raise. Furthermore, this approach is based on the small ore-body with single raise parameters, and it is not useful for larger ore-bodies that need many contiguous stopes. Additionally, in their approach, they have used fixed development costs and operational costs although those are related to the raise location and height.

Sandanayake (2014) and Sandanayake et al. (2015a) offered the algorithm to maximizes the economic value regarding the physical and geotechnical constraints. They claim that the algorithm is flexible enough for varying underground mining situations. At the first step, after standardizing the irregular block model, this algorithm transferred the block model to the economic block model. After defining minimum and maximum stope sizes regarding a number of mining blocks, all possible stopes sets are created, and the positive value stopes are recognized. Then the positive stopes with overlap are defined and removed. Two blocks in each stope as the starting and ending blocks are defined to determine the overlap between two stope. Figure 2.2 shows the starting block and ending block of a stope. Stopes i and j have overlap if $Sx_i, Sy_i, Sz_i \le Sx_j, Sy_j, Sz_j \le Ex_i, Ey_i, Ez_i$ or $Sx_i, Sy_i, Sz_i \le Ex_j, Ey_j, Ez_j \le Ex_i, Ey_i, Ez_i$. Defining the economic value of each non-overlapping and positive stope set and determining the maximum economic value are the final steps of the algorithm. To validate the proposed algorithm, they made a comparison with MVN algorithm. Results indicated that the solution generated by this algorithm achieved the higher economic value than the MVN algorithm. However, the solution time was higher.

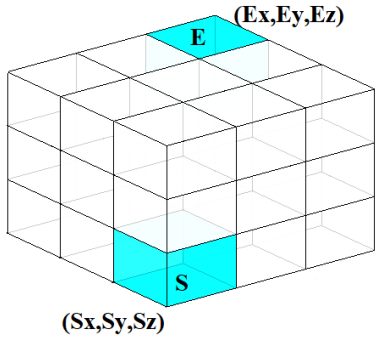



Figure 2.2. Starting and ending blocks of a stope in the presented model by Sandanayake (2014)

Sandanayake et al. (2015b) continued their work on finding optimum stopes layout by developing the previous algorithm. This algorithm considers the fixed and variable stope sizes. They run their algorithm with and without pillars between stopes cases. In addition, due to the infinite number of sets of stopes and the impossibility of finding the optimal solution, they defined an upper bound to limit the number of possible solutions. In fact, the algorithm runs over the number of iterations

until convergence of the solution value. This algorithm ia applicabale for all undergroung mining methods.

Villalba. M et al. (2017) worked on the minimization of inherent internal dilution and conventional profit maximization as the main approaches in the optimization of stopes layout. The economic, geotechnical, operational and ore-body quality and quantity constraints are subjected in their model. In their research, they have defined internal dilution, which is the waste or low-grade waste located within the ore, and external dilution, which is the waste or low-grade waste located on the border between ore and waste. The objective function of this algorithm comes with thirteen sets of constraints such as constraints to ensure that each block is mined only once, constraints for block precedence, constraints to consider minimum and maximum of height and width of stopes should be mined, constraints to define the grade greater than the cut-off grade.

Sotoudeh et al. (2017) presented a stopes layout optimizer called SLO3D. This algorithm is written in C# and SQL-Server programming languages. Their method is developed based on the (Sandanayake et al., 2015a)'s method by adding strategies to decrease the running time. After steps of creating the economic block model and generating the stopes possibilities, probabilistic strategies has been added to the stopes layout optimizer step. Their strategies are based on sorting of all possible sets of non-overlapping stopes according to their economic value to the lowest value and selecting a percentage of the sorted collection randomly and frequently or according to the descending order of the number of stopes in each set. This algorithm is mainly useful for large block model with the high number of stopes possibilities that need the high running time; however, some stopes sets might be ignored.

Nikbin et al. (2017) presented a Greedy Algorithm that is a polynomial time algorithm. The algorithm steps are creating the non-investigated stopes set, determining the most valuable probable stope within the set, adding all of the blocks located in each positive stope to the optimal stope boundaries, removing the defined stope from the non-investigated set, repeating the procedure as long as the positive value stope can be found. They compared the Greedy Algorithm with MVN Algorithm, and the results indicated the higher profit. They mentioned that this algorithm might fail to reach the optimum solution.

Villalba. M et al. (2018) proposed a three-stage stochastic optimization model to evaluate grade uncertainty in stopes layout optimization. To formulate the problem, they used the genetic

algorithm in three stages. The first stage is determining the stopes layout uncertainty. The second stage is generating the average feasible design of stopes layout. The third stage is upgrading the beginning population in all generations. This method discovers the near-optimal solutions whiten the reasonable running time. The presented method reaches the stopes layout that is not sensitive to the grade fluctuations of the ore.

Nikbin et al. (2018) introduced a Hybrid Algorithm that is a one-dimensional polynomial-time algorithm. This algorithm is a combination of 'Dynamic Programming Algorithm and Greedy algorithm. The algorithm includes three main steps. First, the cumulative value for each state (position) at each stage (dimension) is calculated. Then, the highest cumulative value state at the final stage as the optimal stope boundaries value is determined. Finally, based on the defined optimal stope boundary, the dimensions, positions, and the number of the stopes that create the optimal boundaries are recognized. The codes of the algorithm are in C# programming language. They compared their method with Floating Stope Algorithm, Maximum Value Neighborhood and Greedy Algorithm methods. Hybrid Algorithm could reach the higher profit within the reasonable running time; however, the running time is higher than those three methods. This algorithm guarantees the optimal solution for a selected row or column of a block model although may fail to reach the optimum solution.

## 2.3. Production Scheduling Optimization in Sublevel Stoping

The principal of the work to production scheduling presented by Manchuk (2007) is creating the sequence decisions based on information about the stopes and timing of operations and calculating the probabilities, which dictate the sequence based on this information. He categorized the constraints regarding two considerations the timing of events and a stope being mined. He mentioned three types of constraints based on the timing of events to schedule the production. Firstly, when a stope is selected to be mined, the preparation and extraction should be finished before starting the next stope. Secondly, only one stope at a time can be developed. Thirdly, breaks in stope extraction cannot take place. Also, he mentioned some constraints based on the stope being mined considerations. Firstly, while a stope is completely mined, it can be open for some maximum time before requiring backfill. Secondly, adjacent stopes must remain as the pillars until the stope is backfilled. Thirdly, development and production blasting cannot occur near the current stope on the same level and adjacent levels. He presented the simulated Annealing and a Logic

driven algorithm as the optimization techniques. Based on his research, the simulated Annealing technique achieves a better solution in a shorter time.

Nehring (2011) executed the research based on the short and medium term production and activity scheduling for sublevel stoping mining method. His research presented three mathematical optimization models using mixed integer programming to evaluate the relationship between medium and short term scheduling. These models are including the maximizing NPV in order to optimize the production scheduling throughout the medium term, minimizing deviation to a targeted mill feed grade to create the optimal short term production scheduling concerning the machine allocations, integration of last two models by the combination of two objective functions as one mathematical model. Results show that the third technique archived to slightly higher NPV. The presented constraints in the models are general mine development, internal development in the stopes, drilling in the stopes, extraction of the stopes, curing and backfilling in the stopes and grade fluctuations during the extraction of a stope.

Little (2012) presented two approaches to define the stopes layout and production scheduling in sublevel stoping method called integrated approach and isolated approach. In an integrated approach, the optimization of stopes layouts and production schedules are done simultaneously. In an isolated approach, the optimum stopes layout is defined as the first step, and then the production scheduling is applied. He experienced the higher NPV with the higher solution time in the integrated approach. His production scheduling model is to maximize the NPV of mining the stopes while constraints including overlaps between stopes, vertical planes of weakness backfilled stopes over multiple extraction levels, adjacent stopes, only one adjacent stope per period, shared a common extraction level between the adjacent stopes, metal quantity, backfill supply and ore handling capacity are met. He applied three strategies to reduce the solution time includes; abridging the block data by aggregated the data, simplifying formulation of the model constraints and utilizing an integer variable.

Copland et al. (2016) proposed a model to maximize the profit from mining the stopes minus the cost of the level development while considering the stopes with common blocks, single fillmass exposure and non-concurrent production of adjacent stopes, production tonnage, blended metal grades, the offset extraction layers, vertical planes of weakness between stopes and sequential development as the constraints. The model is employed for sublevel stoping underground method.

The outputs of the model are stope boundaries and locations of stopes in combination with time of mining the stopes and the levels. Their primary focus is on using the binary integers programming to help the model in order to decrease the solution time and increasing the applicability compare to the previously presented production scheduling models. By combining the sets of data in a model and preparing the easier access to the data, they could improve the structure of the data, and by applying the summary variable for constraints, they could decrease the range of solution. These two methods are the keys to reduce the solution time in their model.

## 2.4. Summary and Conclusion

In chapter 2 of this dissertation, the relevant literature to the topic has been presented. The literature review showed that in all methods of stopes layout and production scheduling optimization the ore block model is adopted as an input. Although, in some of those having regular and uniform shapes blocks are required, in some cases having partial blocks can be considered as well. Before 2000, few algorithms presented to determine the optimal stopes layout. However, some of those did not introduce the 3D models and the mathematical solution.

Various objective functions are described in the mentioned methods. Maximizing the overall profit or maximizing the NPV are most common objective functions; however other factors such as minimizing the dilution or maximizing tonnage of ore have been mentioned in some cases. Besides, some geotechnical constraints are considered in all algorithms as the constraints to find the optimum slope layout or sequence of stope mining. However, not all methods cover all geotechnical constraints. Additionally, in a few previous works, the economic constraints such as mining cost and the operational constraints such as equipment size are considered as well. Seems dealing with more constraints makes the closer result to the optimum stopes layout.

Simplicity and generality are two characters of some algorithms. Simplicity may be in the concepts, assessments and analysis steps of the algorithm. Moreover, generality is the ability in being applicable to different mining methods. The application of some algorithm are available for different types of stope based underground mining methods; however, some can be applied to the specific method. However, Bai et al. (2012) believed that using one algorithm for all underground mining methods was not a proper decision because of differences between geotechnical constraints of different mining methods.

Almost, all algorithms have covered minimum sizes of stope; nonetheless, not all of the indicated algorithms have acknowledged the maximum limits of the stope dimensions that from ground control considerations point of view are important. Some of the mentioned works can calculate only single stope size, but others can evaluate the variable stope sizes as well. Some of the existing algorithms apply the optimization on a level or panels and some of those works on the total block model.

# CHAPTER 3: THEORETICAL FRAMEWORK

*Chapter 3 presents two methodologies to find stopes layout optimization and two methodologies to discover the production scheduling optimization in sublevel stopping mining. In order to find optimum stopes layout, stopes layout optimization based on the total blocks (LOT) method and the levels (LOL) method are defined. For the sake of finding the optimum production scheduling, production scheduling optimization based on the total stopes (SOT) method and the levels (SOL) method are established. At the end of this chapter, the BIP model's implementation of all methods is discussed. The numerical modeling of the BIP models are reviewed, the different parts of the models such as objective function, and constraints applied in MATLAB programming environment are illustrated.*

## 3.1. Introduction

This chapter describes all steps of two methods of stopes layout optimization. The first method called LOT method considers total blocks of the block model altogether to create the possible stopes and apply the optimization based on those stopes. However, in the second method called LOL, the block model is divided into the levels, and possible stopes are created for each level independently. Then, based on the value of selected stopes in each level, the optimum levels will be selected. The combination of those selected stopes in selected levels creates the optimum stopes layout.

The presented methods of production scheduling optimization are applied on the selected stopes and levels, which are the outputs of LOL method. The first method called SOT considers total selected stopes as a one set and employs the optimization process on those stopes. The second method called SOL method applies the optimization process for each selected level separately. The output of production scheduler is the timing of mining each stope.

This chapter focuses on the developing the BIP model for stopes layout and production scheduling optimization. The objective function of the BIP model for stopes layout optimization is maximizing EV while controlling over: stopes overlap, the number of selected stopes, levels overlap and number of selected levels as the constraints. In addition, the objective function of the BIP model for production scheduling optimization is maximizing NPV while handling the mining capacity, grade blending, only one-time mining, stopes adjacency, connection between mining the stopes and activation of levels, concurrent active levels, and the delay between activation of the levels.

## 3.2. Steps of the Proposed Methodologies

### 3.2.1. Layout Optimization Based on Total Blocks (LOT)

The overall process of the proposed algorithm to implement the layout optimization is generated from five main steps. Figure 3.1 indicates these steps. The process starts by using the economic parameters to create the economic block model. The next steps are generating stopes, calculating stopes value and finding the positive ones. Then, based on the stopes overlap and stopes value, the best combination with the highest EV is discovered. Finally, the optimum solution is visualized.

#### 3.2.1.1. Generate the Block Economic Model

At the first step, the economic block model is prepared. The blocks information is the first group

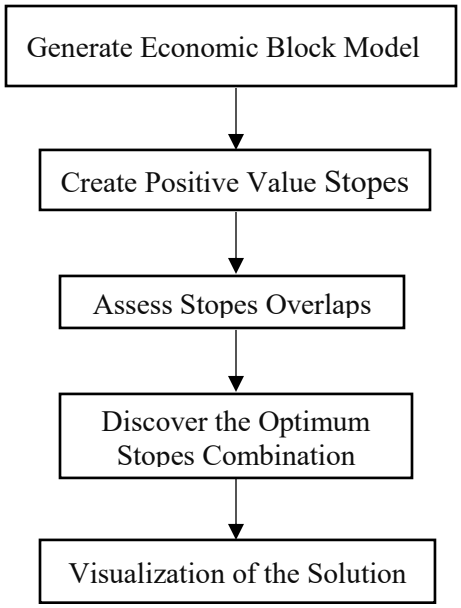



Figure 3.1. The overall process of the algorithm of layout optimization in LOT method

of input to the algorithm. The blocks information includes block ID, coordinates or indexes, grade, rock type, the tonnage of each block. The second group of the input is economic parameters contains the metal price, cost of selling, mining cost, processing costs, and recovery.

To calculate the EV for each block, the cut-off grade, which is the lowest sufficient grade of the material to send it to the processing plant, is required. If the grade of the block is equal or greater than the cut-off grade (ore block), the EV of the block is computed by equation (3.1). However, if the grade of the block is less than the cut-off grade (waste block), the equation (3.2) should be applied.

$$BEV = \left[ (p - c_s) \times g \times r - (c_m + c_p) \right] \times ton_{bl} \tag{3.1}$$

$$BEV = -(c_m \times ton_{bl}) \tag{3.2}$$

Where

$BEV$ : Block economic value

$p$ : Metal price

$c_s$ : Cost of selling

$c_m$ : Cost of mining

$c_p$ : Cost of processing

$g$ : Block average grade

$r$ : Recovery

$ton_{bl}$ : Tonnage of the block

### 3.2.1.2. Create Positive Value Stopes

At the first step, based on the geotechnical and mining constraints, the dimension of the stopes should be defined. The dimension of the stopes is based on the number of the blocks in three directions X, Y and Z. Then, the stope with this dimension is floated along axes to find all stope possibilities. Figure 3.2 indicates an example of a 2D economic block model with six blocks along the X-axis and four blocks along the Y-axis. Furthermore, the starting point for stope floating is shown in this figure. In this example, +2 is considered as the value of each block.

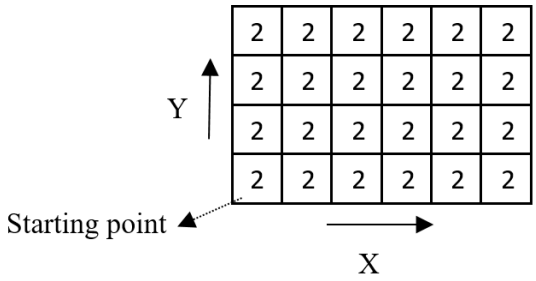



Figure 3.2. An economic block model (6×4)

Figure 3.3 illustrates how a 3×3 stope can float along the axes to create all stope possibilities. In this case, eight possibilities have been created. Then, the EV of each stope should be calculated which is the summation of all blocks value in each stope. For instance, in the current example, the value of nine blocks are summed to have the value of each stope, so the EV is equal to +18. Finally, stopes with the positive value are the output of this step. This means, all the stopes with the negative or zero value are eliminated.

### 3.2.1.3. Assess Stopes Overlaps

The primary consideration in this step is discovering the overlaps between positive possible stopes. In reality, not all stopes can be combined because of exciting overlaps between those stopes. To assess stopes overlap, an all zero elements matrix (overlap matrix) with the same dimensions of i

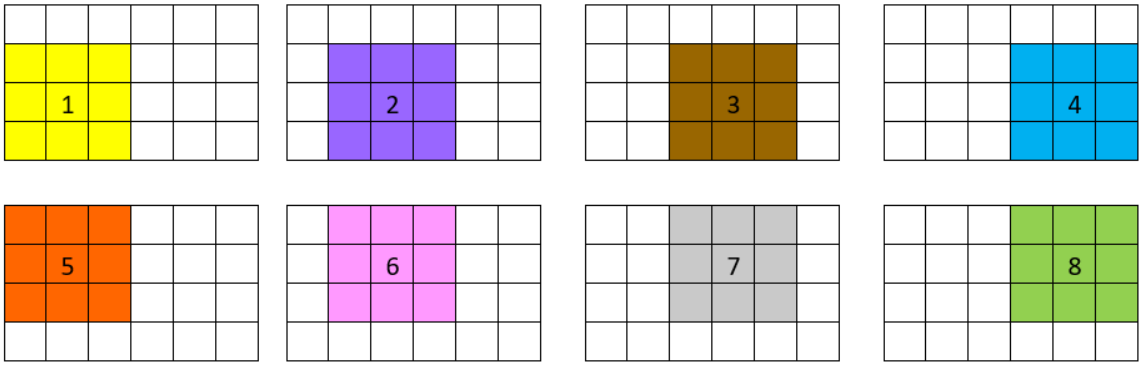



Figure 3.3. All possibilities of stopes

and j, and equal to the number of positive stopes is created, and by looping over all the elements, overlaps can be found. In fact, if two elements (two positive stopes) have one or more common blocks, those are accounted as the stopes with overlap. In the presented algorithm, if two stopes have overlap, element zero is changed to one in the overlap matrix. While, if two stopes do not have any overlaps, element zero is kept in overlap matrix. As a result, the overlap matrix, a matrix with elements zero and one, is created. Figure 3.4 shows the examples of stopes overlap for the current case. It is not possible to have stope number 1 and number 7 or number 6 and number 4 at the same time.

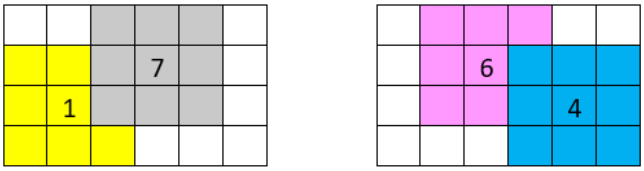



Figure 3.4. Examples of the overlaps between positive possible stopes

### 3.2.1.4. Discover the Best Stopes Combination

After defending the possible stopes with positive EV and the overlaps, the best combination of those stopes is determined. The positive stopes combination is not acceptable when there is the overlap between its stopes. For instance, for the mentioned case, the result of this step is as Figure 3.5 with four acceptable combinations of stopes.

EV of a combination is equal to the summation of EV of stopes in that combination. Due to running the optimization model, the best stopes combination with no overlap and the highest EV is determined. This combination of stopes generates the optimum stopes layout. For the mentioned

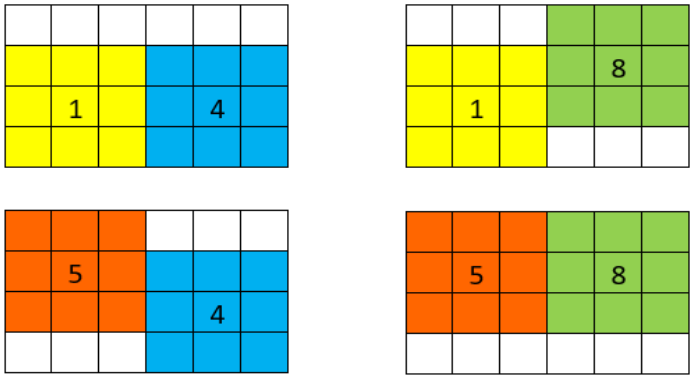



Figure 3.5. Possible stopes combinations

case, since the economic value of all blocks are the same, all of the possible stopes combinations in Figure 3.5 can be the optimum stopes layout.

#### 3.2.1.5. Visualization of the Solution

The output of this step is a plot of the best stopes combination with the highest EV.

Figure 3.6 indicates the better feeling of the optimum stopes layout. This example displays a 3D block model (8×8×8) and the optimum stopes layout solved by LOT method for stopes size (3×3×2).

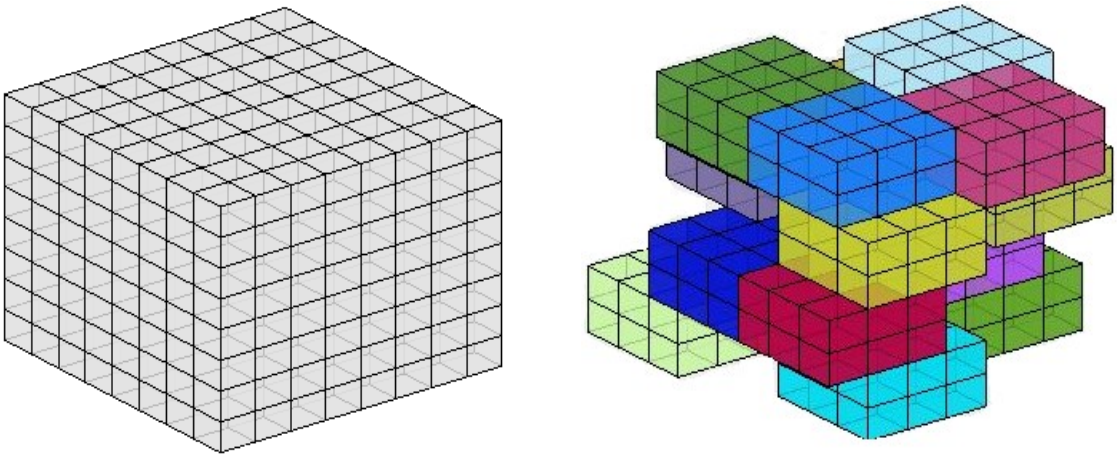


Figure 3.6. Block model and the optimum stopes layout based on LOT method

### 3.2.2. Layout Optimization Based on Levels (LOL)

The process starts by using the economic parameters to create the economic block model. The next step is defining all possible levels. Then, generating the possible stopes, calculating EV of stopes

and finding the stopes with positive EV in each level. According to the stopes overlaps and the EV of stopes, the best stopes combination, which is the combination with the highest EV in each level, is discovered. Based on selected stopes combination in each level and their values, the EV of each level and the best sets of levels with the highest EV is determined. Last step is visualizing the optimum solution. Figure 3.7 shows the seven main steps of the process.

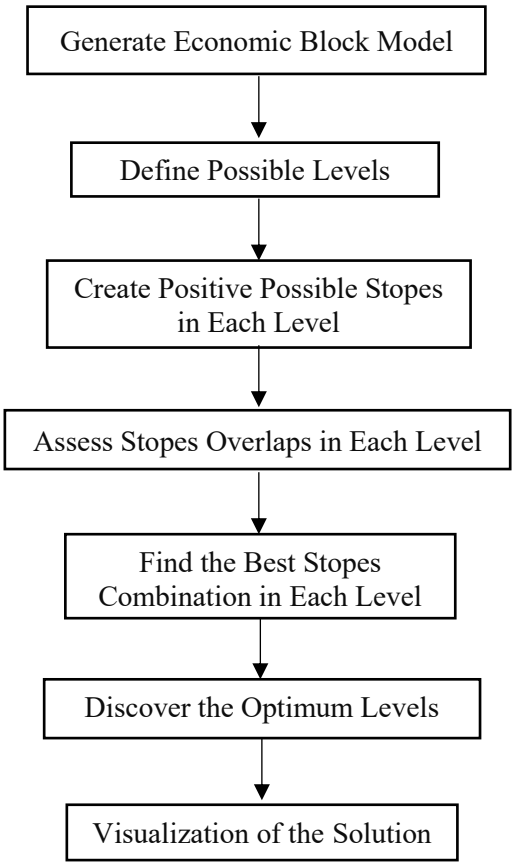



Figure 3.7. The overall process of the algorithm of layout optimization in LOL method

### 3.2.2.1. Generate the Block Economic Model

In the first section, the economic block model is prepared which was explained in detail in section 3.2.1.1.

### 3.2.2.2. Define Possible Levels

In this section, the primary block model is divided into the number of possible levels. Possible level defines as any elevation in Z direction that can be the elevation as the base for creating the stopes. Figure 3.8 indicates the definition of possible levels in the presented algorithm. As is shown in this figure, the desired dimensions of stope in direction X and Z are four (Dimension Y is not

displayed in the figure). According to the stope dimensions, three possible levels can be recognized, which are (Z=1), (Z=2) and (Z=3). In fact, these three elevations are the elevations that can be the base for creating the stopes.

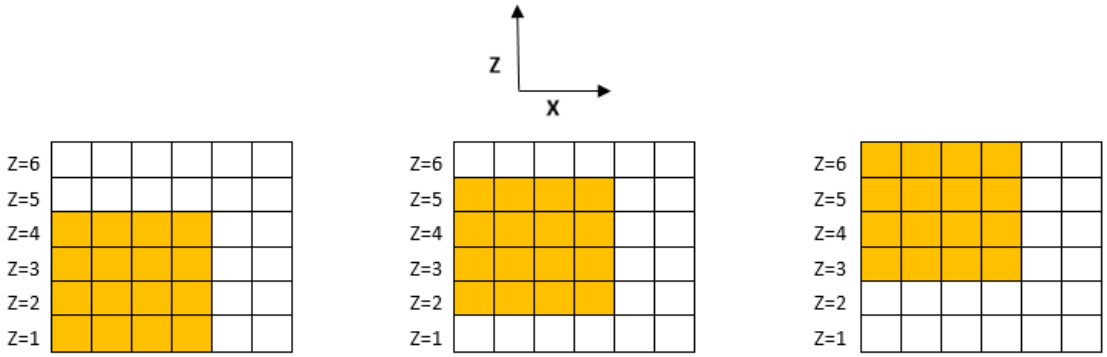



Figure 3.8. Possible levels based on the stope dimensions

A number of possible levels is calculated based on the equation (3.3).

$$Pl = Tz - Lenz + 1 \tag{3.3}$$

Where

$Pl$ : Number of possible levels

$Tz$ : Number of blocks in direction Z of the block model

$Lenz$ : The dimension of stope in the Z direction

For instance, in the case of a block model with 100 blocks in direction Z and ten blocks as $Lenz$, the number of possible levels is 91. As a result, to evaluate the block model, block model can be divided into the 91 levels.

### 3.2.2.3. Create Positive Value Stopes in Each Level

The process of creating positive value stopes in LOL method is same as creating positive possible stopes in LOT method explained in section 3.2.1.2. The only difference is in the LOL method the process of creating positive value stopes should be repeated for every single possible level. As a result, stopes with positive EV in all possible levels are generated.

### 3.2.2.4. Assess Stopes Overlaps in Each Level

This step is about finding all overlaps between possible stopes with positive EV. The same process

as section 3.2.1.3 should be applied in each possible level.

### 3.2.2.5. Find the Best Stopes Combination in Each Level

At this step, the presented algorithm creates the best stopes combination with the highest EV and without stopes overlap in each possible level. The overall process is the same as section 3.2.1.4. However, the algorithm needs to be run for all possible levels separately.

### 3.2.2.6. Discover the Optimum Set of Levels

At this step, the presented algorithm compares the EV of possible levels, which have been calculated by summation of EV of selected stopes in section 3.2.2.5, and finds the best set of levels with highest EV among all possible sets. It is obvious that the difference between elevations of levels in each set should be more than the dimension of stope in the Z direction. Figure 3.9 is an example to clarify how a levels set can be defined. This example demonstrates ten blocks in the Z dimension of block model (Z=1 to Z=10). In this example, dimension of stope in the Z direction is equal to 4. L1 and L2, L2 and L3 or L3 and L4 cannot be in a set because the difference between elevations of these levels is less than the dimension of stope in Z direction. In fact, these levels have overlaps in Z direction. However, the combination of L1 and L5, L2 and L6 or L3 and L7 makes appropriate levels sets. The levels set with the highest economic value is the optimum set of levels.

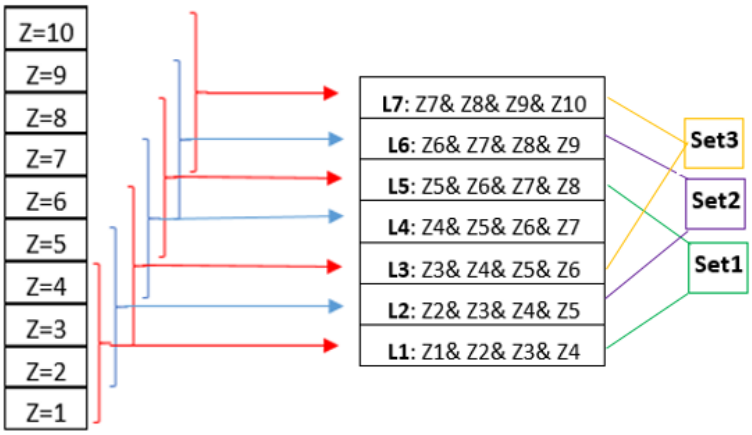



Figure 3.9. Example of the method of creating possible level set

Based on the optimum levels set, the optimum stopes layout generates. In fact, optimum stopes layout is the combination of selected stopes in selected levels.

#### 3.2.2.7. Visualization of the Solution

Finally, the optimum stopes layout as the output of the presented algorithm is displayed.

Figure 3.10 indicates the same example as done by LOT method (see Figure 3.6). Stopes with the same color are the result of stopes layout optimization at the same level. Based on equation (3.3), seven levels (*Pl*=7) can be defined in this case. As explained in Figure 3.9, first, third, fifth and seventh levels are selected which means these levels create the best set of possible levels with the highest EV.

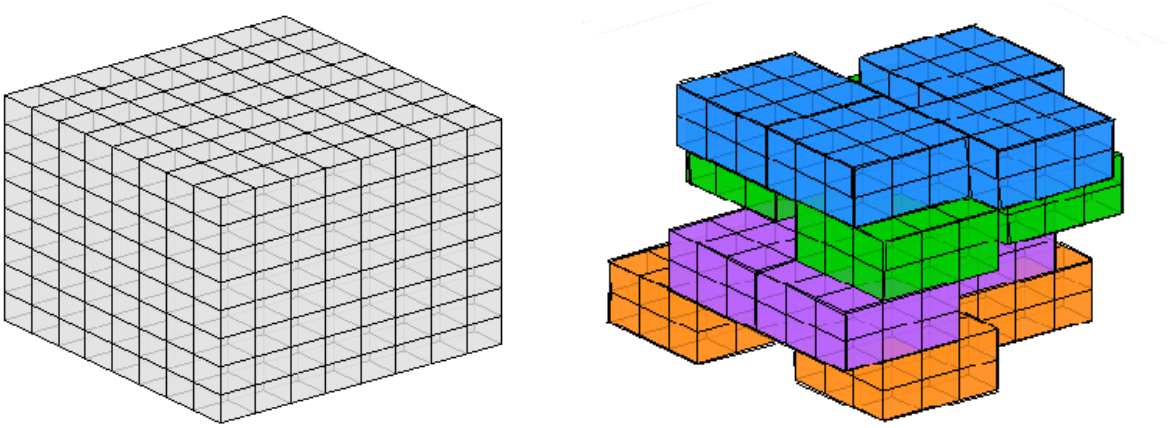


Figure 3.10. Block model and the optimum stopes layout based on LOL method

Discussion and comparison the results of LOT and LOL methods is the last step in stopes layout optimization.

After defining the optimum stopes layout by LOL method and discovering the best stopes to mine, finding the optimum mining sequence of those stopes during the mine life is a next goal. Determining the optimum mining sequence of stopes is called production scheduling optimization. In this research, two methods are applied to produce the production scheduling, SOT and SOL.

### 3.2.3. Production Scheduling Optimization Based on Total Stopes (SOT)

The overall process of the proposed algorithm to implement the production scheduling includes seven steps. Figure 3.11 indicates these steps.

#### 3.2.3.1. Define the Life of Mine (T)

At the first step based on the optimum stopes layout information, the life of mine is defined. In this research, the total tonnage of selected stopes in the optimum stopes layout is considered as the

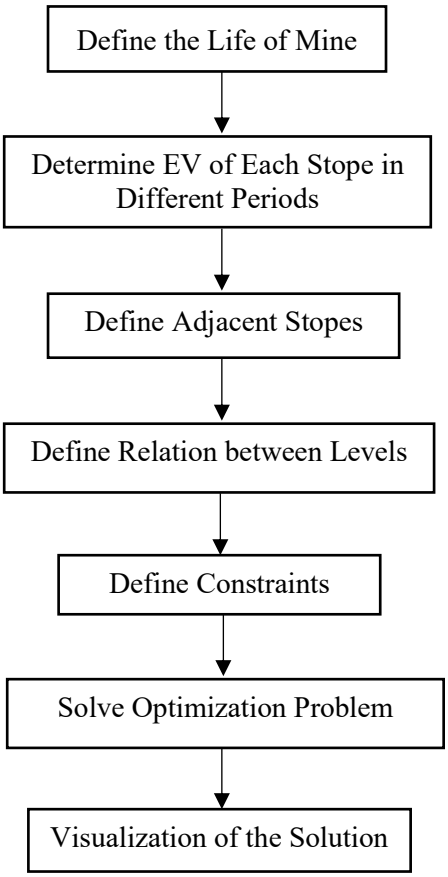



Figure 3.11. The overall process of the algorithm of production scheduling optimization in SOT method

total amount of material needs to be mined. As a result, the total tonnage of material within the stopes located in the optimum layout is considered as the mining tonnage. Based on the total tonnage of material within the stopes, the production rate can be calculated and then the life of mine is determined. In order to calculate the production rate, Long's rule is used. Long's studied on relationship between expected tonnage of deposit and production rate. He assessed 197 underground mines in America and Australia and found this relationship based on equation (3.4) (Dominski et al., 2014). Equation (3.5) is used to calculate the life of mine.

$$\mathrm{Pr} = 0.297 \times Ton^{0.562} \tag{3.4}$$

$$M_l = \frac{Ton}{Pr \times Op} \tag{3.5}$$

Where

$Pr$ : Production rate (ton/day)

*Ton*: Expected tonnage (ton)

$M_l$: Life of mine (year)

*OP*: Number of operation days (day)

### 3.2.3.2. Determine Economic Value of Each Stope in Different Periods

Each selected stope in the optimum stopes layout have a known EV (see section 3.2.2.3). This EV is based on the costs and selling price in the present time, however, to organize the production scheduling, the EV at the mining time should be considered. As a result, the discounted economic value ( *DEV* ) is used which is calculated based on equation (3.6).

$$DEV = \frac{EV}{(1+i)^t} \tag{3.6}$$

Where

*DEV :* Discounted economic value

*i :* Discount rate

*t :* Period

### 3.2.3.3. Define Adjacent Stopes

In order to decrease the chance of geotechnical failures, adjacent stopes are not allowed to be mined concurrently during any periods. As a result, defining the adjacent stopes are a critical step. In SOT method, adjacent stopes are defined as the stopes with the shared boundaries in any coordinate planes. Figure 3.12 demonstrates the definition of the adjacent stopes in SOT method by showing a stope and its adjacent stopes around that.

At the first step of defining adjacent stopes, the distances between all blocks are calculated. Then, the allowable distance between two stopes is determined. Allowable distance between two stopes (D) and calculation of distance between two blocks of those stopes (d) are shown in Figure 3.13. If the distance between any blocks of two stopes is less than allowable distance, those two stopes are considered as the adjacent stopes.

### 3.2.3.4. Define the Relation between Levels

In order to determine practical production scheduling, mining the stopes from different levels

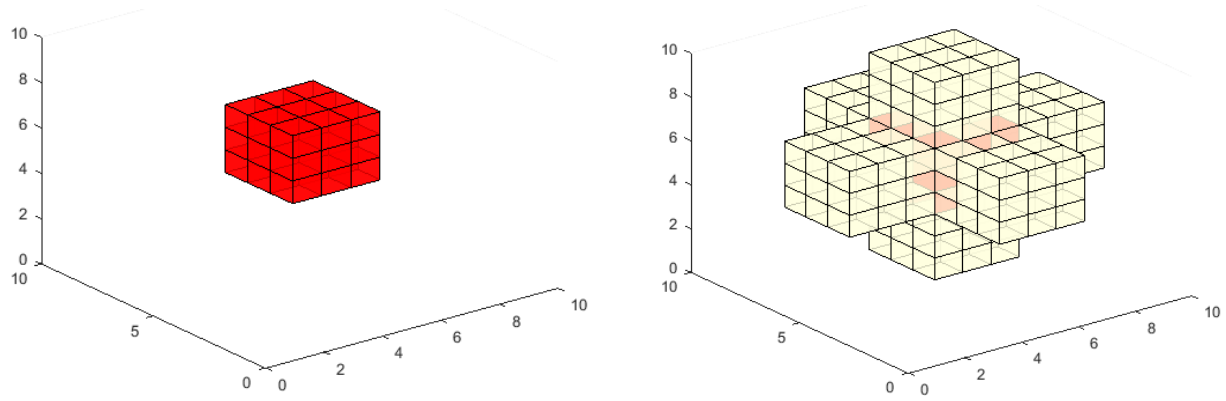


Figure 3.12. A stope and its adjacent stopes in SOT method

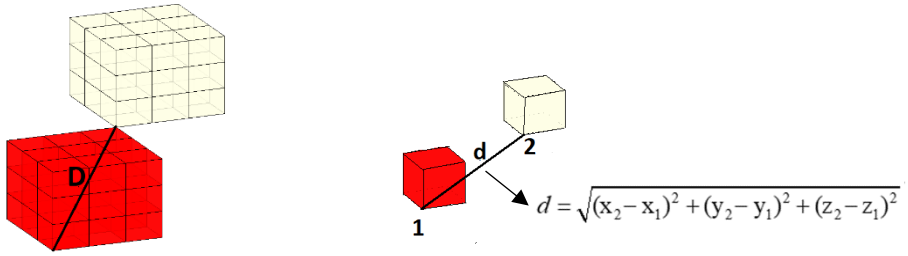



Figure 3.13. Calculation the distance between two blocks and allowable distance between two stopes

should follow the reasonable sequence. For instance, it is not possible to jump from the first level to the third level without starting the mining in the second level. In addition, the reasonable delay between activation of subsequent levels needs to be defined. Also, the maximum number of active levels at the same time should be defined. Different factors such as mining operation capacities can control this number.

### 3.2.3.5. Define Constraints

In addition to mentioned constraints in section 3.2.3.3 and 3.2.3.4, to generate the production schedule, following constraints are considered:

- Mining capacity
- Grade blending
- A desirable direction of mining between levels is essential in the algorithm. It means the mining can be from highest level toward the lowest level or opposite direction.

#### 3.2.3.6. Solve Optimization Problem

After creating the objective function and all the constraints, the optimum stope production schedule with highest NPV is genertaed.

#### 3.2.3.7. Visualization of the Solution

Finally, the last step is displaying the solution.

### 3.2.4. Production Scheduling Optimization Based on Levels (SOL)

As Figure 3.14 indicates, production scheduling optimization in SOL method includes seven steps. This method does the production scheduling in each level separately. In fact, after finishing the scheduling and finding the optimum production schedule on a level, the algorithm starts scheduling at the next level.

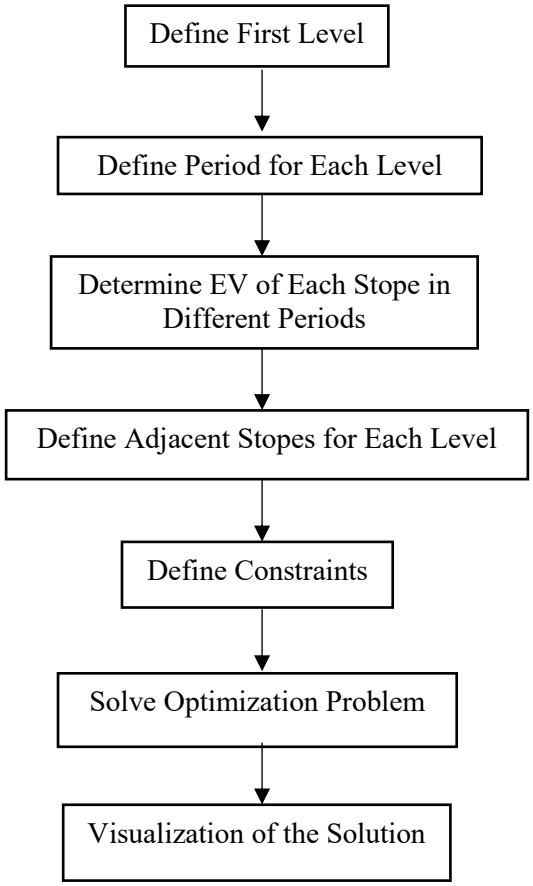



Figure 3.14. The overall process of the algorithm of production scheduling in SOL method

#### 3.2.4.1. Define the First Level

Since starting production scheduling optimization in a level occurs after finishing schedule optimization at the previous level, selecting the first level is critical. In order to choose the first level, a desired direction of mining must be known. If we have a plan on mining from the highest level toward the lowest level, it is evident that the highest level is the first level to do schedule optimization. To be able to compare the result of two methods, the mining direction in SOL method is selected same as mining direction in SOT method.

#### 3.2.4.2. Define Period for Each Level

At the first part based on the information of selected levels and selected stopes by LOL method, the required mining period is defined for each level. The total tonnage of stopes in each level is considered as required mining tonnage in that level. It needs to define a reasonable duration to mine this amount of material in each level. Since there are the different number of stopes in each level, the required mine duration is different for each level.

As it mentioned in section 3.2.3.1, life of mine can be determined by applying Long's rule. The defined mine life is required duration for mining the whole material in all the levels. By distribution, the mine life to levels base on the tonnage of mineral in each level, the required period for each level can be assigned.

#### 3.2.4.3. Determine NPV of Each Stope in Different Periods

Calculation of NPV for each stope has been explained in section 3.2.3.2.

#### 3.2.4.4. Define Adjacent Stopes for Each Level

The reason to define the adjacency has been explained in section 3.2.3.3. In SOL method since all stopes are in the same level with the same Z coordinate, adjacent stopes are defined as the stopes with the shared boundaries in X-Y coordinate planes. Figure 3.15 demonstrates the definition of adjacent stopes in SOL method by indicating a stope and the adjacent stopes around that.

The method of defining the adjacent stopes is described in section 3.2.3.3. The only difference in SOL method is the distance between blocks and the allowable distance are calculated in 2D dimensions.

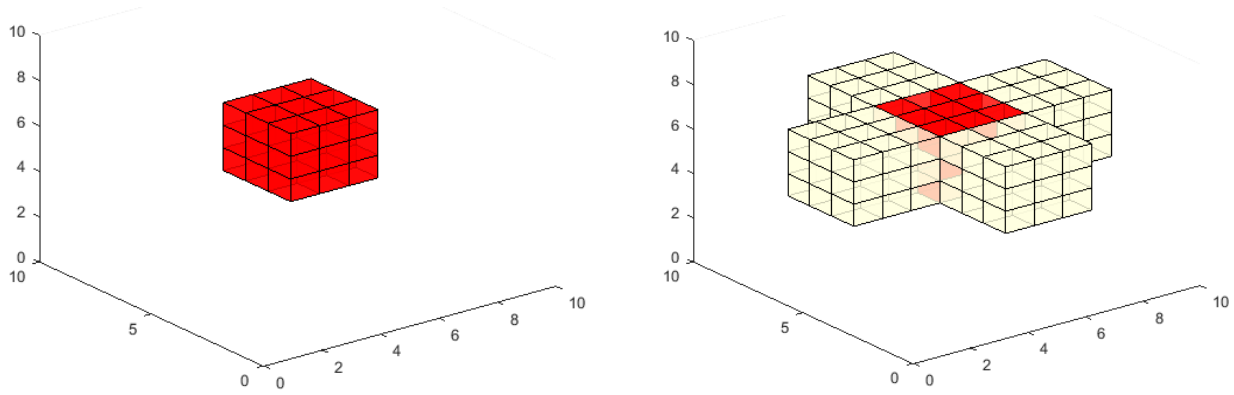



Figure 3.15. A stope and its adjacent stopes in SOL method

#### 3.2.4.5. Define Constraints

For the sake of generating the production schedule in SOL method, the following constraints are considered:

- Mining capacity
- Grade blending
- Precedence between levels

#### 3.2.4.6. Solve Optimization Problem

After considering all conditions, the optimum stope production scheduling with highest NPV is created for each level separately. As a result, the algorithm must be run as many levels as we have.

#### 3.2.4.7. Visualization of the Solution

The last step is displaying the result of optimum stope production scheduling in SOL method. Comparison of the results of SOT and SOL methods is the last step in production scheduling optimization. To plot the solutions in this research, the Plotcube function (PLOTCUBE) is used.

### 3.3. Mathematical Programming Formulation

#### 3.3.1. Layout Optimization Based on Total Blocks (LOT)

In this part, the BIP formulation for finding the optimum stopes layout for LOT method is presented. The purpose is to maximize the EV, while selected stopes in the layout do not have any overlaps.

#### 3.3.1.1. Model Assumptions

The following assumptions are used in the BIP formulations for optimum stopes layout in LOT method.

- The numerical data such as tonnage, grade, coordinates, and economic value are applied to identify the ore-body attributes in each block.
- In order to create stope, no partial block is considered. In other words, a stope consists of the number of complete blocks.
- The present EV of stopes is considered to find the optimum solution. In fact, the factor of time is not considered.

#### 3.3.1.2. Objective Function

The objective function of the BIP formulation is to maximize the EV of stopes in the layout. The EV of each stope depends on the value of the blocks in the stope.

#### 3.3.1.3. Constraints

The following set of constraints is required in the formulation of this method:

***Stopes Overlap***

The primary constraint in the formulation is the overlap condition. It means only stopes without overlap can be in the optimum layout.

***Number of Selected Stopes***

This constraint controls the number of selected stopes in the solution. In fact, the maximum number of stopes in the optimum stopes layout is equal to the total number of positive stopes. This constraint satisfies the condition of the knapsack problem. The knapsack problem will be explained in section 3.3.1.4.

#### 3.3.1.4. BIP Formulations

Finding the optimum stopes layout, which is a combination of positive stopes with highest EV, can be formulated as a knapsack problem with conflict graph.

Pferschy et al. (2009) used the standard 0-1 knapsack problem and added the weight and the incompatibilities for the certain pairs of items as the constraints. They defined that from each conflicting pair the highest value item can be packed into the knapsack. Moreover, to model the conflicts between the items, they used the conflict graph. The graph is represented by a (number of items × number of items) matrix where the value of (i,j) member in the matrix is equal to one if item i and item j cannot be packed together. Based on the definition of the knapsack problem, each item has its utility and weight. In addition, a knapsack has a capacity that is maximum allowable weight in the knapsack.

In this analogy in the selection of stopes, the stopes combination is knapsack, the positive stopes are items that can be picked to put in a knapsack, the weights of all items are equal to one, and the positive stope values are the utilities, and the total number of positive stopes is the capacity of the knapsack.

In order to solve the problem of optimum stopes layout based on LOT method, one set of decision variable is employed. This decision variable is a binary variable and indicates the selection of each stope. The required indices, set, decision variable, and parameters to formulate the problem are as follows:

***Indices***

| | |
|---|---|
| $s \in \{1,...,S\}$ | Index for stopes |

***Set***

| | |
|---|---|
| $Os$ | Set of all stopes overlaps contains all overlaps for each stope |

***Decision variable***

| | |
|---|---|
| $x_s \in \{0,1\}$ | Binary variable controlling the selection of stope s. It is equal to 1, if the stope s is selected in the stopes combination; otherwise is 0. (Decision variable indicating whether item s is picked in knapsack) |

***Parameters***

| | |
|---|---|
| $EV_s$ | Economic value of stope s (Utility of items) |

| | |
|---|---|
| $Ns$ | Total number of positive stopes (Capacity of knapsack) |
| $S$ | Maximum number of positive stopes (Number of items) |

***Objective function***

$$Max\sum_{s=1}^{S} EV_s \times x_s \tag{3.7}$$

***Constraints***

$$x_s + x_{s'} \leq 1, \qquad \forall (s, s') \in Os \tag{3.8}$$

$$\sum_{s=1}^{S} x_s \leq Ns, \qquad \forall s \in \{1, ..., S\} \tag{3.9}$$

The objective function, equation (3.7) consists of the stopes EV and a binary decision variable that indicates the selection or not the selection of each stope in the combination. The stope combination with highest EV is the output of this objective function. The stopes overlap constraint, equation (3.8) ensures that not two stopes, $s$ and $s'$ with overlap can be in the same stope combination. The number of selected stopes constraint, equation (3.9), presents the maximum allowable number of stopes in the combination that is equal to the total number of positive stopes.

### 3.3.2. Layout Optimization Based on Levels (LOL)

In this section, the BIP formulation for finding the optimum stopes layout for LOL method is presented. The purpose of the algorithm not only is maximizing the EV from stopes selection in each level but also is maximizing the EV from levels selection, while selected stopes in each level, as well as selected levels, do not have any overlaps.

#### 3.3.2.1. Model Assumptions

Beside the mentioned assumption for LOT method, the following assumptions are applied in the BIP formulations for finding the optimum stopes layout in LOL method.

- In order to create a level, no partial stope is considered. In fact, levels must create only at top or bottom of the stopes, not middle.

- A level covers of all possible stopes with the same base elevation, not some of those stopes.

#### 3.3.2.2. Objective Function

In the LOL method, the objective function of the BIP formulation is to maximize the EV of stopes in each level and the EV of levels compose the objective function.

#### 3.3.2.3. Constraints

The following set of constraints is required in the formulation:

***Stopes Overlap***

This constraint is same as stopes overlap constraint in section 3.3.1.3 and indicates only stopes without overlap can be in the optimum layout in each level.

***Number of Selected Stopes***

This constraint is same as the number of selected stopes constraints in section 3.3.1.3 and controls the maximum number of stopes in the optimum stopes layout in each level.

***Levels Overlap***

The concept of overlap in this constraint is same as overlap in stopes overlap constraint; however, this constraint focuses on overlaps between levels.

***Number of Selected levels***

This constraint controls the number of selected levels in the solution. This constraint is the knapsack problem condition.

#### 3.3.2.4. BIP Formulations

The formulation of finding the optimum stopes in each level in LOL method is same as LOT method. Nevertheless, the problem is not solved only once. The problem should be formulated for every single level, and the optimum stopes should be found at all levels. To find the optimum levels in LOL method, the knapsack problem with conflict graph is applied as well.

In this analogy in levels selection case, the levels set is knapsack, the levels are items that can be picked to put in a knapsack, the weights of all items are equal to one, the level values are the utilities, and the total number of levels is the capacity of the knapsack.

Formulation the problem of selection the stopes in each level is same as what described in section 3.3.1.4. Toward formulating the problem of selection of the optimum levels, one set of decision variable is used. This decision variable is a binary variable and indicates the selection of each level. The required indices, set, decision variable, and parameters to formulate the problem are as followed.

***Indices***

$l \in \{1,..., L\}$ Index for levels

***Set***

$O_l$ Set of all levels overlaps which contain all overlaps for each levels

***Decision variable***

$x_l \in \{0,1\}$ Binary variable controlling the selection of level $l$. It is equal to 1, if the levels $l$ are selected in the levels set; otherwise is 0. (Decision variable indicating whether item $l$ is picked in knapsack)

***Parameters***

$EV_l$ Economic value of level $l$ (Utility of items)

$Nl$ Total number of levels (Capacity of knapsack)

$L$ Maximum number of levels (Number of items)

***Objective function***

$$Max \sum_{l=1}^{L} EV_l \times x_l \tag{3.10}$$

***Constraints***

$$x_l + x_{l'} \leq 1, \quad \forall l \,\&\, l' \in Ol \tag{3.11}$$

$$\sum_{l=1}^{L} x_l \le Nl, \qquad \forall l \in \{1,...,L\} \tag{3.12}$$

The objective function, equation (3.10) consists of the levels EV of levels and a binary decision that indicates the selection or not selection of each level in the optimum level set. The optimum level set with the highest EV is the output of this objective function. The levels overlap constraint, equation (3.11), ensures that two levels $l$ and $l'$ with overlap cannot be in the same level set. The number of selected levels, equation (3.12), presents the maximum allowable number of levels in a set that is equal to the total number of levels.

### 3.3.3. Production Scheduling Optimization Based on Total Stopes (SOT)

In this section, the BIP formulation to optimize the production scheduling based on SOT method is presented. The purpose is to maximize NPV net present value while considering mining capacity, grade blending, only one-time mining, stope adjacency, the connection between mining the stopes and activation of levels, concurrent active levels and delay between activation of the levels conditions.

#### 3.3.3.1. Model Assumptions

The following assumptions are used in the BIP formulations for production scheduling optimization in SOT method.

- No partial stope is considered in production scheduling. In other words, a stope must thoroughly be mined in a period or not be mined at all.
- When the model selects a stope in a period, it includes all the required preparation, extraction and backfilling of that stope during that period. In this research, the combination of these three functions called mining.
- The market fluctuations during mine life are not considered.
- There is no material mixing between stopes and within a stope during mine operation.

#### 3.3.3.2. Objective Function

The objective function of the BIP formulation is to maximize the NPV of the mining operation.

#### 3.3.3.3. Constraints

The following set of constraints is required in the formulation of this method:

***Mining capacity***

This constraint includes all operation capacities such as drilling, blasting and hauling system. The desired production amount can be achieved by this constraint because this constraint controls the tonnage of material mined in each period.

***Grade blending***

This constraint controls the production's average grade mined from stopes in each period.

***Only one-time mining***

This constraint guarantees each stope is mined only one-time during mine life.

***Stop adjacency***

To drop the chance of geotechnical failures, adjacent stopes are not allowed to be mined during the same period.

***Connection between mining the stopes and activation of levels***

This constraint controls the connection between a level and the stopes in that level. In fact, the stopes in a level can be mined if that level is active. This constraint makes a connection between two required sets of decision variables in the formulation of SOT method.

***Concurrent active levels***

This constraint controls the maximum number of active level at a period. A level is active while is being prepared, extracted or backfilled.

***Delay between activation of the levels***

This constraint guarantees the reasonable time delay between subsequent levels.

#### 3.3.3.4. BIP Formulations

In order to solve the problem of finding optimum production schedule in SOT method, two sets of decision variable are employed. These decision variables are binary variables. One of the binary decision variables ( $y_{l*}^{t}$ ) indicates the activation of levels and one ( $x_{s*}^{t}$ ) controls the mining of the

stopes. The required indices, sets, decision variables, and parameters to formulate the problem are as followed. It should be noted that stopes and levels in this section are the output of stopes layout optimization by LOL method and are shown by * ($s*$ and $l*$).

***Indices***

| | |
|---|---|
| $s* \in \{1,...,S*\}$ | Index for selected stopes as the output of LOL and input of SOT |
| $l* \in \{1,...,L*\}$ | Index for selected levels as the output of LOL and input of SOT |
| $t \in \{1,...,T\}$ | Index for scheduling period in SOT method |

***Sets***

| | |
|---|---|
| $A_{s*}$ | Set of all stopes adjacency which contains all adjacency for every stopes |
| $B_{l*}^{s*}$ | Set of the stopes $s*$ in level $l*$ |

***Decision variables***

| | |
|---|---|
| $x_{s*}^{t} \in \{0,1\}$ | Binary variable controlling mining of stope $s*$ in period t. It is equal to 1, if the stope s* is mined in period t; otherwise is 0. |
| $y_{l*}^{t} \in \{0,1\}$ | Binary variable controlling the activation of level $l*$ in period t. It is equal to 1, if the level $l*$ is active in period t; otherwise is 0. |

***Parameters***

| | |
|---|---|
| $EV_{s*}$ | Economic value of stope $s*$ in SOT method |
| $S*$ | Maximum number of stopes |
| $T$ | Maximum number of scheduling periods (Mine life) |
| $L*$ | Maximum number of levels $l*$ |
| $Ton_{s*}$ | Tonnage of stope $s*$ |

| | |
|---|---|
| $Cu_t$ | Upper bound of mining capacity in period t |
| $Cl_t$ | Lower bound of mining capacity in period t |
| $G_{s^*}$ | Average grade of stope $s^*$ |
| $Gu_t$ | Upper bound of the acceptable average grade in period t |
| $Gl_t$ | Lower bound of the acceptable average grade in period t |
| $N_{l^*}^{s^*}$ | Number of selected stopes $s^*$ in level $l^*$ |
| $Mcl$ | Maximum number of concurrent active levels |
| $D$ | The required delay between activation of levels |
| $\mathrm{R_t}$ | Metal processing recovery in period t |

***Objective function***

$$Max\sum_{t=1}^{T}\sum_{s^*=1}^{S^*}\frac{EV_{s^*}}{(1+\mathrm{i})^{\mathrm{t}}}\times x_{s^*}^{t} \tag{3.13}$$

***Constraints***

$$Cl_t \le \sum_{s^*=1}^{S^*} Ton_{s^*}\times x_{s^*}^{t} \le Cu_t, \qquad \forall t \in \{1,...,T\} \tag{3.14}$$

$$Gl_t \le \sum_{s^*=1}^{S^*} Ton_{s^*}\times x_{s^*}^{t}\times G_{s^*}\times \mathrm{R_t} \le Gu_t, \qquad \forall t \in \{1,...,T\} \tag{3.15}$$

$$\sum_{t=1}^{T} x_{s^*}^{t} = 1, \qquad \forall s^* \in \{1,...,S^*\} \tag{3.16}$$

$$x_{s^*}^{t} + x_{s^{*'}}^{t} \le 1, \qquad \forall s^* \,\&\, s^{*'} \in As^* \tag{3.17}$$

$$\sum_{s^* \in B_{l^*}^{s^*}} x_{s^*}^t \leq N_l^{s^*} \times y_{l^*}^t, \qquad \forall t \in \{1,...,T\}, l^* \in \{1,...,L^*\} \tag{3.18}$$

$$\sum_{l^*}^{L^*} y_{l^*}^t \leq Mcl, \qquad \forall t \in \{1,...,T\} \tag{3.19}$$

$$\mathrm{y}_{l^*}^t \leq \sum_{t'=1}^{t-D} y_{l^*+1}^{t'}, \qquad \forall t \in \{1,...,T\}, l^* \in \{1,...,L^*\} \tag{3.20}$$

The objective function, equation (3.13) consists of the EV of selected stope, discount rate, and a binary decision that indicates mining or not mining of each stope in each period. The stope with the highest EV is chosen to be part of the production in order to maximize the NPV.

The required constraints are presented by equations (3.14) to (3.20). Equation (3.14) represents the mining capacity. The binary variable $x_{s^*}^t$ controls this constraint. One constraint should be defined for each period.

Equation (3.15) ensures that the production's average grade is in the acceptable range. The binary variable $x_{s^*}^t$ controls this constraint. One constraint per period is required.

Equation (3.16) shows every single stope $s^*$ must be mined only once. The binary variable $x_{s^*}^t$ controls this constraint. One constraint should be defined for each stope.

Equation (3.17) is related to the adjacent stopes. This constraint is controlled by the binary variable $x_{s^*}^t$. This constraint indicates stopes $s^*$ and $s^{*\prime}$must not be mined at the same period, if $s^*$ and $s^{*\prime}$belong to $A_{s^*}$.

Equation (3.18) presents the connection between mining the stopes and activation of levels. This constraint is controlled by binary variables $x_{s^*}^t$ and $y_{l^*}^t$. $x_{s^*}^t$ controls stopes that belong to level $l^*$ and $y_{l^*}^t$ controls the levels. One constraint should be defined for each level in each period.

Equation (3.19) is defined for concurrent active levels. This constraint is controlled by the binary variable $y_{l^*}^t$. One constraint should be defined for each period.

Equation (3.20) keeps the desired delay between the activation of levels. The binary variable $y^t_{l*}$ controls this constraint. One constraint should be defined for each level in each period.

### 3.3.4. Production Scheduling Optimization Based on Levels (SOL)

In this part, the BIP formulation for optimizing the production scheduling for SOL method is presented. The purpose is to maximize NPV of mining stopes in each level separately while considering mining capacity, grade blending, only one-time mining, stope adjacency.

#### 3.3.4.1. Model Assumptions

In addition to assumptions in section 3.3.3.1, the following assumptions are used in the BIP formulations for production scheduling optimization in SOL method.

- Only based on the proportion of material tonnage in each level out of total material tonnage, the proportion of mine life is assigned to each level and other factors are not considered.
- Production scheduling is done at each level separately, and the connection between levels is not considered.
- Mining capacity and required average grade in each period in SOL method are assumed same as SOT method other factors are not considered.

#### 3.3.4.2. Objective Function

The objective function of the BIP formulation is to maximize NPV from stope mining in each level independently.

#### 3.3.4.3. Constraints

The following set of constraints is required in the formulation of this method:

***Mining capacity***

This constraint includes all operation capacities such as drilling, blasting and hauling system. The desired production amount can be achieved by this constraint because this constraint controls the tonnage of material mined in each period.

***Grade blending***

This constraint controls the production's average grade mined from stopes in each period.

***Only one-time mining***

This constraint guarantees each stope in a level is mined only one-time during mining period of each level.

***Stope adjacency***

In order to decrease the geotechnical failures, adjacent stopes in each level are not allowed to be mined during the same period.

**3.3.4.4. BIP Formulations**

In order to solve the problem of finding optimum production schedule in SOL method, one set of decision variable is employed. This decision variable is a binary variable controls the mining of the stopes in each level. In SOL method, the BIP formulation and running the model must be applied for each level separately. It should be noted that stopes and levels in this section are the output of stopes layout optimization by LOL method and shown by as $s*$ and $l*$. The required indices, sets, decision variables, and parameters to formulate the problem are as follows.

***Indices***

| | |
|---|---|
| $s^*_{l*} \in \{1,...,S^*_{l*}\}$ | Index for selected stopes $s*$ in selected level $l*$ as the output of LOL method |
| $t_{l*} \in \{1,...,T_{l*}\}$ | Index for scheduling period in level $l*$ in SOL method |

***Set***

| | |
|---|---|
| $A^{s*}_{l*}$ | Set of all stopes adjacency which contains all adjacency for every stopes |

***Decision variable***

| | |
|---|---|
| $x^{t_{l*}}_{s*,l*} \in \{0,1\}$ | Binary variable controlling mining of stope $s*$ of level $l*$ in period $t_{l*}$ in SOL method. It is equal to 1, if the stope s*is mined in period t; otherwise is 0. |

***Parameters***

| | |
|---|---|
| $EV_{l*}^{s*}$ | Economic value of stope s* of level l* in SOL method |
| $S_{l*}^{*}$ | Maximum number of stopes in level $l*$ |
| $T_{l*}$ | Maximum number of scheduling periods for level $l*$ |
| $Ton_{l*}^{s*}$ | Tonnage of stope $s*$ in level $l*$ |
| $Cu_{l*}^{t_{l*}}$ | Upper bound of mining capacity in level $l*$ in period $t_{l*}$ |
| $Cl_{l*}^{t_{l*}}$ | Lower bound of mining capacity in level $l*$ in period $t_{l*}$ |
| $G_{l*}^{s*}$ | Average grade of stope $s*$ in level $l*$ |
| $Gu_{l*}^{t_{l*}}$ | Upper bound of the acceptable average grade in level $l*$ in period $t_{l*}$ |
| $Gl_{l*}^{t_{l*}}$ | Lower bound of the acceptable average grade in level $l*$ in period $t_{l*}$ |
| $R_{l*}^{t}$ | Metal processing recovery in period $t_{l*}$ |

***Objective function***

$$Max \sum_{t_{l*}=1}^{T_{l*}} \sum_{s_{l*}^{*}=1}^{S_{l*}^{*}} \frac{EV_{l*}^{s*}}{(1+i)^{t_{l*}}} \times x_{s*,l*}^{t_{l*}} \tag{3.21}$$

***Constraints***

$$Cl_{l*}^{t_{l*}} \le \sum_{s_{l*}^{*}=1}^{S_{l*}^{*}} Ton_{l*}^{s*} \times x_{s*,l*}^{t_{l*}} \le Cu_{l*}^{t_{l*}}, \qquad \forall t_{l*} \in \{1,...,T_{l*}\} \tag{3.22}$$

$$Gl_{l*}^{t_{l*}} \le \sum_{s_{l*}^{*}=1}^{S_{l*}^{*}} Ton_{l*}^{s*} \times x_{s*,l*}^{t_{l*}} \times G_{l*}^{s*} \times R_{l*}^{t_{l*}} \le Gu_{l*}^{t_{l*}}, \qquad \forall t_{l*} \in \{1,...,T_{l*}\} \tag{3.23}$$

$$\sum_{t_{l^*}=1}^{T_{l^*}} x_{s^*,l^*}^{t_{l^*}} = 1, \qquad \forall s_{l^*}^* \in \{1,...,S_{l^*}^*\} \tag{3.24}$$

$$x_{s^*,l^*}^{t_{l^*}} + x_{s'^*,l^*}^{t_{l^*}} \leq 1, \qquad \forall s_{l^*}^* \,\&\, s_{l^*}^{*'} \in A_{l^*}^{s^*} \tag{3.25}$$

The objective function, equation (3.21) consists of the economic value of selected stopes in selected levels, discount rate, and a binary decision that indicates mining or not mining of each stope of each level in each period.

The constraints are presented by equations (3.22) to (3.25). Equation (3.22) represents the mining capacity. This constraint is controlled by the binary variable $x_{s^*,l^*}^{t_{l^*}}$. One constraint should be defined for each period in a level.

Equation (3.23) ensures that the production's average grade in each level is in the acceptable range. This constraint is controlled by the binary variable $x_{s^*,l^*}^{t_{l^*}}$. One constraint per period in a level is required.

Equation (3.24) indicates every single stope $s_{l^*}^*$ in each level must be mined only once. The binary variable $x_{s^*,l^*}^{t_{l^*}}$ controls this constraint. One constraint should be defined for each stope in a level.

Equation (3.25) is related to the adjacent stopes. This constraint is controlled by the binary variable $x_{s^*,l^*}^{t_{l^*}}$. This constraint indicates stopes $s_{l^*}^*$ and $s_{l^*}^{'*}$ must not be mined at the same period, if $s_{l^*}^*$ and $s_{l^*}^{'*}$ belong to $A_{l^*}^{s^*}$.

### 3.4. BIP Formulation Implementation

In this research, MATLAB (MathWorks Inc, 2017) is employed as a numerical modeling platform and IBM ILOG CPLEX Optimization Studio 12.7.1 (IBM, 2017) as a solver to optimize stopes layout. MATLAB is a high-level language to analyze data, develop algorithms and models. CPLEX solver is a tool to solve a large-scale mixed-integer linear and quadratic programming (Pourrahimian, 2013).

### 3.4.1. Numerical Modeling

The BIP formulation for mine problems optimization usually creates a large-scale optimization problem. CPLEX is an appropriate optimization solver to handle such a comprehensive problem. Using a branch-and-cut algorithm makes CPLEX able to solve large-scale problems with BIP formulation. Branch-and-cut is a combinatorial optimization method to solve the integer-programming problems (Horst et al., 1996; Wolsey, 1998). An optimization termination criterion in CPLEX is the gap tolerance (EPGAP). In fact, the EPGAP places an absolute tolerance between the best integer objective and the objective of the remained the best node to measure the optimality. Also, CPLEX is able to terminate when a feasible integer solution within the set EPGAP is found (Pourrahimian, 2013; Malaki, 2016).

### 3.4.2. General Formulation

General structure to solve a BIP problem in IBM/CPLEX is as followed equations:

$$\min f.x \tag{3.26}$$

Subject to:

$$Aineq.x \leq bineq \tag{3.27}$$

$$Aeq.x = beq \tag{3.28}$$

$$lb \leq x \leq ub \tag{3.29}$$

Where

$f$ : Column vector for linear objective function

$x$ : Decision variable of the model

$Aineq$ : Matrix for linear inequality constraints

$bineq$ : Column vector for linear inequality constraints

$Aeq$ : Matrix for linear equality constraints

$beq$ : Column vector for linear equality constraints

$lb$ : Column vector of lower bounds

$ub$ : Column vector of upper bounds

#### 3.4.2.1. Layout Optimization Based on Total Blocks (LOT)

- **The BIP Objective Function**

The objective function of the finding optimum stopes layout problem as stated by equations (3.7) is to maximize EV. As mentioned in equation (3.26), the general form of the objective function in CPLEX is minimization. Therefore, the coefficient vector of the objective function (*f*) in equations (3.7) should be multiplied by a negative sign. As a result, the objective function will change to minimizing the –EV.

Table 3.1 shows the size and structure of the coefficient vector of objective functions in LOT method. $Ns$ is the number of stopes and $\boldsymbol{EV}$ is a $1\times Ns$ vector which contains the economic value of the stopes.

Table 3.1. Size and structure of the coefficient vector of objective function

| Method | Size | Structure |
|---|---|---|
| LOT | $1\times Ns$ | $[EV]$ |

Table 3.2 indicates the size of the vector for decision variables. $\boldsymbol{X}$ is a $Ns\times 1$ vector holding the binary decision variables controlling the selection of stopes in the optimum stopes layout.

Table 3.2. Size and structure of the decision variables' vector

| Method | Size | Structure |
|---|---|---|
| LOT | $Ns\times 1$ | $[X]$ |

- **The Constraints of the BIP Models**

The constraints of the BIP model for layout optimization in LOT method are presented by equations (3.8) and (3.9).

Table 3.3 states the number of rows for each constraint. $No$ is the number of stopes overlaps. Figure 3.16 demonstrates the structure of these constraints' coefficient matrix.

In the following section, the structure of each constraint will be described in detail. It should be noted that in order to save the memory space, the size of some constraints coefficient matrices

needs to be reduced. In order to decrease the size of matrices, MATLAB's sparse function is applied.

Table 3.3. Number of rows in constraint' coefficient matrix

| Constraints | Number of rows |
|---|---|
| Stopes Overlap | $No \times Ns$ |
| Number of Selected Stopes | $1 \times Ns$ |

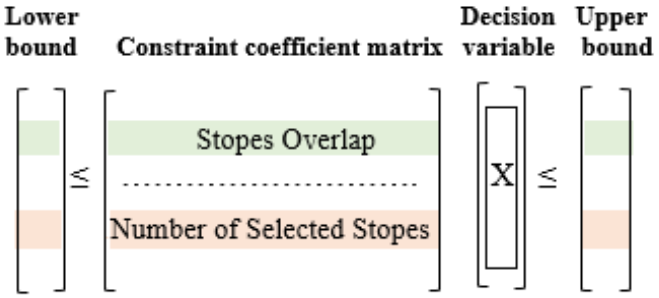



Figure 3.16. Order of constraints in the constraint coefficient matrix in the formulation of LOT method

***Stopes Overlap***

Based on equation (3.8), two stopes with overlap cannot be in the solution. As a result, the constraint coefficient matrix should contain every single overlap that each stope has. It means the constraint coefficient matrix consist of $No$, the total number of overlaps, as rows and $Ns$, the total number of stopes, as columns. Each rows contains two 1 elements to show an overlap between two stopes. The structure of this constraint in the formulation is as equation (3.30).

$$[X_{so}] \leq [ub_{so}] \tag{3.30}$$

Equation (3.31) shows the structure of the coefficient matrix of this constraint. The first row of this matrix shows if first stope and second stope have overlap. If yes, $O_{s1\&s2}$ is equal to 1; otherwise this row does not need to be in the matrix. The fourth row shows if first stope and last stope have overlap and the second last row indicates if second last stope and last stope have overlap. This order is repeated for all other stopes. $ub_{so}$ matrix is all 1 elements with $No$ rows and one column.

***Number of Selected Stopes***

Based on the equation (3.9), the maximum allowable number of stopes in the solution should be considered. This number is equal to the total number of positive stopes. In fact, this constraint is

$$\begin{bmatrix} 1 & O_{s1\&s2} & 0 & . & . & . \\ 1 & 0 & O_{s1\&s3} & 0 & . & . \\ . & . & . & . & . & . \\ 1 & 0 & . & . & . & O_{s1\&sNs} \\ 0 & 1 & O_{s2\&s3} & 0 & . & . \\ . & . & . & . & . & . \\ . & . & . & 0 & 1 & O_{s2\&sNs} \\ . & . & . & . & . & . \end{bmatrix} \tag{3.31}$$

$$[X_{sn}] \leq [ub_{sn}] \tag{3.32}$$

to define the capacity of the knapsack problem. The structure of this constraint is as equation (3.32).Coefficient matrix of this constraint has $1 \times Ns$ elements of all 1 elements. Also, $ub_{sn}$ matrix has one element is equal to $Ns$.

### 3.4.2.2. Layout Optimization Based on Levels (LOL)

- **The BIP Objective Function**

In the LOL method, the objective function of the BIP formulation is to maximize the EV from selection stopes in each level and selection levels, equation (3.7) and (3.10). As it mentioned to find the solution by LOL method, the BIP formulation should be run in each level separately to select stopes in each level and then another BIP formulation should be run to select levels. As explained in section 3.4.2.1, the objective function is minimizing the –EV of stopes selection and levels selection.

Table 3.4 shows the size and structure of the coefficient vector of objective functions in LOL method. $N_l^s$ is the number of stopes in each level, and $\boldsymbol{EV}_l^s$ is a $1 \times N_l^s$ vector includes the EV of stopes in each level. Also, $Nl$ is the number of levels and $\boldsymbol{EV}_l$ is a $1 \times Nl$ vector includes the EV of levels.

Table 3.4. Size and structure of the coefficient vector of objective function

| Method | Size | Structure |
|---|---|---|
| LOL (stope selection in each level) | $1 \times N_l^s$ | $[EV_l^s]$ |
| LOL (level selection) | $1 \times Nl$ | $[EV_l]$ |

Table 3.5 indicates the size of the vectors for decision variables $\boldsymbol{X}_l^s$ and $\boldsymbol{X}_l$. $\boldsymbol{X}_l^s$ is a $N_l^s \times 1$ vector holding the binary decision variables controlling the selection of stopes in each level. Besides, $\boldsymbol{X}_l$ is a $Nl \times 1$ vector holding the binary decision variables controlling the selection of levels in an optimum layout.

Table 3.5. Size and structure of the decision variables' vector

| Method | Size | Structure |
|---|---|---|
| LOL (Stope selection in each level) | $N_l^s \times 1$ | $[X_l^s]$ |
| LOL (level selection) | $Nl \times 1$ | $[X_l]$ |

- **The Constraints of the BIP Models**

The constraints of the BIP model for layout optimization in LOL method are presented by equations (3.8) and (3.9) for stope selection in each level and by equations (3.11) and (3.12) for level selection. Table 3.6 states the number of rows for each constraint. $No_l^s$ is the number of all stopes overlaps in each level and $No_l$ is number of all levels overlaps. Figure 3.17 demonstrates the structure of these constraints' coefficient matrix.

Table 3.6. Number of rows in constraint' coefficient matrix

| Constraints | Number of rows |
|---|---|
| Stopes overlap in each level | $No_l^s \times N_l^s$ |
| Number of selected stopes | $1 \times N_l^s$ |
| Levels overlap | $No_l \times Nl$ |
| Number of selected levels | $1 \times Nl$ |

In the following section, the structure of each constraint will be described in detail.

***Stopes Overlap***

Based on equation (3.7) two stopes with overlap cannot be in the selected stopes in each level. As a result, the constraint coefficient matrix consists of $No_l^s$, total number of stopes overlaps in a level, as the number of rows and $N_l^s$, the total number of stopes at that level, as the number of columns. Each row contains two 1 elements to show an overlap between two stopes.

The structure of this constraint in the formulation for stope selection in each level is similar to equation (3.30), and the structure of the coefficient matrix of this constraint is similar to equation

(3.31). Since the number of stopes in each level are different, the upper bound matrix of different levels does not have the same dimension.

***Number of Selected Stopes***

Based on equation (3.8), the maximum allowable number of stopes in the solution is considered. This number for each level is equal to the total number of positive stopes in that level. The structure of this constraint is similar to equation (3.32). Coefficient matrix of this constraint has $1 \times N_l^s$ elements of 1. Also, $\boldsymbol{ubsn}_l^s$ matrix has one element is equal to $N_l^s$ in each level.

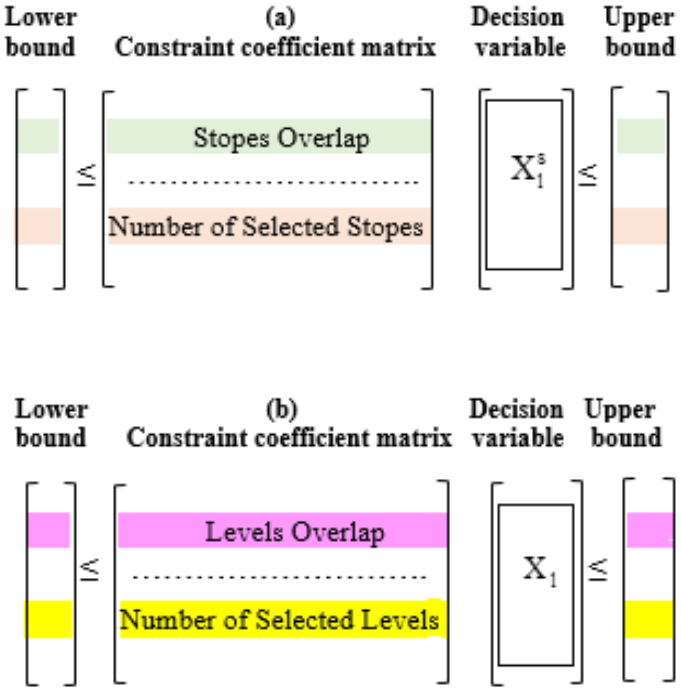



Figure 3.17. Order of constraints in the constraint coefficient matrix in the formulation of selection stopes in each level (a) and selection of levels (b) in LOL method

***Levels Overlap***

Based on equation (3.11), two levels with overlap cannot be in the solution. As a result, the constraint coefficient matrix should contain every single overlap that each level has. It means the constraint coefficient matrix consist of $No_l$, the total number of levels overlaps, as rows and $Nl$, the total number of levels, as columns. Each row contains two 1 elements to show an overlap. The structure of this constraint in the formulation is as equation (3.33).

$$[X_{lo}] \leq [ub_{lo}] \tag{3.33}$$

Equation (3.34) shows the structure of the coefficient matrix of this constraint. This matrix has $No_l \times Nl$ elements. The first row of this matrix shows if the first level and second level have overlap. If yes, $O_{l1\&l2}$ is equal to 1, it will be kept in the matrix; otherwise this row does not need to be in the matrix. The fourth row shows if the first level and the last level have overlap, and the second last row indicates if the second last level and the last level have overlap. This order is continued for all other levels. $\boldsymbol{ub_{lo}}$ matrix is all 1 elements with $Nl$ rows and one column.

$$\begin{bmatrix} 1 & O_{l1\&l2} & 0 & . & . & . \\ 1 & 0 & O_{l1\&l3} & 0 & . & . \\ . & . & . & . & . & . \\ 1 & 0 & . & . & . & O_{l1\&lNl} \\ 0 & 1 & O_{l2\&l3} & 0 & . & . \\ . & . & . & . & . & . \\ 0 & 1 & 0 & . & . & O_{l2\&lNl} \\ . & . & . & . & . & . \end{bmatrix} \tag{3.34}$$

***Number of Selected Levels***

Based on equation (3.12), the maximum allowable number of levels in the solution is considered which is equal to the total number of levels. This constraint is to define the capacity of the knapsack problem. The structure of this constraint is as equation (3.35).

$$[X_{ln}] \leq [ub_{ln}] \tag{3.35}$$

The coefficient matrix of this constraint is a $1 \times Nl$ matrix of all 1 elements. $ub_{ln}$ matrix has one element equals to $Nl$.

### 3.4.2.3. Production Scheduling Optimization Based on Total Stopes (SOT)

- **The BIP Objective Function**

The objective function of the finding optimum stopes layout problem as stated by equations (3.13) is to maximize NPV. As mentioned in previous sections, the general form of the objective function in CPLEX is minimization. Therefore, the objective function will change to minimizing the – NPV

of the mining operation. Table 3.7 shows the size and structure of the coefficient vector of the objective function of SOT method. The second column of the table shows the size of this matrix; where $Ns*$ is the number of selected stop by LOL method, $Nl*$ is the number of selected levels as an output of LOL method. In the third column of the table, $\boldsymbol{DEV}_{s*}$ is a $1\times(Ns*\times T)$ vector that contains the discounted economic valve of the selected stope. Besides, $\mathbf{0}$ is a $1\times(Nl*\times T)$ vector with all elements equal to zero where $T$ is the number of scheduling periods.

Table 3.7. Size and structure of the coefficient vector of objective function

| Method | Size | Structure |
|---|---|---|
| SOT | $1\times((Ns*+Nl*)\times T)$ | $[DEV_{s*},0]$ |

Table 3.8 indicates the size of the vector for decision variables. $\boldsymbol{X}$ is a $(Ns*\times T)\times 1$ vector holding binary decision variables controlling the mining of selected stopes $x_{s*}^{t}\in\{0,1\}$. $\boldsymbol{Y}$ is a $(Nl*\times T)\times 1$ vector contains binary decision variables controlling the activation of the selected level $y_{s*}^{t}\in\{0,1\}$

Table 3.8. Size and structure of the decision variables vector

| Method | Size | Structure |
|---|---|---|
| SOT | $((Ns*+Nl*)\times T)\times 1$ | $[X;Y]$ |

- **The Constraints of the BIP Models**

The constraints of the BIP model for production scheduling in SOT method are presented by equations (3.14) to (3.20). Table 3.9 states the number of rows for each constraint. The number of columns in all constraint is $((Ns*+Nl*)\times T)$. $Na$ is the total number of stopes adjacency which contains all possible adjacency for all selected stopes.

Figure 3.18 demonstrates the structure of constraints coefficient matrix and order of constraints in the matrix in SOT method. The constraints coefficient matrix itself is divided into different sections based on the decision variables. Figure 3.19 indicates these sections. Since in this research, there are two types of decision variables, the matrix is divided into two sections. Also, each section is divided into smaller sections and the number of smaller sections depends on the number of scheduling periods. These smaller sections are shown in Figure 3.20. Figure 3.20 illustrates the structure of each variable in the constraint coefficient matrix and decision variable vector based on periods. Number of columns in each period for variable $\boldsymbol{X}$ is $Ns*$ and for variable $\boldsymbol{Y}$ is $Nl*$.

In the following section, the structure of each constraint will be described in detail.

Table 3.9. Number of rows in constraint' coefficient matrix

| Constraints | Number of rows |
|---|---|
| Mining capacity | $2T$ |
| Grade blending | $2T$ |
| Only one- time mining | $Ns*$ |
| Adjacency | $Na \times T$ |
| Connection | $Nl* \times T$ |
| Concurrent levels | $T$ |
| Delay between levels | $Nl* \times T$ |

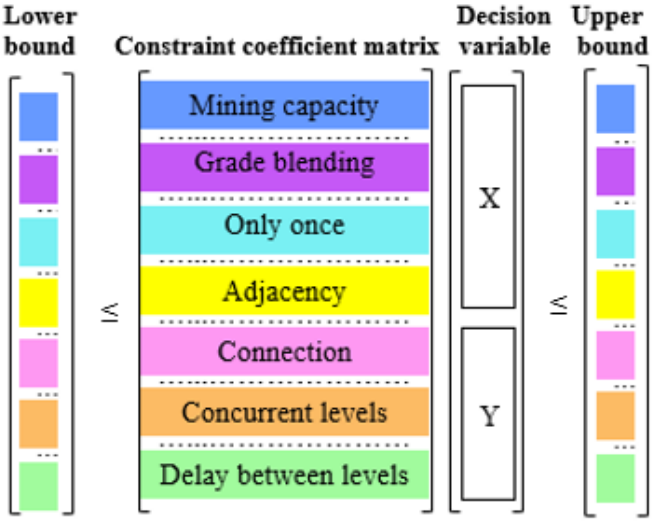



Figure 3.18. Order of constraints in the constraint coefficient matrix in SOT method

***Mining capacity***

Based on equation (3.14) minimum and the maximum of mining capacity should be considered in the formulation, and it is called mining capacity constraint. Mining capacity constraint has $2T$ rows. Equation (3.14) has two parts as following equations:

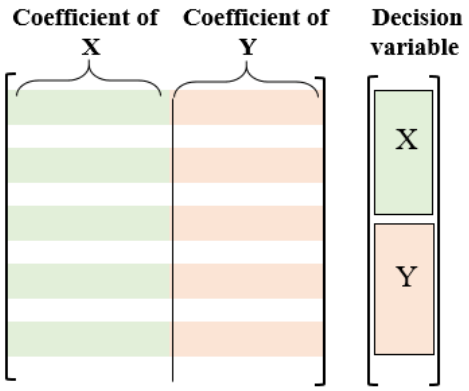



Figure 3.19. Section of the constraints coefficient matrix based on the decision variables in SOT method

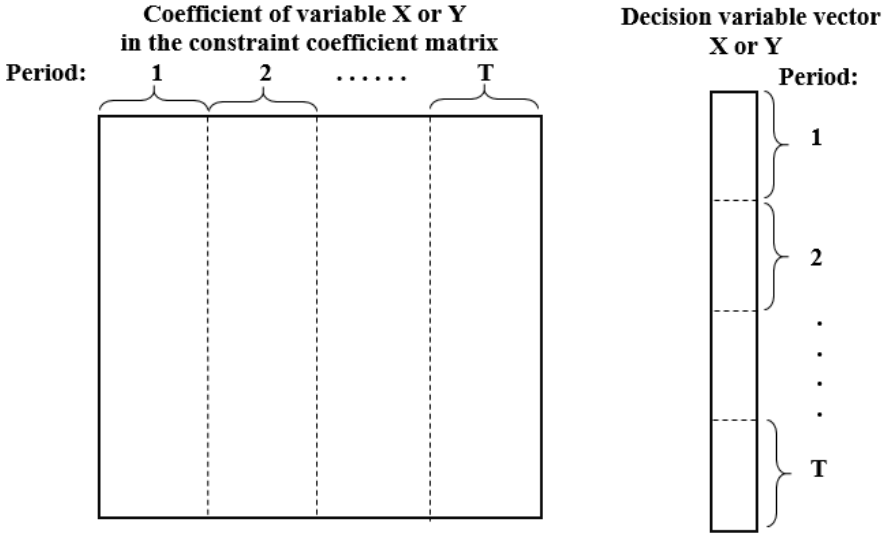



Figure 3.20. The structure of a decision variable and the constraints coefficient matrix based on $T$

$$\sum_{s^*=1}^{S^*} Ton_{s^*} \times x_{s^*}^t \le Cu_t, \qquad \forall t \in \{1,...,T\} \tag{3.36}$$

$$Cl_t - \sum_{s^*=1}^{S^*} Ton_{s^*} \times x_{s^*}^t \le 0, \qquad \forall t \in \{1,...,T\} \tag{3.37}$$

Equation (3.36) indicates the upper bound part of the constraint. The first T rows of mining capacity constraint belong to this part. Equation (3.37) demonstrates the lower bound part of the constraint. The second T rows of mining capacity constraint belong to this part.

Equation (3.38) shows the structure of this constraint. The part related to decision variable $\boldsymbol{X}$ in the coefficient matrix is a $2T \times (Ns^* \times T)$ matrix and $Y$ part of the coefficient matrix is a

$2T \times (Nl^* \times T)$ matrix. $[lb_C]$ and $[ub_C]$ are $2T \times 1$ matrices, and all the elements are equal to zero. Equation (3.39) shows the structure of the area related to the $X_C$ and $Y_C$ in the coefficient matrix of this constraint, each row and each column is related to a period. The left side of this matrix is related to $X_C$ and the right side is related to $Y_C$. The upper part of the matrix is related to the upper bound part of the constraint, and the lower part is related to the lower bound part of the constraint. $X_C$ part includes $t_1,\ldots,t_{Ns^*}$ and $-t_1,\ldots,-t_{S^*}$. $t_1,\ldots,t_{Ns^*}$ is a $1 \times Ns^*$ row vector contains tonnage of the first stope to the last stope and $-t_1,\ldots,-t_{S^*}$ is a $1 \times Ns^*$ row vector contains minus tonnage of all stopes. All elements of $Y_C$ part are zero.

$$[lb_C] \le [X_C \quad Y_C] \le [ub_C] \tag{3.38}$$

$$\left[\begin{array}{cccc|ccc} t_1,\ldots,t_{S^*} & 0 & . & 0 & 0 & . & 0 \\ 0 & t_1,\ldots,t_{S^*} & . & 0 & . & . & . \\ . & . & . & . & . & . & . \\ 0 & 0 & . & t_1,\ldots,t_{S^*} & . & . & . \\ \hline -t_1,\ldots,-t_{S^*} & 0 & . & 0 & . & . & . \\ 0 & -t_1,\ldots,-t_{S^*} & . & 0 & . & . & . \\ . & . & . & . & . & . & . \\ 0 & 0 & . & -t_1,\ldots,-t_{S^*} & 0 & . & . \end{array}\right] \tag{3.39}$$

***Grade blending***

Based on equation (3.15) minimum and the maximum of required average grade should be considered in the formulation, and it is called grade blending constraint. This constraint has *2T* rows. Equation (3.15) has two parts as following:

$$\sum_{s^*=1}^{S^*} Ton_{s^*} \times x_{s^*}^t \times (G_{s^*} - Gu_t) \times R_t \le 0, \qquad \forall t \in \{1,...,T\} \tag{3.40}$$

$$\sum_{s^*=1}^{S^*} -Ton_{s^*} \times x_{s^*}^t \times (G_{s^*} - Gl_t) \times R_t \le 0, \qquad \forall t \in \{1,...,T\} \tag{3.41}$$

Equation (3.40) indicates the upper bound part of the constraint. The first T rows of grade blending constraint belong to this part. Equation (3.41) demonstrates the lower bound part of the constraint. Second T rows of grade blending constraint belong to this part.

Equation (3.42) shows the structure of this constraint. The part related to decision variable $\boldsymbol{X}$ in the coefficient matrix is a $2T\times(Ns*\times T)$ matrix and $\boldsymbol{Y}$ part of the coefficient matrix is a $2T\times(Nl*\times T)$ matrix. $\left[lb_g\right]$ and $\left[ub_g\right]$ are $2T\times 1$ matrices, and all the elements are equal to zero. Equation (3.43) shows the structure of the area related to the $X_g$ and $\boldsymbol{Y}_g$ in the coefficient matrix of this constraint. The left side of this matrix is related to $X_g$ and the right side is related to $\boldsymbol{Y}_g$. The upper part of the matrix is related to the upper bound part of the constraint, and the lower part is related to the lower bound part of the constraint. $X_g$ part includes $g_1,\ldots,g_{S*}$ and $-g_1,\ldots,-g_{S*}$. $g_1,\ldots,g_{S*}$ is a $1\times Ns*$ row vector contains an average grade of the first stope to the last stope and $-g_1,\ldots,-g_{S*}$ is a $1\times Ns*$ row vector contains minus average grade of all stopes. All elements of $\boldsymbol{Y}_g$ part are zero.

$$\left[lb_g\right]\le\left[X_g \;\; Y_g\right]\le\left[ub_g\right] \tag{3.42}$$

$$\left[\begin{array}{cccc|ccc} g_1,\ldots,g_{S*} & 0 & . & 0 & 0 & . & 0 \\ 0 & g_1,\ldots,g_{S*} & . & 0 & . & . & . \\ . & . & . & . & . & . & . \\ 0 & 0 & . & g_1,\ldots,g_{S*} & . & . & . \\ \hline -g_1,\ldots,-g_{S*} & 0 & . & 0 & . & . & . \\ 0 & -g_1,\ldots,-g_{S*} & . & 0 & . & . & . \\ . & . & . & . & . & . & . \\ 0 & 0 & . & -g_1,\ldots,-g_{S*} & 0 & . & 0 \end{array}\right] \tag{3.43}$$

***Only one-time mining***

Based on equation (3.16) each stope must be mined exactly once in all periods. Since this constraint is a linear equality constraint, is formulated based on equation (3.28). This constraint has $Ns*$ rows.

Equation (3.44) indicates the structure of this constraint. The part related to decision variable $\boldsymbol{X}$ in the coefficient matrix is a $Ns*\times(Ns*\times T)$ matrix and $\boldsymbol{Y}$ part of the coefficient matrix is a $Ns*\times(Nl*\times T)$ matrix. Also, $\left[beq_o\right]$ is a $Ns*\times 1$ matrix and all the elements are equal to 1. Equation (3.45) shows the structure of the area related to the $\boldsymbol{X_o}$ and $\boldsymbol{Y_o}$ in the coefficient matrix of this constraint. The left side of this matrix is related to $\boldsymbol{X_o}$, and the right side is related to $\boldsymbol{Y_o}$. At $\boldsymbol{X_o}$ side, each section belongs to a period. All elements of $\boldsymbol{Y_o}$ side are zero.

$$\left[X_o \;\; Y_o\right]=\left[beq_o\right] \tag{3.44}$$

$$\left[\begin{array}{cccccccccccccc|ccc}
1 & 0 & . & 0 & 0 & . & . & . & . & 1 & 0 & . & 0 & 0 & 0 & . & 0 \\
0 & 1 & . & 0 & 0 & . & . & . & . & 0 & 1 & . & 0 & 0 & 0 & . & . \\
. & . & . & . & . & . & . & . & . & . & . & . & . & . & . & . & . \\
. & . & . & . & . & . & . & . & . & . & . & . & . & . & . & . & . \\
0 & 0 & . & 0 & 1 & . & . & . & . & 0 & 0 & . & 0 & 1 & 0 & . & 0
\end{array}\right] \tag{3.45}$$

t=1 … t=T

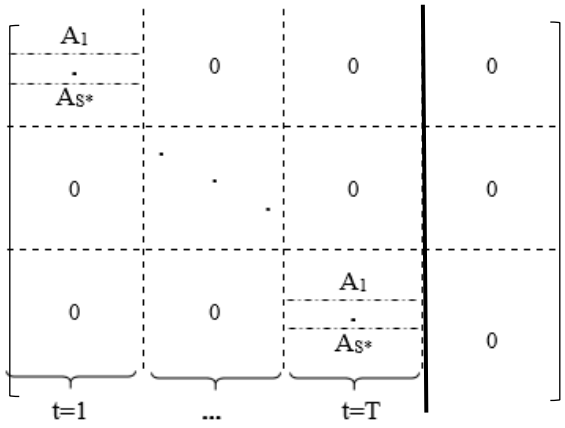
***Stope adjacency***

Based on equation (3.17) two adjacent stopes must not be mined at the same period, and it is called stope adjacency constraint. In order to create this constraint, an adjacency matrix has been created, and it includes all the stope adjacency. In fact, the adjacency matrix makes the adjacency set ($A_{s*}$). Adjacency constraint has $Na\times T$ rows.

Equation (3.46) demonstrates the structure of this constraint. The part related to decision variable $\boldsymbol{X}$ in the coefficient matrix is a $(Na\times T)\times(Ns*\times T)$ matrix and $\boldsymbol{Y}$ part of the coefficient matrix is a $(Na\times T)\times(Nl*\times T)$ matrix. $\left[ub_a\right]$ is a $(Na\times T)\times 1$ matrix and all the elements of the matrix are equal to 1. Equation (3.47) indicates the structure of the area related to the $\boldsymbol{X_a}$ and $\boldsymbol{Y_a}$. In this matrix, the first cell indicates all adjacency of stopes in the first period, and the last cell shows all adjacency of stopes in period *T*. A1 at first cell shows all adjacency of the first stope in the first period. A1 is a vector contains rows, which each row shows one of the adjacencies of the first stope. Also, As* is a vector contains rows, which each row shows one of the adjacencies of the last stope. In fact, each row of A1 and As* has two elements of 1. All elements of $\boldsymbol{Y_a}$ side are zero.

$$\left[X_a \;\; Y_a\right] \le \left[ub_a\right] \tag{3.46}$$

It should be noted that, for saving the memory space and decrease the size of the coefficient matrix of adjacency constraint, MATLAB's sparse function is used to squeezing out any zero elements.




(3.47)

***Connection between mining the stopes and activation of levels***

Based on equation (3.18) the stopes in a level should be mined only while the level is activated. This constraint has $(Nl^* \times T) \times 1$ rows. This equation can be written as equation (3.48).

$$\sum_{s^* \in B_{l^*}^{s^*}} x_{s^*}^t - N_l^{s^*} \times y_{l^*}^t \le 0, \qquad \forall t \in \{1,...,T\}, l^* \in \{1,...,L^*\} \tag{3.48}$$

Equation (3.49) shows the structure of this constraint. The part related to decision variable $\boldsymbol{X}$ in the coefficient matrix is a $(Nl^* \times T) \times (Ns^* \times T)$ matrix and $\boldsymbol{Y}$ part of the coefficient matrix is a $(Nl^* \times T) \times (Nl^* \times T)$ matrix. $\left[ub_{con}\right]$ is $(Nl^* \times T) \times 1$ matrix and all the elements are equal to zero. Equation (3.50) indicates the structure of the area related to the $\boldsymbol{X}_{con}$ and $\boldsymbol{Y}_{con}$. In this matrix, the left side belongs to decision variable $\boldsymbol{X}$, and the right side is for decision variable $\boldsymbol{Y}$. The first cell of the left side indicates set of stopes in each level ($B_{l^*}^{s^*}$) in the first period and the last cell of the left side indicates set of stopes in each level in period $T$. $B_1^{s^*}$ shows set of stopes in level one and $B_{Nl^*}^{s^*}$ shows set of stopes in the last level. The first cell of the right side shows the number of stopes in first selected level ($N_1^{s^*}$) to the last level ($N_{l^*}^{s^*}$) in the first period, and the last cell of the right

side indicates the number of stopes in first selected level ( $N_1^{s^*}$ ) to the last level ( $N_{l^*}^{s^*}$ ) in period *T*. $0,....,-N_1^{s^*}\times 1$ shows the first level is active and $0,....,-N_{l^*}^{s^*}\times 1$ shows the last level is active.

$$\left[X_{con} \quad Y_{con}\right] \le \left[ub_{con}\right] \tag{3.49}$$

$$\left[\begin{array}{cccccc|c} B_1^{s^*} & & & & & & \\ \cdot & 0 & 0 & -N_1^{s^*}\times 1,0,\ldots & 0 & 0 & \\ B_{Nl^*}^{s^*} & & & \cdot & & & \\ & & & 0,....,-N_{l^*}^{s^*}\times 1 & & & \\ \hline \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \\ \hline & & B_1^{s^*} & & & -N_1^{s^*}\times 1,0,\ldots & \\ 0 & 0 & \cdot & 0 & 0 & \cdot & \\ & & B_{Nl^*}^{s^*} & & & 0,....,-N_{l^*}^{s^*}\times 1 & \end{array}\right] \tag{3.50}$$

***Concurrent active levels***

Based on equation (3.19) number of the active level at the same time has the limitation. This constraint has *T* rows. Equation (3.51) shows the structure of this constraint. The part related to decision variable $\boldsymbol{X}$ in the coefficient matrix is a $T\times(Ns*\times T)$ matrix and $\boldsymbol{Y}$ part of coefficient matrix is a $T\times(Nl*\times T)$ matrix. $\left[ub_{concur}\right]$ is $T\times 1$matrix and the elements are equal to the maximum number of concurrent active levels (*Mcl*) in each period. Equation (3.52) indicates the structure of the area related to the $\boldsymbol{X}_{concur}$ and $\boldsymbol{Y}_{concur}$. In this matrix, the left side related to $\boldsymbol{X}_{concur}$ and the right side is related to $\boldsymbol{Y}_{concur}$. *C* is a $1\times Nl*$row vector contains all elements equal to 1. All the elements of the left side are zero.

$$\left[X_{concur} \quad Y_{concur}\right] \le \left[ub_{concur}\right] \tag{3.51}$$

$$\left[\begin{array}{ccc|cccc} 0 & . & 0 & C & 0 & . & 0 \\ . & . & . & 0 & C & . & 0 \\ . & . & . & . & . & . & . \\ . & . & . & . & . & . & . \\ 0 & . & 0 & 0 & 0 & . & C \end{array}\right] \tag{3.52}$$

***Delay between activation of levels***

Based on equation (3.20), between periods of activation a level and the next level should be a gap

($D$). This constraint can be written as equation (3.53).

$$y_{l^*}^{t} - \sum_{t'=1}^{t-D} y_{l^*+1}^{t'} \le 0, \qquad \forall t \in \{1,...,T\}, l^* \in \{1,...,L^*\} \tag{3.53}$$

This constraint has $Nl^* \times T$ rows. Equation (3.54) shows the structure of this constraint. The part related to decision variable $\boldsymbol{X}$ in the coefficient matrix is a $(Nl^* \times T) \times (Ns^* \times T)$ matrix and $\boldsymbol{Y}$ part of the coefficient matrix is a $(Nl^* \times T) \times (Nl^* \times T)$ matrix. $\left[ub_d\right]$ is $1 \times (Nl^* \times T)$ matrix and the elements are equal to zero. Equation (3.55) indicates the way $\boldsymbol{X_d}$ side and $\boldsymbol{Y_d}$ side is defined in the coefficient matrix. In this matrix, the left side related to $\boldsymbol{X_d}$ and the right side is related to $\boldsymbol{Y_d}$. All the elements on the left side are zero. D is the delay between activation of the levels. Each section in this matrix belongs to a period that is a $Nl^* \times Nl^*$ square vector. In D1 vector, main diagonal elements except the first active level are equal to 1, and all other elements are zero. In D2 vector, elements with row $l^*$ and column ($l^*$+1) are equal to -1 and all other elements are zero.

$$\left[X_d \;\; Y_d\right] \le \left[ub_d\right] \tag{3.54}$$

$$\begin{array}{l} D \\ \\ \\ \\ \\ \\ \\ \end{array} \left[\begin{array}{ccc|cccccc} 0 & . & 0 & D1 & 0 & . & 0 & . & 0 \\ . & . & . & 0 & D1 & . & 0 & . & . \\ . & . & . & D2 & 0 & . & 0 & . & . \\ . & . & . & D2 & D2 & . & 0 & . & . \\ . & . & . & . & . & . & . & . & . \\ 0 & . & 0 & D2 & D2 & . & D2 & 0 & D1 \end{array}\right] \begin{array}{l} \\ \\ \\ \\ \\ \\ \\ \end{array} \tag{3.55}$$

$$\underbrace{\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad}_{D}$$

### 3.4.2.4. Production Scheduling Optimization Based on Levels (SOL)

- **The BIP Objective Function**

The objective function of the finding optimum production scheduling problem as stated by equations (3.21) is to maximize NPV of mining stopes in each level separately. Table 3.10 shows the size and structure of the coefficient vector of the objective function in SOL method. The second

column of the table shows the size of the matrix; where, $N_{l*}^{s*}$ is the number of the selected stop in level $l^*$. In the third column of the table, $DEV_{l*}^{s*}$ is a $1\times(N_{l*}^{s*}\times T_{l*})$ vector that is the discounted economic value of selected stopes.

Table 3.10. Size and structure of the coefficient vector of objective function

| Method | Size | Structure |
|---|---|---|
| SOL | $1\times(N_{l*}^{s*}\times T_{l*})$ | $[DEV_{l*}^{s*}]$ |

Table 3.11 indicates the size of the vector for decision variables. $\boldsymbol{X}$ is a $(N_{l*}^{s*}\times T_{l*})\times 1$ vector holding binary decision variables controlling the mining of selected stopes in the selected levels $x_{s*,l*}^{t_{l*}}\in\{0,1\}$.

Table 3.11. Size and structure of the decision variables vector

| Method | Size | Structure |
|---|---|---|
| SOL | $(N_{l*}^{s*}\times T_{l*})\times 1$ | $[X]$ |

- **The Constraints of the BIP Models**

The constraints of the BIP model for production scheduling in SOL method are presented by equations (3.22) to (3.25). Equation (3.56) indicates the upper bound part of the constraint. The first $Tl*$ rows of mining capacity constraint belong to this part. Equation (3.57) demonstrates the lower bound part of the constraint. The second $Tl*$ rows of mining capacity constraint belong to this part. This constraint should be applied for each selected level separately.

Table 3.12 states the number of rows for each constraint. The number of columns in all constraint is $(N_{l*}^{s*}\times T_{l*})$. $N_{l*}^{a}$ is the total number of stopes adjacency which contains all possible adjacency for all stopes in level $l^*$. Figure 3.21 demonstrates the structure of constraints coefficient matrix and order of constraints in the matrix in SOL method.

In the following section, the structure of each constraint will be described in detail.

***Mining capacity***

Based on equation (3.22) minimum and the maximum of mining capacities in each level are noted in the formulation as mining capacity constraint. This constraint has $2T_{l*}$ rows. Equation (3.22) has two parts as following equations:

$$\sum_{s_{l*}^{*}=1}^{S_{l*}^{*}} Ton_{l*}^{s*} \times x_{s*,l*}^{t_{l*}} \le Cu_{l*}^{t_{l*}}, \qquad \forall t_{l*} \in \{1,...,T_{l*}\} \tag{3.56}$$

$$Cl_{l*}^{t_{l*}} - \sum_{s_{l*}^{*}=1}^{S_{l*}^{*}} Ton_{l*}^{s*} \times x_{s*,l*}^{t_{l*}} \le 0, \qquad \forall t_{l*} \in \{1,...,T_{l*}\} \tag{3.57}$$

Equation (3.56) indicates the upper bound part of the constraint. The first $T_{l*}$ rows of mining capacity constraint belong to this part. Equation (3.57) demonstrates the lower bound part of the constraint. The second $T_{l*}$ rows of mining capacity constraint belong to this part. This constraint should be applied for each selected level separately.

Table 3.12. Number of rows in constraint' coefficient matrix in SOL method

| Constraints | Number of rows |
|---|---|
| Mining capacity | $2T_{l*}$ |
| Grade blending | $2T_{l*}$ |
| Only one-time mining | $N_{l*}^{s*}$ |
| Adjacency | $N_{l*}^{a} \times T_{l*}$ |

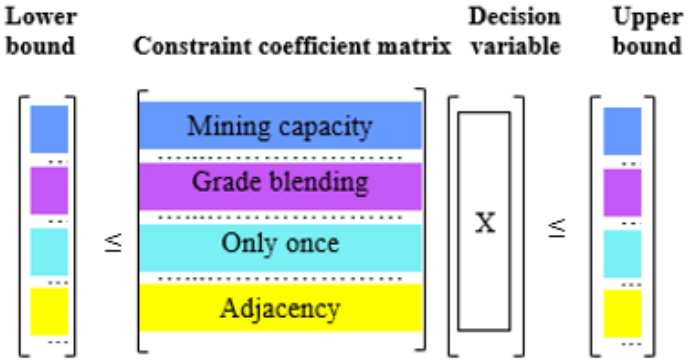



Figure 3.21.Order of constraints in the constraint coefficient matrix in SOL method

Equation (3.58) shows the structure of this constraint. The part related to decision variable $\boldsymbol{X}$ in the coefficient matrix is a $2T_{l*} \times (N_{l*}^{s*} \times T_{l*})$ matrix. $\left[lb_{C,l*}\right]$ and $\left[ub_{C,l*}\right]$ are $2T_{l*} \times 1$ matrices in which all the elements are equal to zero and should be defined for each level separately. The structure of

the area related to the $\boldsymbol{X}_{C,l^*}$ follows the same pattern and same equation as SOT method. However, in SOL method, the mentioned matrix in equation (3.39) does not need the right side anymore.

$$\left[lb_{C,l^*}\right] \le \left[X_{C,l^*}\right] \le \left[ub_{C,l^*}\right] \tag{3.58}$$

***Grade blending***

Based on equation (3.23) minimum and the maximum of required average grade are considered in the formulation as grade blending constraint. This constraint has $2T_{l^*}$ rows and it should be used for all of the selected levels. This equation has two parts as following:

$$\sum_{s_{l^*}^*=1}^{S_{l^*}^*} Ton_{l^*}^{s^*} \times x_{s^*,1^*}^{t_{l^*}} \times (G_{l^*}^{s^*} - Gu_{l^*}^{t_{l^*}}) \times R_{l^*}^{t_{l^*}} \le 0, \qquad \forall t_{l^*} \in \{1,...,T_{l^*}\} \tag{3.59}$$

$$\sum_{s_{l^*}^*=1}^{S_{l^*}^*} -Ton_{l^*}^{s^*} \times x_{s^*,1^*}^{t_{l^*}} \times (G_{l^*}^{s^*} - Gl_{l^*}^{t_{l^*}}) \times R_{l^*}^{t_{l^*}} \le 0, \qquad \forall t_{l^*} \in \{1,...,T_{l^*}\} \tag{3.60}$$

Equation (3.59) indicates the upper bound part of the constraint. The first $Tl^*$ rows of grade blending constraint belong to this part. Equation (3.60) demonstrates the lower bound part of the constraint. The second $Tl^*$ rows of grade blending constraint matrix are for this part.

Equation (3.61) shows the structure of this constraint. The part related to decision variable $\boldsymbol{X}$ in the coefficient matrix is a $2T_{l^*} \times (N_{l^*}^{s^*} \times T_{l^*})$ matrix. $\left[lb_{g,l^*}\right]$ and $\left[ub_{g,l^*}\right]$ are $2T_{l^*} \times 1$ matrices in which all the elements are equal to zero and should be determined for each level separately. The structure of the area related to the $\boldsymbol{X}_{\boldsymbol{g},\boldsymbol{l}^*}$ follows the same pattern and the same equation as SOT method. However, the matrix in equation (3.43) does not need the right side in SOL method.

$$\left[lb_{g,l^*}\right] \le \left[X_{g,l^*}\right] \le \left[ub_{g,l^*}\right] \tag{3.61}$$

***Only one-time mining***

Based on equation (3.24) each stope in each level must be mined precisely once in all periods. Since this constraint is a linear equality constraint, is formulated based on equation (3.28). This constraint has $N_{l^*}^{s^*}$ rows. Equation (3.62) indicates the structure of this constraint. The part related

to decision variable $\boldsymbol{X}$ in the coefficient matrix is a $N_{l*}^{s*} \times (N_{l*}^{s*} \times T_{l*})$ matrix. Also, $\left[beq_{o,l*}\right]$ is a $N_{l*}^{s*} \times 1$ matrix in which all the elements are equal to 1 and should be determined for each level separately. The structure of the area related to the $\boldsymbol{X}_{o,l*}$ follows the same pattern and the same equation as SOT method although the matrix in equation (3.45) does not have the right side in SOL method.

$$\left[X_{o,l*}\right] = \left[beq_{o,l*}\right] \tag{3.62}$$

***Stope adjacency***

Based on equation (3.25) two adjacent stopes in a level must not be mined at the same period, and it is called adjacency constraint for each level. In order to create this constraint, an adjacency set has been created and it includes all the stope adjacency. Adjacency constraint has $N_{l*}^{a} \times T_{l*}$ rows, where $N_{l*}^{a}$ is different for each level.

Equation (3.63) demonstrates the structure of this constraint. The part related to decision variable $\boldsymbol{X}$ in the coefficient matrix is a $(N_{a-l*} \times T_{l*}) \times (N_{l*}^{s*} \times T_{l*})$ matrix. $\left[ub_{a,l*}\right]$ is a $(N_{a-l*} \times T_{l*}) \times 1$ matrix in which all the elements are equal to 1 and should be determined for each level separately. The structure of the area related to the $\boldsymbol{X}_{a,l*}$ follows the same pattern and the same equation as SOT method; nonetheless, the matrix in equation (3.47) does not need the right side for SOL method.

$$\left[X_{a,l*}\right] \leq \left[ub_{a,l*}\right] \tag{3.63}$$

## 3.5. Summary and Conclusion

In summary, two methods of stopes layout optimization, LOT method and LOL method are explained. Also, two methods to production schedule optimization are presented, SOT method and SOL method.

In LOT method, after calculating the economic value of blocks, all possible stopes are created. Among all those stope possibilities, possibilities with positive EV are determined. Then, the stopes overlap between those positive stopes are defined. Then, the BIP formulation framework is developed with the objective function of maximizing EV of stopes and the stopes overlap as the

primary constraint. In LOL method, at the first step, all possible levels are defined. Then, the positive stope possibilities and the overlaps between those are determined for each level. Afterwards, the BIP formulation is run at each level to find the stopes with the highest value in each level. According to EV of each level, another BIP formulation is run to find the best levels set, where the objective function is maximizing EV.

SOT method starts with the selected stopes by LOL method as the input. Then, BIP formulation is employed in which the objective function is to maximize NPV in the presence of some practical constraints. As a result, the sequence of mining the stopes is established. SOL method begins with the selected stopes and selected levels as the outputs of LOL method. Then, BIP formulation with the objective function of maximizing NPV and different practical constraints are applied to find the production scheduling in each level independently.

The numerical model of the BIP formulations are created in MATLAB (MathWorks Inc, 2017) and IBM ILOG CPLEX Optimization Studio 12.7.1 (IBM, 2017) is used to solve large-scale BIP problems. The structure of all the vectors and matrices related to objective functions and constraints are explained in this chapter.

# CHAPTER 4: VERIFICATION, IMPLEMENTATION, AND DISSCUSSION OF RESULTS

*Chapter 4 presents the implementation of the BIP model framework to find the optimum stopes layout and to determine the optimum production scheduling. Two presented algorithms for stopes layout optimization, LOT and LOL, are carried out and the results of two methods are compared. Besides, the algorithms of SOT and SOL methods are applied to generate optimal production schedule, and the outputs of these methods are differentiated.*

## 4.1. Introduction

In this chapter, the mathematical formulations will be implemented on a drillhole data set in order to demonstrate how the presented methodologies work. First, the information of data set will be explained. Then, presented methodologies will be applied to the data set to do the stopes layout optimization by LOT and LOL methods. Afterwards, the presented methodologies of SOT and SOL will be examined on data set to generate the production schedule then, some investigations and comparisons will be studied to assess the methods.

## 4.2. Case study

The algorithms were tested on a block model that was created from the drillhole data. This block model belongs to a silver (AG) deposit. Figure 4.1shows the block model. Also, Table 4.1 indicates the block model information.

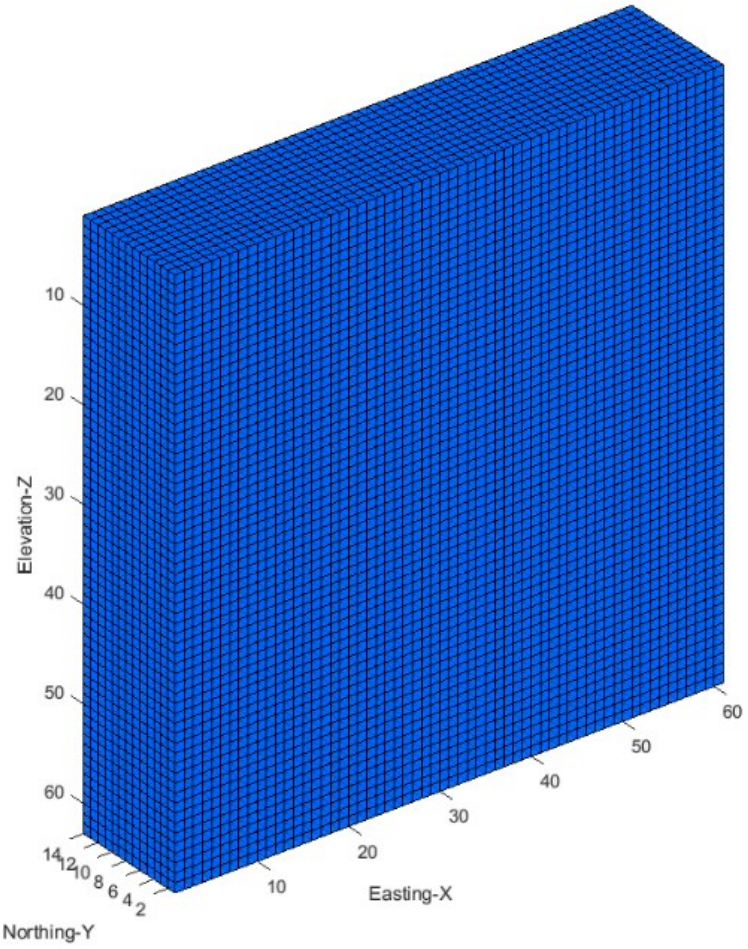



Figure 4.1. The block model

Table 4.1. Block model information

| | |
|---|---|
| Number of blocks | 48,362 |
| Blocks size ($m^3$) | 10×10×10 |
| Blocks tonnage (tonne) | 2700 |
| Blocks grade (g/tonne) | 0 - 1580 |
| X Coordinate (X index) | 1-60 |
| Y Coordinate (Y index) | 1-13 |
| Z Coordinate (Z index) | 1-62 |

Figure 4.2 demonstrates the situation of the ore-body of the block model with total tonnage of 37.9 (Mt) and Figure 4.3 shows the ore-body based on the different range of grade of silver. Also, Figure 4.4 indicates the grade distribution of ore-body.

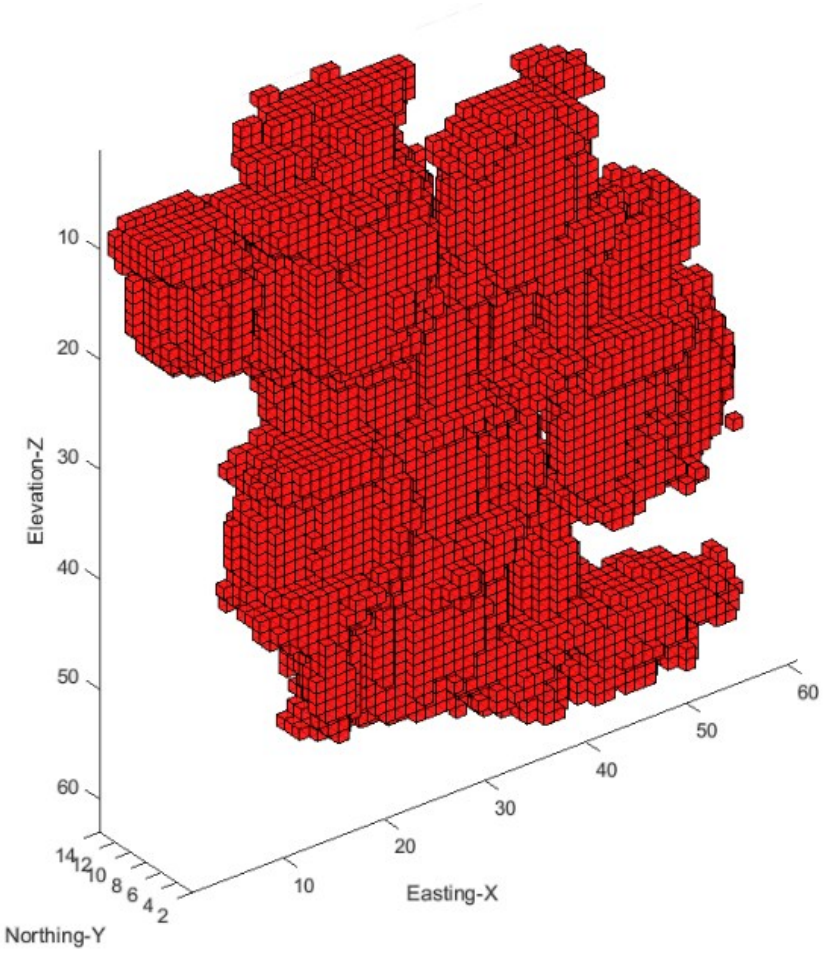



Figure 4.2. Ore in the block model

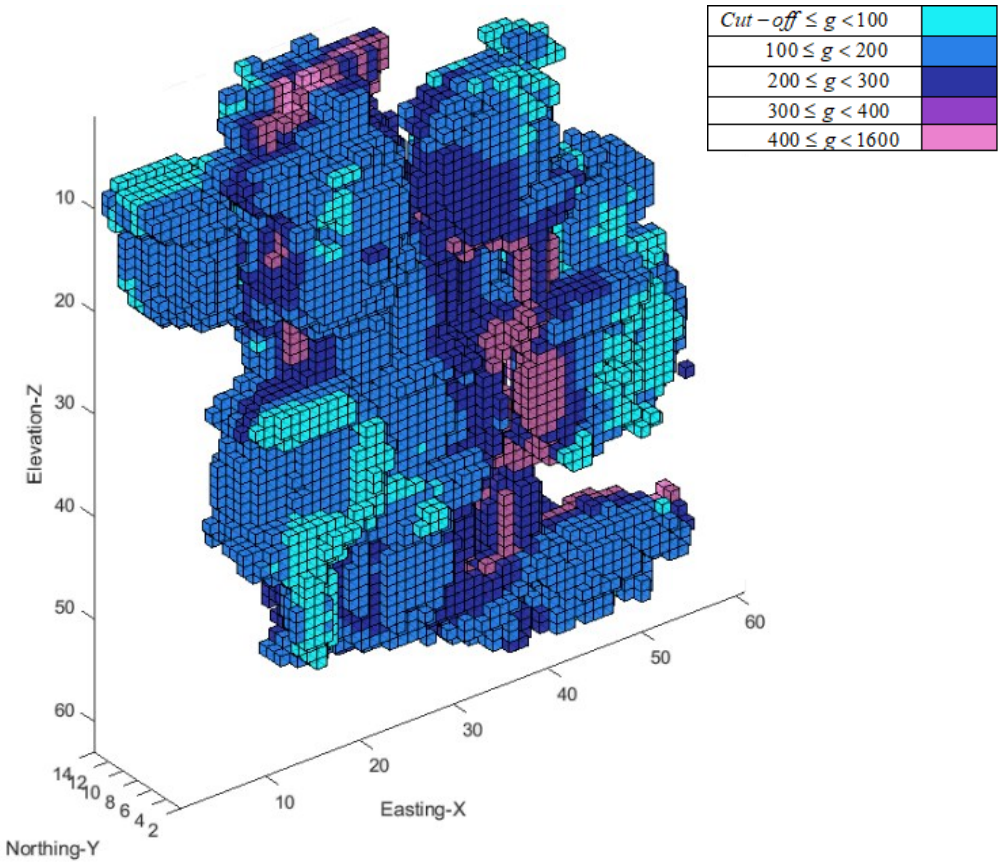



Figure 4.3. Grade of AG (g/t) in the ore blocks

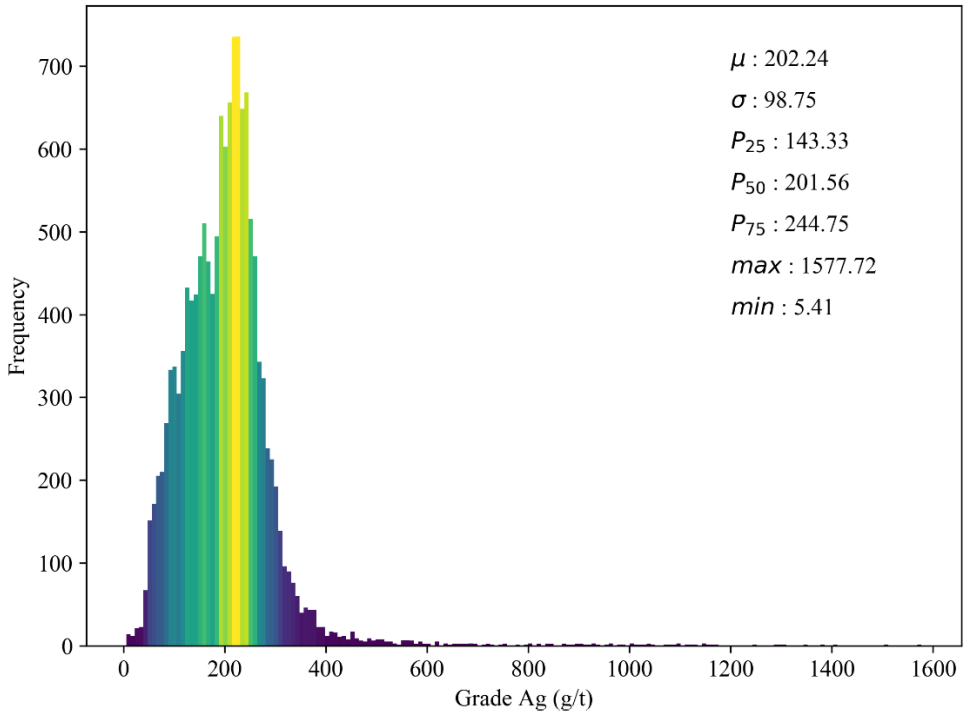



Figure 4.4. Grade distribution of the ore-body

### 4.3. Layout Optimization Based on Total Blocks (LOT)

The economic value (EV) of the blockes was calculated based on the parameters listed in Table 4.2. The defined Ag price in this table is based on the price in last two years (SILVERPRICE). After calculating the cut-off grade, waste or ore blocks were determined and the BEV was calculated using equations (3.1) and (3.2). The calculated cut-off grade for this block model was 66.7 (g/t), and the block values were between $ -0.64M and $ +2,20M.

Table 4.2. Economic parameters

| | |
|---|---|
| Metal price ($/g) | 0.6 |
| Cost of mining ($/t) | 24 |
| Cost of processing ($/t) | 12 |
| Recovery | 90% |

According to (Hartman, 1992; Haycocks et al., 1992), stope width at least must be 6 meters, and the range for length and height of the stopes is between 45 and 120 meters. In order to have a practical condition, the stope dimension 40×40×120 cubic meters (4×4×12 blocks) was chosen in this study. However, there is the possibility of changing this dimension and scanning the impacts of changing. Creating the stopes and separating the positive ones were the following steps in this section. In LOT method, 29,070 initial stopes were generated which 13,403 of those stopes had positive EV.

Then a 13,403×13,403 matrix was created as the overlap matrix. This matrix contains the element of 0 for positive stopes without the overlap and the element of 1 for positive stopes with overlap. By using computer with processor: Intel(R) Core(TM) i5-4460 CPU @ 3.20GHz and RAM: 6:00 GB, the solution time for recognizing the overlap matrix was 18 (hr):59(min):06(sec).

By running the LOT algorithm and removing all overlaps, the combination of 85 stopes was discovered. In fact, this combination of stopes was the optimum stopes layout. The EV of stopes in this layout was $ 2,341.1M. Also, the total solution time was 62(hr):02(min):07(sec).

Figure 4.5 illustrates the stopes in the result of the LOT method. The selected stopes in the optimum stopes layout are demonstrated with the different colors.

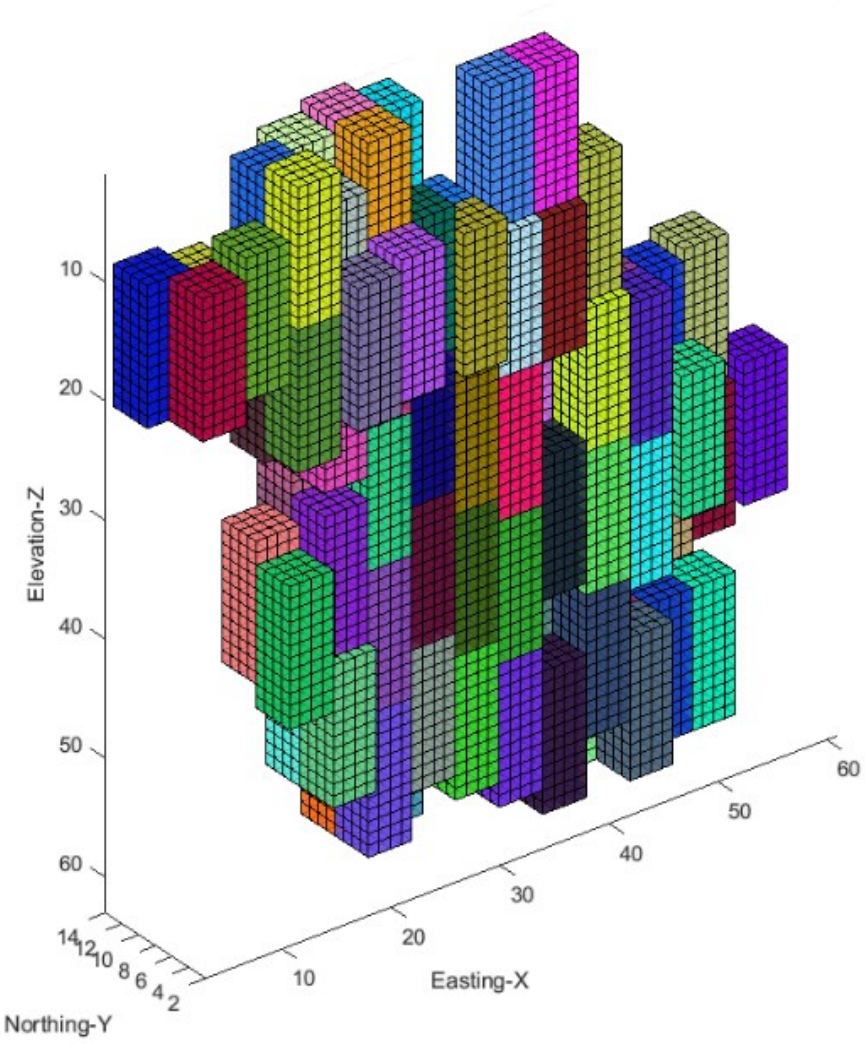



Figure 4.5. Optimum stopes layout based on the LOT method

## 4.4. Layout Optimization Based on Levels (LOL)

Calculation of block economic model in LOL method is the same as LOT method. In LOL method, the primary block model was divided into a number of possible levels that was the base elevation for creating the stope. In the case study, the number of blocks in direction Z is 62 (see Figure 4.1) and dimension of stope in the Z direction is 12, so based on the equation (3.3) number of possible levels is equal to 51.

For all 51 possible levels, the process of creating stopes was applied, and initial and positive stopes were defined in each level. For instance, in the first level, 597 initial stopes were determined which 165 of those have positive EV.

In total, 30, 447 initial stopes and 13,403 positive stopes were generated in LOL method. Then, the overlap matrices were created in all possible levels. The dimension of overlap matrices varies based on the number of positive stopes in each possible levels.

After running the algorithm and considering stopes overlaps, the best stopes combinations for all possible levels were determined. For instance, in the first possible level, 11 stopes with the value of $ 219.7M were found as the best stope combination with the highest EV in this possible level.

By comparing the EV of levels and considering the constraints, the best set of possible levels among all possible sets was discovered. A set of possible levels 2 (Z=13 of block model), 14 (Z=25 of block model), 26 (Z=37 of block model), 38 (Z=49 of block model) and 50 (Z=61 of block model) was determined as the optimum set with the highest EV. Table 4.3 demonstrates a number of initial stopes, positive stopes and the selected stopes in each selected level. In addition, the EV of the selected levels are indicated in this table. Generally, 5 levels including 93 stopes and total value of $ 2,252.2M were the result of LOL method. The LOL method reached the answer in 00:08(min):54(sec). Figure 4.6 indicates the stopes layout based the result of the LOL method. The selected stopes in the optimum stopes layout in different levels are demonstrated with the different colors.

Table 4.3. Information of selected levels

| Level ID | Possible level ID | Initial stopes (Num) | Positive stopes (Num) | Selected stopes (Num) | Value (M$) |
|---|---|---|---|---|---|
| Level 1 | 2 | 597 | 202 | 12 | 263.6 |
| Level 2 | 14 | 597 | 283 | 20 | 434.8 |
| Level 3 | 26 | 597 | 265 | 21 | 599.6 |
| Level 4 | 38 | 597 | 294 | 22 | 630.1 |
| Level 5 | 50 | 597 | 232 | 18 | 324.1 |

## 4.5. Comparison between LOT and LOL Methods

As Table 4.4 indicates number of selected stopes in the optimum layout defined by LOL method, is eight stopes more than LOT method and total mined tonnage in LOL method is 8.5% higher than LOT method. The achieved value by LOT method is 4% more than that value for LOL method. However, there is a world of difference between the running times the methods. As in LOT method all the stopes are evaluated at one set, some steps of the algorithm including assessing stopes overlaps, creating the overlap constraint and running optimization model are time-

consuming processes. Nevertheless, in LOL method the stopes divided to the smaller sets, levels, so the processing time drops significantly.

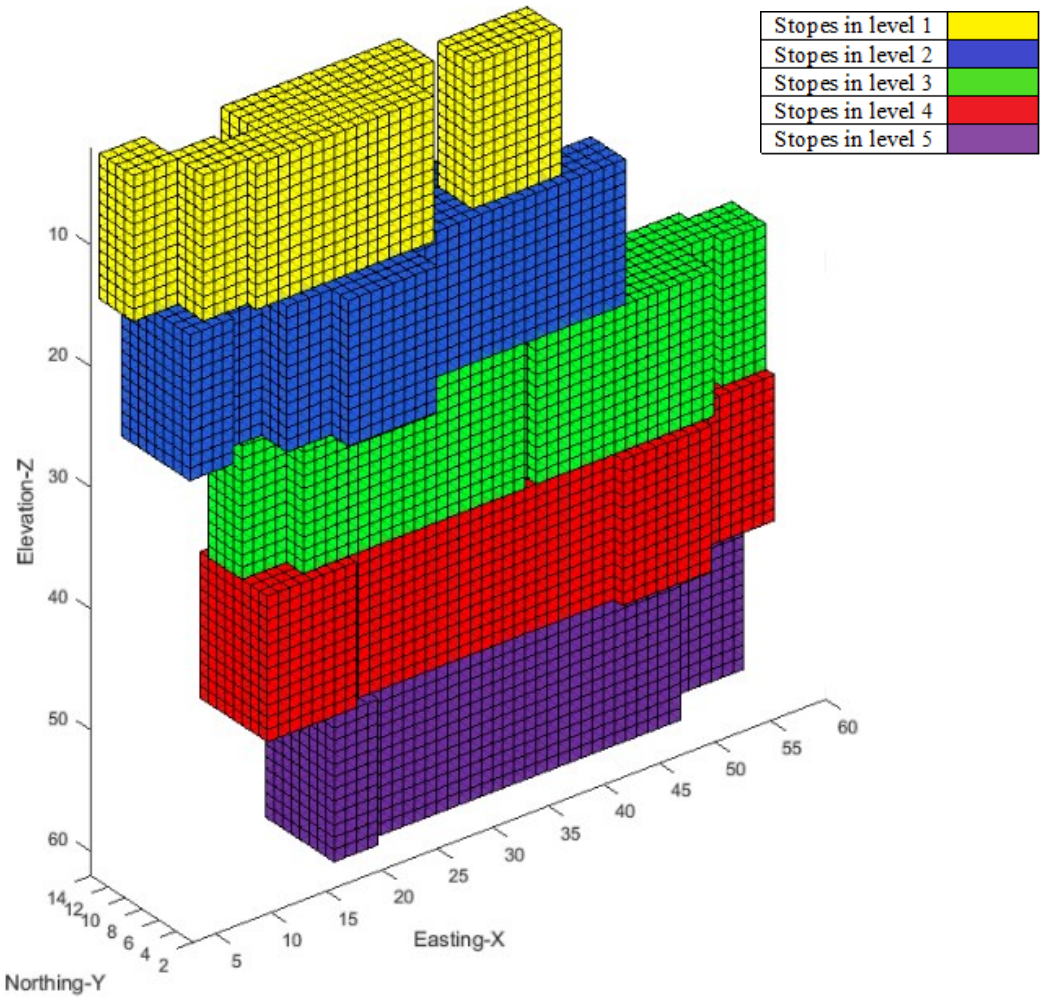



Figure 4.6. Optimum stopes layout based on the LOL method

Table 4.4. Results of LOT and LOL methods

| Method | Selected Stopes (Num) | Total tonnage (Mt) | Value (M$) | Running time (hr): (min) :(sec) |
|---|---|---|---|---|
| LOT | 85 | 44.1 | 2,341.1 | 62:02:07 |
| LOL | 93 | 48.2 | 2,252.2 | 00:08:54 |

As indicated in Figure 4.6, in LOL method, selected stopes in the optimum stopes layout were in the same levels. As a result, this method achieved to the practical solution. However, in LOT method (Figure 4.5), selected stopes in the optimum stopes layout were in different elevations so

it is not possible to design production levels to have access to all the stopes. Therefore, the achieved solution by LOT method was not practical.

After defining the optimum stopes layout by LOL method and discovering the best stopes to mine, finding the optimum mining sequence of those stopes during mine life by two methods SOT and SOL was employed.

### 4.6. Production Scheduling Optimization Based on Total Stopes (SOT)

The tonnage of selected stopes in the optimum layout was considered as the total tonnage of material needs to be mined. To apply the production scheduling defining a reasonable life of mine is the first step. The total tonnage of 93 stopes was 48.21Mton. According to equations (3.4) and (3.5), the life of mine was 22 years. Discount rate of 10% was used in this case study for production scheduling.

As it mentioned five levels selected as the output of LOL method. To have the practical situation in this study, mining between levels is in the order, one by one level from lowest level: Level 5 (possible level ID: 50) toward highest level: Level 1 (possible level ID: 2).

Additionally, two periods are defined as the delay between activation of subsequent levels (D). Moreover, the maximum number of active levels at a period (Mcl) is three. The maximum and minimum capacity of the mining operation in each period as the boundaries of mining capacity constraint and the maximum and minimum required average grade in each period as the boundaries of grade blending constraint are defined in this section. Table 4.5 provides the required parameters for production scheduling based on the SOT method.

Table 4.5. Production scheduling parameters in SOT method

| | |
|---|---|
| Life of mine (year) | 22 |
| Discount rate (%) | 10 |
| Minimum mining capacity (Mt) | 1.55 |
| Maximum mining capacity (Mt) | 3.1 |
| Minimum average grade (g/t) | 55 |
| Maximum average grade (g/t) | 220 |
| Delay between activation (period) | 2 |
| Maximum number of active levels at a period | 3 |
| Direction of mining the levels | Upward |

The adjacency matrix contained 20,020 adjacencies and was generated in 00:45(min):32(sec). This was the most time-consuming part of the production scheduling algorithm in SOT method. After generating all the constraints including mining capacity, grade blending, only one-time mining, stop adjacency, the connection between mining the stopes and activation of levels, concurrent active levels and delay between activation of the levels, the production schedule was generated.

The achieved NPV by SOT method was $ 1,288.1M. Running time for SOT method was 00:46(min):14(sec) and the optimality gap is zero percent. Figure 4.7 shows the achieved discounted economic value (DEV) of SOT method over life of the mine and cumulative DEV of mining the stopes. According to this figure, the overall pattern of DEV is descending during the mine life. The reason for less DEV in the first two periods in SOT graph comes back to the delay between activation of subsequent levels that forces the model to select stopes within the first level (Level 5).

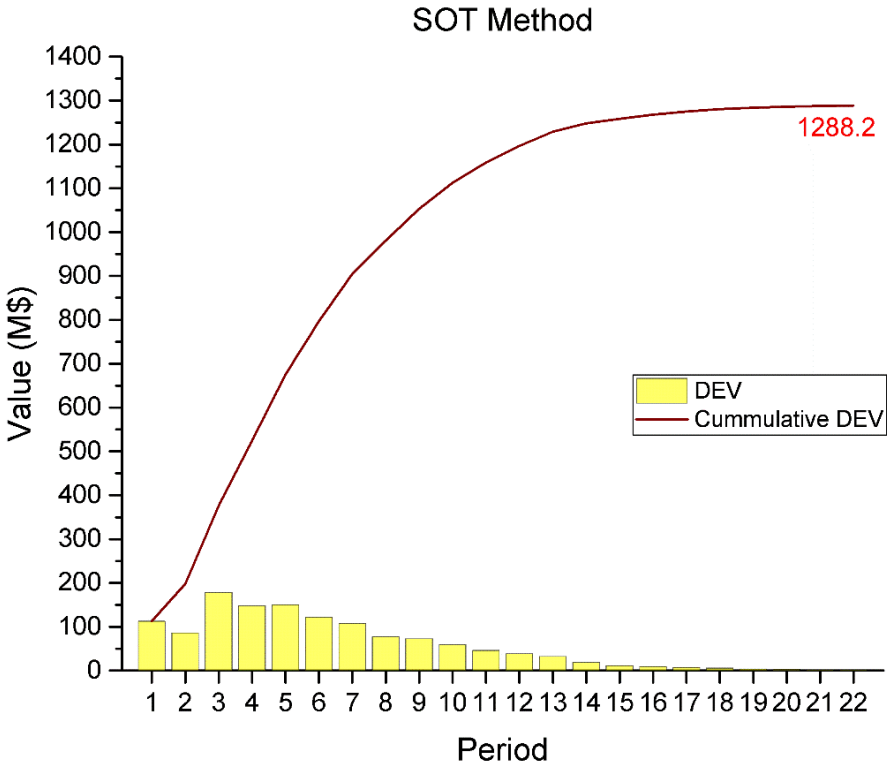



Figure 4.7. DEV in each period of each level and cumulative DEV in SOT method

Figure 4.8 illustrates the sequence of mining the stopes based on SOT method. Different colors show the different periods. According to this figure, each stope is selected only one time and not

two adjacent stopes are mined in the same period that show the only one-time mining and stopes adjacency constraints have been satisfied.

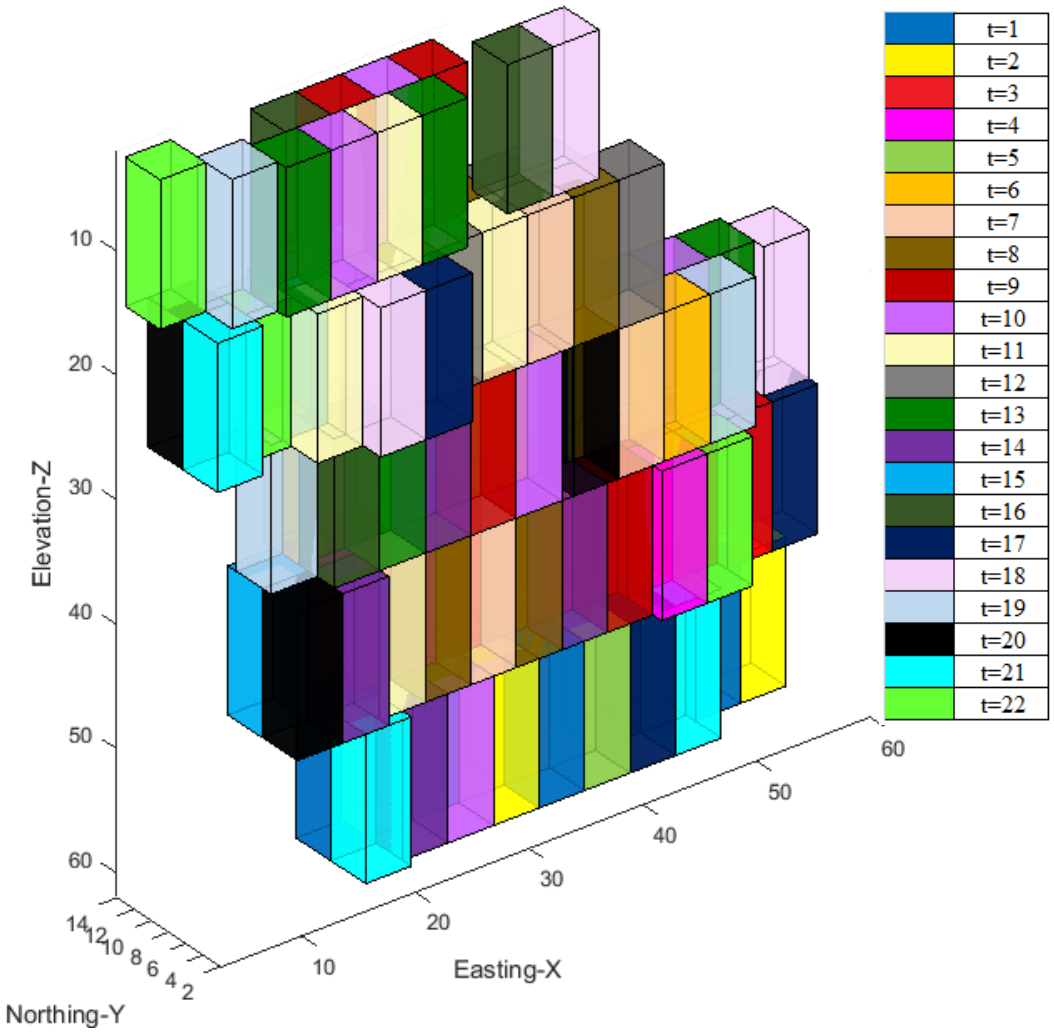



Figure 4.8. Optimum production scheduling based on the SOT method

Figure 4.9 shows the activation of each level and number of mined stopes in each period as the result of SOT method. This figure demonstrates the maximum number of active levels at each period is three and each level starts to be active two periods after activation of its previous level that shows meeting the concurrent active levels and delay between activation of the levels constraints. According to this graph, in period 1 and 2, only Level 5 is active. After two periods as the activation delay, Level 4 starts to be active in period 3. In all first 13 periods, 5 stopes are mined. In period 14, 4 stopes and in last 8 periods only 3 stopes are mined. Also, the slope of increasing the number of completed stopes almost stays the same between 1 stope in first and 93 stopes in last period.

Figure 4.10 indicates the production tonnage of mining stopes and average grade of mining stopes in each period in this method. As it can be seen in this figure, the mining production in each period

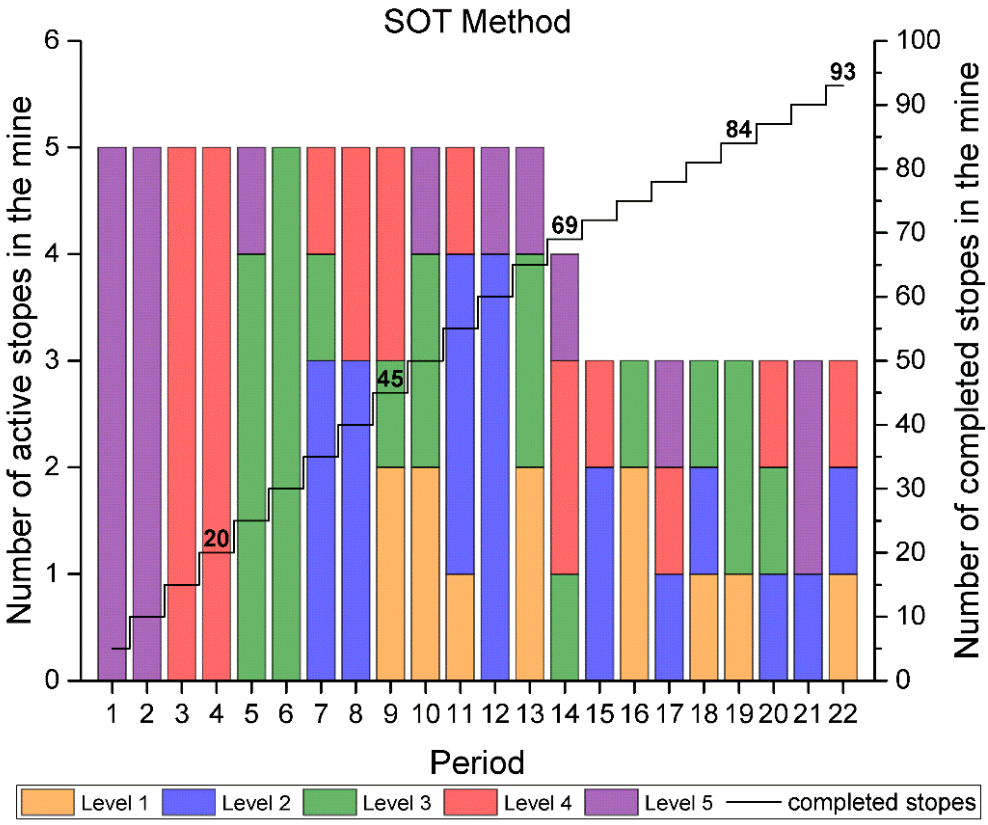



Figure 4.9. Activation of the levels and number of completed stopes in each period in SOT method

is between minimum and maximum miming capacities that shows the satisfaction of mining capacity constraint. For the first thirteen periods, the production is even (2.6 Mt). In period fourteenth, the production drops to 2.07 Mt. In addition, in the last eight periods, mining production remains even and equals the minimum mining capacity (1.55 Mt). Also, Figure 4.10 demonstrates the average grade of production in each period. As is shown the average grade production in each period is between minimum and maximum required grade. The Average grade drops from 137.2 (g/t) to 123.2(g/t) between first and second period, then the graph steeply raised between the second and third periods to reach to 216.1(g/t). The season for less average grade in first two periods is limitation in stopes selection due to the delay between activation of the levels constraint (only stopes at Level 5 can be selected). Production average grade decreases in period four and then grows to 219.6 (g/t) in period five. Also, after the period five, the slope of declining the production grade almost stayed the same.

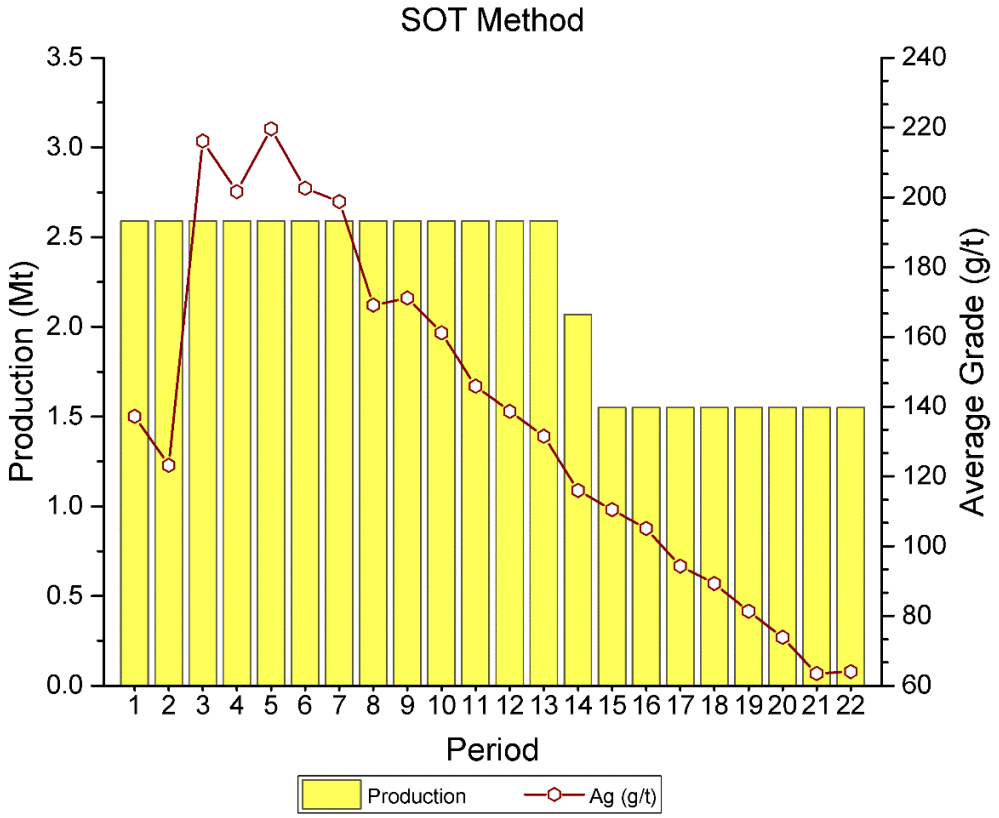



Figure 4.10. Production tonnage of mining stopes and average grade of mining stopes in each period in SOT method

## 4.7. Production Scheduling Optimization Based on Levels (SOL)

To be able to compare the result of SOT and SOL methods, the mining direction in SOL method was selected same as mining direction in SOT that was from the lowest level (Level 5: possible level ID: 50) toward the highest level (Level 1: possible level ID: 2). As a result, the first level to run the algorithm was Level 5. Summation of tonnage of stopes in each level was considered as the required mining tonnage in that level. 22 years was considered as life of the mine. By distributing the mine life to levels based on the total tonnage of stopes in each level, mining period for each level was determined.

Table 4.6 shows the number of stopes in each level, total tonnage of each level and the assigned period for each level. It should be noted that to calculate DEV of stopes in a level (equation (3.6)), $t$ starts after finishing the period of the previous level. For instance, according to Table 4.6, level 5 is active in the first four periods then level 4 will be active from period fifth to period ninth. As a result, to calculate DEV of stopes in level 4, $t$ will be 5, 6, 7, 8 and 9.

Table 4.6. Assigned period for each level

| Level ID | Stope (Num) | Average grade of level (g/t) | Tonnage (Mt) | Period (year) |
|---|---|---|---|---|
| Level 1 | 12 | 138.3 | 6.2 | 3 |
| Level 2 | 20 | 137.8 | 10.4 | 5 |
| Level 3 | 21 | 163 | 10.9 | 5 |
| Level 4 | 22 | 162.9 | 11.4 | 5 |
| Level 5 | 18 | 120.7 | 9.3 | 4 |
| Total | 93 | 146.2 | 48.2 | 22 |

To be able to compare the result of SOT method and SOL method, the maximum and minimum capacity of the mining operation in each period and maximum and minimum required average grade in each period were same. Table 4.7 provides the required parameters for production scheduling based on the SOL method.

Table 4.7. Production scheduling parameters in SOL method

| | |
|---|---|
| Life of mine (year) | 22 |
| Discount rate (%) | 10 |
| Minimum mining capacity (Mt) | 1.55 |
| Maximum mining capacity (Mt) | 3.1 |
| Minimum average grade (g/t) | 55 |
| Maximum average grade (g/t) | 220 |
| Direction of mining the levels | Upward |

Adjacent stopes in SOL method is assessed in 2D dimension because all stopes in a level have the same coordinate of Z. Same as SOT method, generating stopes adjacent matrices are the most time-consuming part of the algorithm. After generating all the constraints including mining capacity, grade blending, only one-time mining, stop adjacency, the production schedule was generated.

The total obtained NPV and the running time for the SOL method with the optimality gap of zero percent were $ 1,052.5 M and 00:13(min):34(second), respectively. Table 4.8 indicates the NPV achieved by applying the production scheduling in each level. According to Table 4.8, Level 4 has the highest NPV and Level 1 has the lowest NPV. Since Level 5 contains less number of stopes than Level 4, Level 4 reaches to the highest NPV. Also, Figure 4.11 shows the NPV in each period of each level and cumulative NPV of mining the stopes. This figure includes five parts that each of them shows the descending pattern between the first period and the last period of mining in a

level. Descending pattern of these parts confirm the fact that maximization model tries to select the stopes with the highest EV in the earlier periods to achieve the highest NPV.

Table 4.8. Maximum achieved value of each level in SOL method

| Level ID | NPV (M$) |
|---|---|
| Level 1 | 39.7 |
| Level 2 | 102.2 |
| Level 3 | 230.9 |
| Level 4 | 388.7 |
| Level 5 | 291.0 |
| NPV | 1,052.5 |

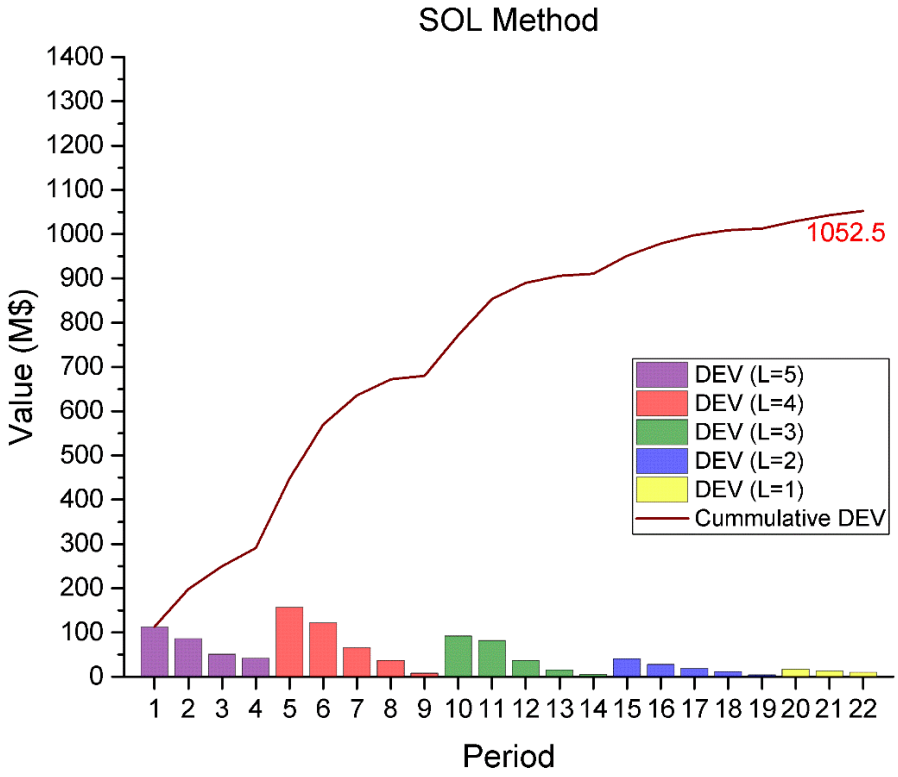



Figure 4.11. DEV in each period of each level and cumulative DEV over the periods in SOL method

Figure 4.12, Figure 4.13, Figure 4.14, Figure 4.15 and Figure 4.16 display the solution in levels 1, 2, 3, 4 and 5 respectively. Different colors show the different mining periods.

According to Figure 4.12, 5 stopes in the first period, 4 stopes in the second period, and 3 stopes in the third period of mining in Level 1 will be mined. Overall, 12 stopes in 3 periods are scheduled in this level.

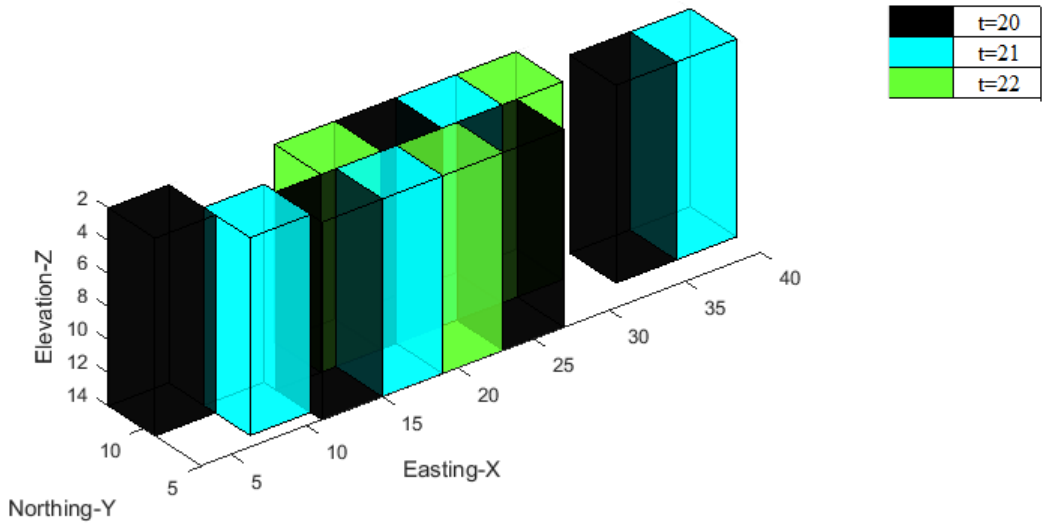



Figure 4.12. Optimum production scheduling in Level 1 based on the SOL method

Figure 4.13 shows the mining of 6, 5, 3, 3 and 3 stopes in the first, second, third, fourth and fifth period of mining in Level 2, respectively. Overall, 20 stopes are mined in 5 periods in this level.

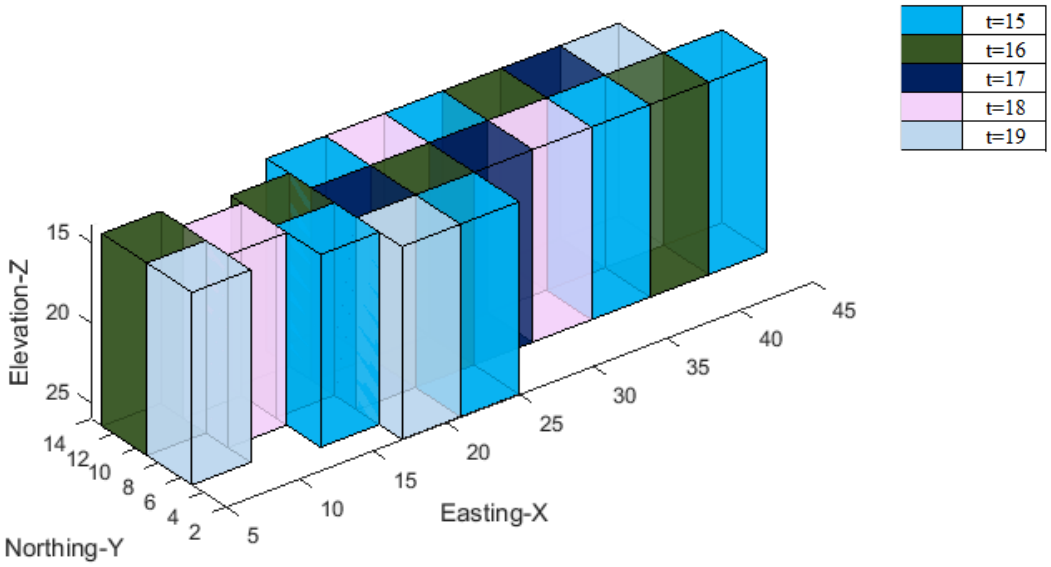



Figure 4.13. Optimum production scheduling in Level 2 based on the SOL method

Figure 4.14 indicates out of total 21 stopes in Level 3, 6 stopes in period 10, 5 stopes in period 11, 4 stopes in period 12, 3 stopes in period 13, and 3 stopes are mined in period 14.

Figure 4.15 shows that in Level 4, 6 stopes in period five, 5 stopes in period six, 5 stopes in period seven, 3 stopes in period eight, and 3 stopes in the period nine will be extracted. Overall, 22 stopes were scheduled to mine in 5 periods in this level.

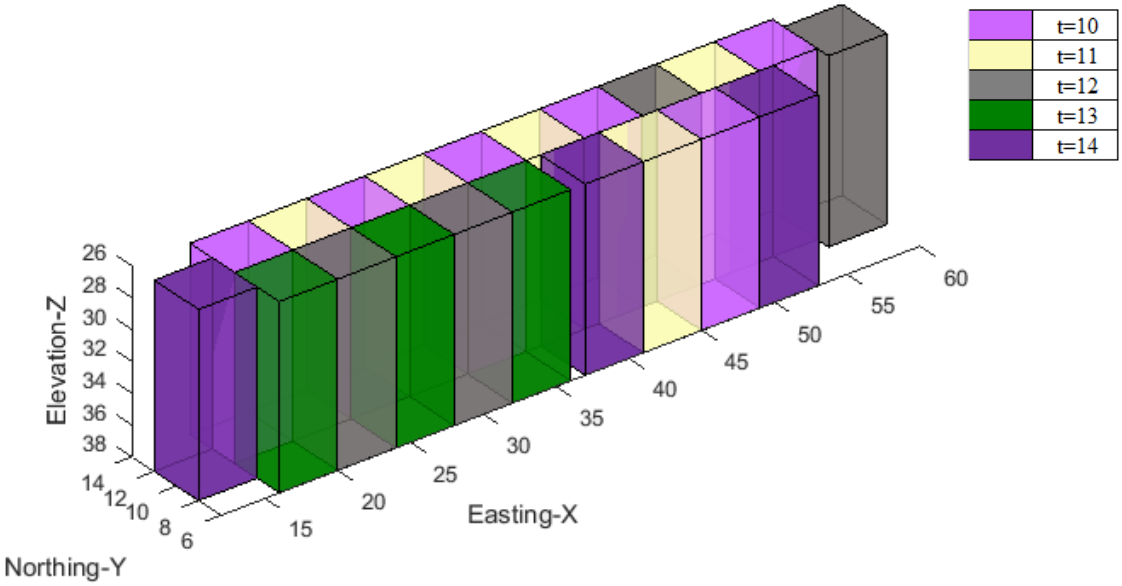



Figure 4.14. Optimum production scheduling in Level 3 based on the SOL method

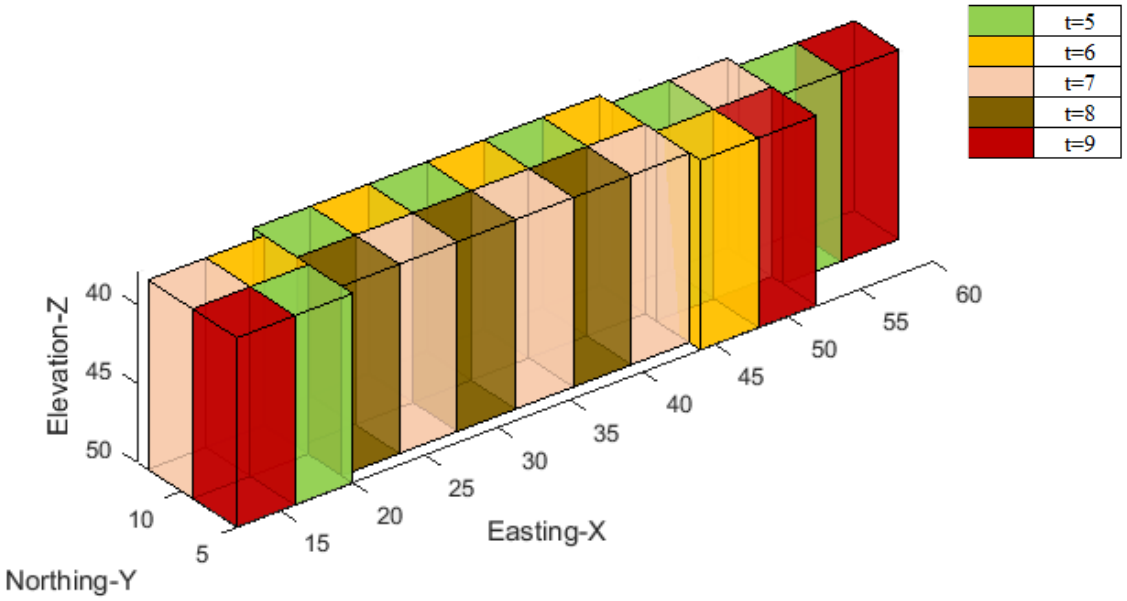



Figure 4.15. Optimum production scheduling in Level 4 based on the SOL method

Finally, Figure 4.16 demonstrates the mining of 18 stopes in four periods in Level 5. 5, 5, 4 and 4 stopes are mined in the first, second, third and fourth period in this level.

According to Figure 4.12, Figure 4.13, Figure 4.14, Figure 4.15 and Figure 4.16 in all levels each stope is selected only one time and not two adjacent stopes are mined in the same period that show the only one-time mining and stopes adjacency constraints have been satisfied.

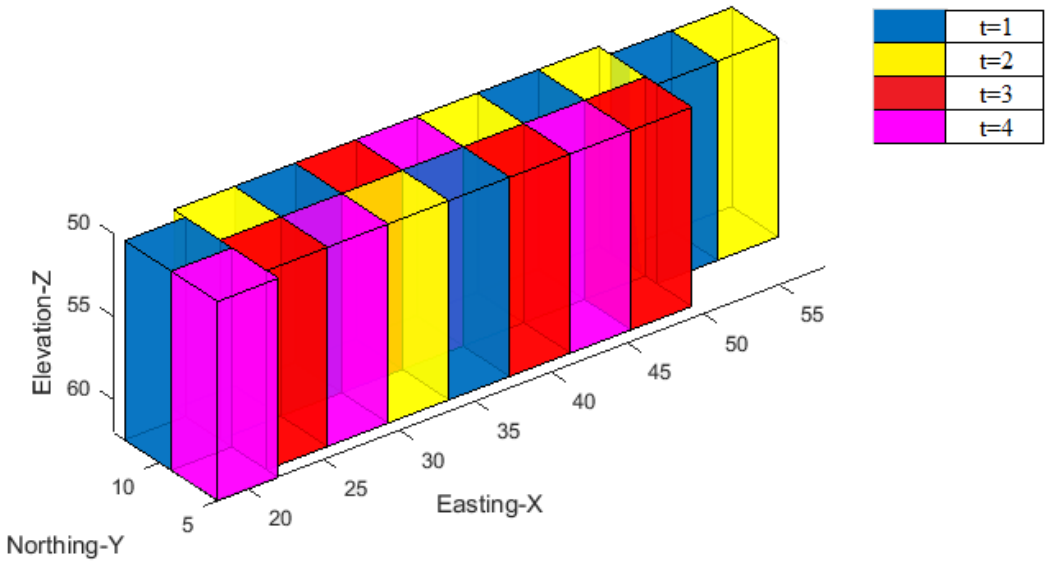



Figure 4.16. Optimum production scheduling in Level 5 based on the SOL method

Figure 4.17 shows the activation of levels and number of mined stopes in each period as the achieved solution by SOL method. According to this figure, number of mined stopes in each period changes between 3 and 5. The slope of increasing the number of completed stopes almost stays the same between 1 stope in period 1 and 93 stopes in period 22.

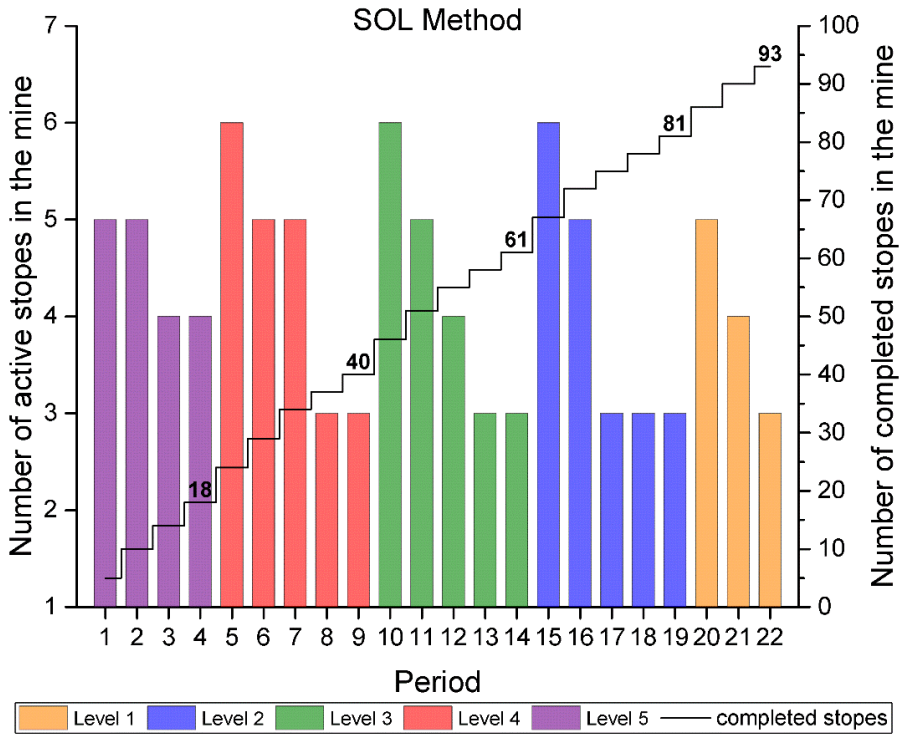



Figure 4.17. Activation of the levels and number of completed stopes in each period in SOL method

Figure 4.18 indicates the production tonnage of mining stopes and average grade of mining stopes in each period for each level in SOL method. This figure shows the satisfaction of mining capacity constraint since the mining production in each period of each level is between minimum and maximum miming capacities. According to Figure 4.18, mining production at Level 5 stabilizes in period one and two in 2.6Mt, and then falls to 2.07 Mt in periods three and four. In Level 4 it declines from 3.1Mt in the fifth period to 2.6Mt in the sixth period of mine life, then stabilizes in period seven, and eventually drops to minimum mining capacity in period eight and nine. In Level 3 it goes down from 3.1Mt to 2.07 Mt in the first three periods and remains constant at minimum mining capacity in the last two periods. In Level 2 it fells from 3.1 Mt to 2.6 Mt during the first two periods and it stays the same in the last three periods to reach minimum mining capacity. Finally, in Level 1 it decreases moderately from 2.6 Mt to minimum mining capacity during three periods.

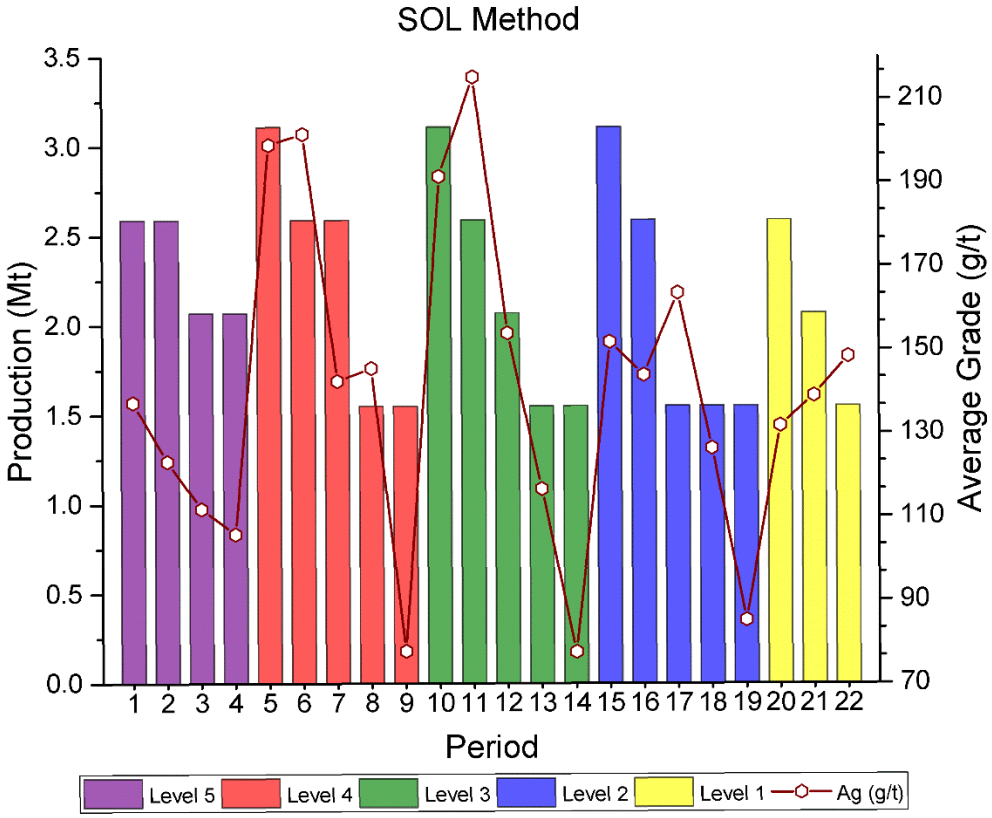



Figure 4.18. Production tonnage of mining stopes and average grade of mining stopes in each period in SOL method

In addition, Figure 4.18 demonstrates the average grade of production in each period for all levels. As is shown in this figure, the grade blending constraint is satisfied. Production average grade in

Level 5 it gently drops between 137.2(g/ton) in the first period and 111.8(g/ton) in four periods. In Level it has fluctuated descending pattern between 199(g/ton) in period five and 77.7(g/ton) in period nine. In Level 3 it grows from 191.4(g/t) to 215.2(g/t) during periods ten and eleven and it smoothly descendes from 215.2(g/t) to 77.5(g/t) between periods eleven and fourteen. In Level 2 it falls between periods fifteen to sixteen from 151.8(g/t) to 143.9(g/t), and it declines from 163.5(g/t) to 85.2(g/t) between periods seventeen and nineteen. In Level 1 it raises gently for three periods from 131.7(g/t) to 148.3(g/t).

### 4.8. Comparison between SOT and SOL Methods

Table 4.9 presents a brief comparison between production schedules generated by SOT and SOL methods. According to this table, the achieved NPV by SOT method is 22% more than what is got by SOL method. However, the running time of SOT method is 241% more than SOL method. In SOT method, all the stopes are considered together while in SOL method, stopes are divided to the few sets, levels, and then the optimization algorithm is applied for each set separately. As a result, the running time of SOT method is higher than the running time of SOL method.

Table 4.9. Result of SOT and SOL methods

| Method | NPV (M$) | Running time (hr): (min) :(sec) |
|---|---|---|
| SOT | 1,288.1 | 00:46:14 |
| SOL | 1,052.5 | 00:13: 34 |

### 4.9. Summary and Conclusion

A study was used in this chapter to verify the proposed mathematical formulations and BIP models. First, the information of data set was explained. Then, the presented methodologies of stopes layout optimization by LOT method and stopes layout optimization by LOL was applied on the dataset.

Optimum stopes layout of LOT method included 85 stopes. This method reached $ 2,341.1M as economic value and the total running time was 62(hr):02(min):07(sec) and this solution was the optimum solution with the zero percent gap. In LOL method, five levels with 93 stopes and the economic value of $ 2,252.1M were obtained. Total running time of this method was 00:08(min):54(sec).

Then two different methods of SOT and SOL were considered for generating the optimum production schedule over 22 years.

The SOT method achieved NPV of $ 1,288.1M with the running time of 00:46(min):14(sec) and zero percent gap. In the SOL method, the model was applied on all five levels with the different number of stopes. The obtained NPV was $ 1052.5 M. The running time was 00:13(min):34(sec) with the zero percent gap.

# CHAPTER 5: SUMMARY, CONCLUSSIONS AND RECOMMENDATIONS

*Chapter 5 contains the summary and conclusion of this thesis. The contributions of this research are emphasized, as well as the recommendation for future work in stopes layout and production scheduling optimization in sublevel stoping method.*

## 5.1. Summary of Research

The optimum stopes layout goal is to choose the best combination of blocks of the block model to be extracted. By determining the dimension and the locations of the stopes that contain numbers of blocks, the achieved profit can be maximized. Finding the stopes production schedule is the response to questions that which stope and when that stope can be extracted. The primary goal of production scheduling is to determine a sequential order to mine the stopes and provides the maximum net present value (Manchuk, 2007). Since different mining methods obtain the different geotechnical constraints, it is not reasonable to define a general purpose optimization algorithm suited for all underground mining methods (Bai et al., 2012). The existing algorithms for underground stope optimization are divided into two sets of level-based and field-based. Level-based algorithms of stope optimization implement the optimization on the different levels or panels of the block model; however, field-based stope optimization algorithms are applied on the block model before dividing into levels or panels (Sotoudeh et al., 2017). However, the goal of both methods is selection the stopes to have the highest economic value and to define the timing of the mining the stopes to achieve the highest NPV during the life of mine.

Some major weaknesses of the previous studies about designing stopes layout and production scheduling in underground mining are: (i) only a few mathematical programming models with optimum solutions; (ii) not confined to the specific mining method; (iii) not covered all the practical constraints for sublevel stoping; (iv) not considered the level-based and field-based optimization with the same data set; v) restriction on solving large-scale problems.

The focus of this study was on two aspects of three aspects of underground mine planning optimization presented by Topal (2008) which are stopes layout, production schedule and infrastructure. In this study the optimization of stopes layout and production scheduling specifically in sublevel stoping method, which is suitable for the wide vein-type steeply dipping deposits, was investigated. The proposed methodologies generate an optimal solution with meeting constraints such as stopes overlap for stopes layout optimization and only one-time mining, stopes adjacency, the connection between mining the stopes and activation of levels, concurrent active levels, and the delay between activation of the levels constraints for production scheduling optimization. In this study, the stopes layout optimization was done based on the total blocks of the block model altogether (LOT) and based on the separated levels of block model (LOL) to see the impact of

leveling in the stopes layout optimization. Also, two frameworks for optimization of production schedule in sublevel stoping were presented.

MATLAB (MathWorks Inc, 2017) programming platform was used to create the objective function and constraints. IBM ILOG CPLEX Optimization Studio (IBM, 2017) which is a solver for large-scale optimization problems was used in this research.

In case of the stopes layout optimization by LOT method, after generating the block economic model, the positive value stopes was created, and stopes overlaps were assessed. Finally, the best stopes layout was discovered. Optimum stopes layout of this method included 85 stopes. This method reached $ 2,341.1M as the EV and the total running time was 62(hr):02(min):07(sec) and this solution was the optimum solution with the zero percent gap. In case of stopes layout optimization by LOL method, after generating the block economic model, possible levels were defined. Assessing the stopes overlaps in each level and finding the best stopes combinations in each level were the following steps. Discovering the optimum set of levels was the final step. Running the LOL method found 5 levels contain 93 stopes and $ 2,252.1M as the EV. Total running time of this method was 00:08(min):54(sec).

In order to optimize the production schedule by SOT method, determining the life of mine and discounted economic value of each stope in different periods, defining adjacent stopes, determining the relationship between levels and determining all other considerations as the model constraints were all the steps before generating the optimum stope production schedule. In case of the total period of 22 and 93 stopes, achieved NPV was $ 1,288.1M with the running time of 00:46(min):14(sec) and zero percent gap. To optimize the production schedule by SOL. Defining the first level and the period for each level were the first steps. Then, determining NPV of each stope in different periods, adjacent stopes for each level and all other considerations as the model constraints were the following steps. After running the model for all five levels with the different number of stopes in each level, total NPV of $ 1052.5 M was reached. In addition, the running time was 00:13(min): 34(sec) for the zero percent gap.

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

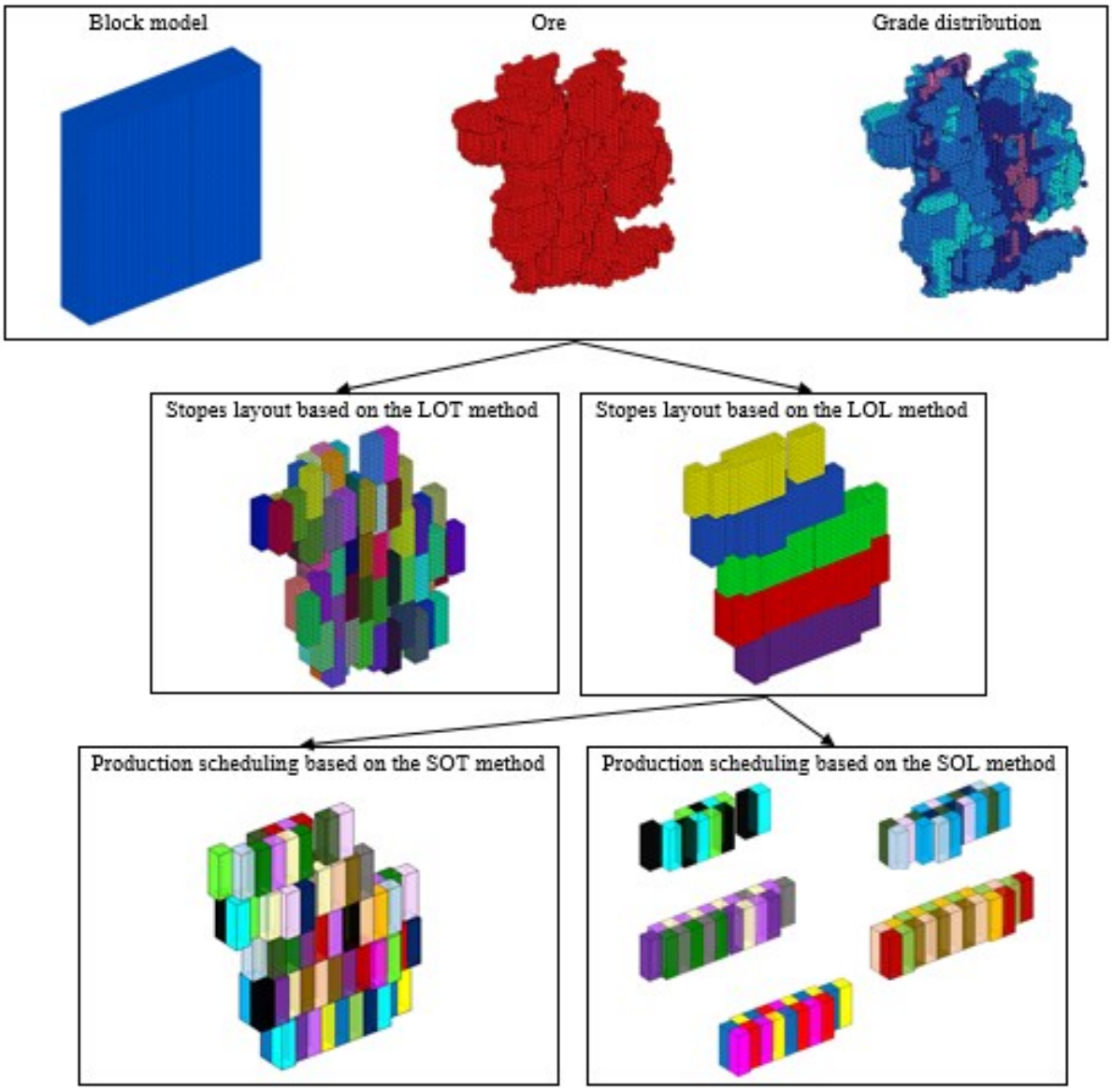



Figure 5.1. Summary of the research methods in the considered case study

One of the purposes of this research was comparing the achieved results of stopes layouts and production schedules by two methods: (i) considering all blocks in the block model together, and (ii) considering all blocks in each level. Figure 5.2 indicates the summary of comparisons between achieved NPVs and running times.

## 5.2. Conclusions

All the research objectives outlined in Chapter 1, have been achieved. The following conclusions

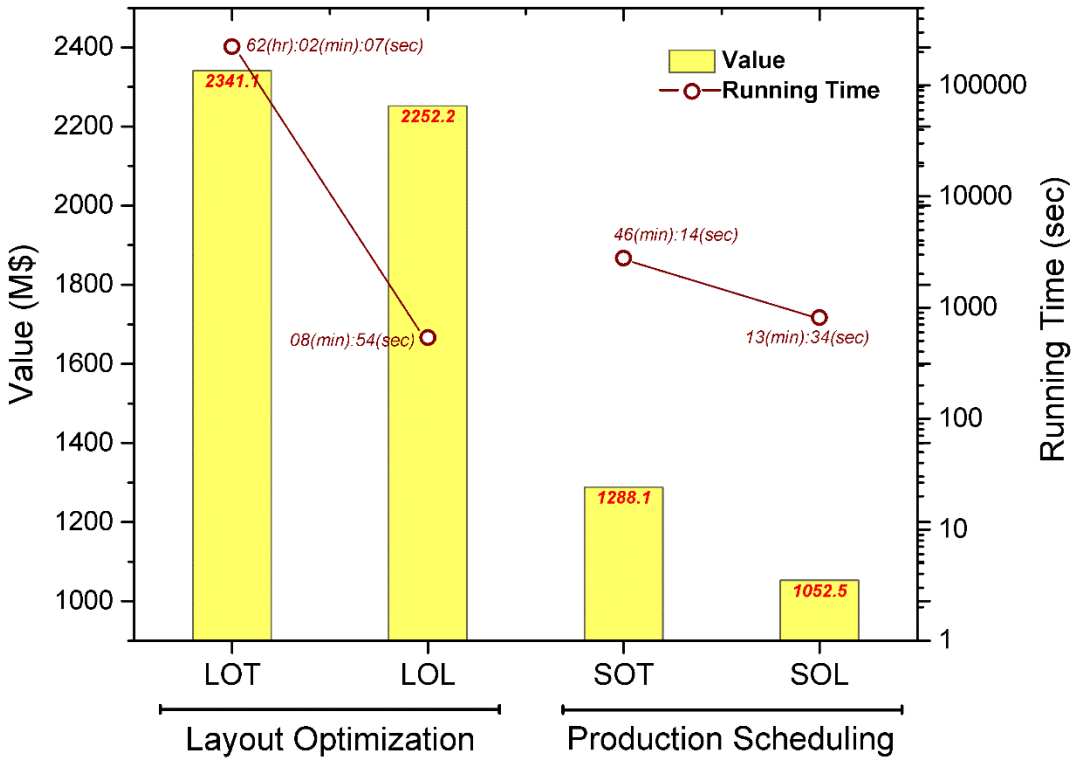



Figure 5.2. Summary of results of all presented methods

were obtained from generating optimization models for stopes layout and production scheduling:

1. The presented methods are able to find the optimum stopes layout in sublevel stoping based on the total blocks and based on the levels while considering the constraints.
2. The proposed BIP models maximizes the NPV of the production scheduling in sublevel stoping while enforcing the model to satisfy the required constraints.
3. The proposed methodologies are verified regarding both feasibility and optimality on a case study and the optimality gap for all four methods are zero percent.
4. The comparison between LOT and LOL methods to find the optimum stopes layout indicates the following results:
    - The achieved value by LOT method is 4% higher than what is got by LOL method.
    - The number of selected stopes by LOT method is 9% less than LOL method.
    - The running time of LOT method is 418 times more than LOL method.

- Regarding the accessibility of production levels to all the stopes, LOL presents the practical stops layout.

5. The comparison between SOT and SOL methods to determine the optimal production schedule demonstrates the following results:
    - The NPV of SOT method is 22% higher than the NPV of SOL method.
    - The running time of SOT method is 3.4 times more than SOL method.
6. According to the results, for stopes layout optimization, it is reasonable to use the leveling to save the time although the achieved EV is a little bit lower. Also, from practicality point of view, LOL reaches the better solution. However, for production scheduling, the model does not show a specific advantage of using leveling method.

## 5.3. Contributions of the Research

This research has implemented optimization techniques to design the stopes layout and developed the mathematical formulations for production scheduling, which contributes notably to generate an optimal stopes layout and production scheduling for the sublevel stoping mining. The following constitute the main contributions of this research.

1. Determining a mathematical model with the optimum solution to create the optimal stopes layout in sublevel stoping mining based on the LOT method, as well as based on the LOL method while satisfying the constraints.
2. Providing a mathematical model with the optimum solution to generate optimal production schedule in sublevel stoping mining method according to the SOT and SOL method maximize the NPV over the life of mine by considering the constraints.
3. Comparing the result of applying the stopes layout based on the LOT and LOL methods and production scheduling optimization based on the SOT and SOL methods and choosing the better method according to the situations.
4. Using the created models on the real size industrial applications since it has been tested on a real sublevel stoping mining case.

5. Considering the practicality by defining the set of practical constraints suitable for sublevel stoping method including stopes overlap, mining capacity, grade blending, only one-time mining, stopes adjacency, the connection between mining the stopes and activation of levels, concurrent active levels, and the delay between activation of the levels constraints.
6. Paying attention to the constraints that were not considered in the previous works include concurrent active levels and the delay between activation of the levels.

## 5.4. Recommendations for Future Research

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

1. One of the assumptions in the proposed model, material mixing between and within the stopes while mining was not considered. In future research, the dilution should be considered during optimization.
2. Another assumption in this study was in order to create stope, no partial block and in order to create a level, no partial stope were considered and considering those for future studies is recommended.
3. In this study, selection of a stope in a period includes of preparation, extraction and backfilling at that period. Considering those functions individually suggested for future works.
4. Using variable size of the stopes for stopes layout optimization is suggested for future researches.
5. In all presented algorithms in this study, the pillars between stopes are not defined which will be grate to contemplate to those.
6. This study presented the deterministic model, so it was not able to capture the uncertainty. It is good to consider this matter in future.

# BIBLIOGRAPHY


[1] Alford, C. (1996). Optimization in Underground Mine Design. *International Journal of Rock Mechanics and Mining Sciences and Geomechanics 220A* 561.

[2] Ataee-Pour, M. (2000). A Heuristic Algorithm to Optimise Stope Boundaries.Thesis, University of Wollongong,

[3] Ataee-Pour, M. (2005). A Critical Survey of the Existing Stope Layout Optimization Techniques *mining science*, *41, No.5* (UDC 519.256:622.2),

[4] Atlas Copco (2000). Underground Mining, viewed 20 August 2011. Retrieved from: http://194.132.104.143/Websites/RDE/Website.nsf/vIndexLookup/Applications&&Underground_Mining?open>.

[5] Bai, X., Marcotte, D. and Simon, D. (2012). Underground Slope Optimization with Network Flow Method. *Computers and Geosciences*, 361-371.

[6] Bootsma, M. T., Alford, C., Benndorf, J. and Buxton, M. W. N. (2014). Cut-off Grade-based Sublevel Stope Mine Optimisation – Introduction and Evaluation of an Optimisation Approach and Method for Grade Risk Quantification. in *OREBODY MODELLING AND STRATEGIC MINE PLANNING SYMPOSIUM* PERTH, WA, pp. 24–26.

[7] Cawrse, I. (2001). Multiple Pass Floating Stope Process. in *4th Biennial Strategic Mine Planning Conference*. Melbourne

[8] Cawrse, I. (2007). Multiple pass floating stope process. in *Strategic Mine Planning Conference*. Perth, Australia, pp. 87-94.

[9] Cheimanoff, N. M., Deliac, E. P. and Mallet, J. L. (1989). GEOCAD: An Alternative CAD and Artificial Intelligence Tool That Helps Moving from Geological Resources to Mineable Reserves. *21st Application of Computers and Operations Research in the Mineral Industry:21st International Symposium*, Page 471.

[10] Copland, T. and Nehring, M. (2016). Integrated optimization of stope boundary selection and scheduling for sublevel stoping operations. *The Southern African Institute of Mining and Metallurgy*, 1135-1142.

[11] Datamine International (1981). Mineable Reserve Optimizer. Wells, United Kingdom.

[12] Deraisme, J., De Fouquet, C. and Fraisse, H. (1984). Geostatistical Orebody Model for Computer Optimization of Profits from Different Underground Mining Methods. in *Proceedings of the 18th APCOM Symposium*. London, England, pp. 583–590.

[13] Deswik Mining Consultants Pty Ltd (2007). Deswik.

[14] Dominski, T., Kolankowski, B., Marck, A., Pasternak, M., andShamba, S. (2014). Estimation of the potential production rate. Mine Design Queen's University, Retrieved from: http://minewiki.engineering.queensu.ca/mediawiki/index.php/Estimation_of_the_potential_production_rate

[15] Hartman, H. L. (1992). *Sublevel stoping in SME Mining Engineering Handbook.* Society for Mining, Metallurgy and Exploration Inc.: Littleton,

[16] Haycocks, C. and Aelick, R. C. (1992). *Sublevel Stoping.* Littleton,

[17] Horst, R. and Tuy, H. (1996). Global optimization: Deterministic approaches. *Springer*, (New York), 727.

[18] Howard, L. H. and Mutmansky, J. M. (2002). *Introductory Mining Engineering.*

[19] IBM (2017). IBM ILOG CPLEX Optimization Studio 12.7.1.

[20] Lawrence, B. W. (1998). Considerations for sublevel stoping techniques in underground mining. *Society for Mining, Metallurgy and Exploration Inc.: Littleton*,

[21] Little, J. (2012). Simultaneous optimisation of stope layouts and production schedules for long-term underground mine planning.Thesis, University of Queensland,

[22] Little, J. and Topal, E. (2011). Strategies to assist in obtaining an optimal solution for an underground mine planning problem using mixed integer programming. *International journal of Mining and Mineral Engineering*, *3, No. 2* 152-172.

[23] Malaki, S. (2016). Block-Cave Extraction Level and Production Scheduling Optimization under Grade Uncertainty.Thesis, University of Alberta,

[24] Manchuk, J. (2007). Stope Design and Sequencing.Thesis, University of Alberta,

[25] Mann, C. (1998). Sublevel Stoping. *RE Gertsch & RL Bullock (ed.), Techniques in underground mining: selections from underground mining methods handbook society for Mining, Metallurgy and Exploration Inc*, 223-224.

[26] Maptek Co (1981). Vulcan 8.1 Stope Optimizer. Adelaide, Australia.

[27] MathWorks Inc (2017). MATLAB (R2017a). Ver. 9.2,

[28] Mintec Inc. (1970). MineSight.

[29] Nehring, M. (2011). Integrated Production Schedule Optimisation for Sublevel Stoping Mines.Thesis, The University of Queensland,

[30] Nhleko, S., Tholana, T. and Neingo, P. (2018). A review of underground stope boundary optimization algorithms. *Resources Policy*, *https://doi.org/10.1016/j.resourpol.2017.12.004*

[31] Nikbin, V., Ataee-Pour, M., Shahriar, K. and Pourrahimian, Y. (2017). A Greedy Algorithm for Stope Boundaries Optimization in *Mining Optimization Laboratory (MOL) University of Alberta*, pp. Report Eight: 246–252.

[32] Nikbin, V., Ataee-Pour, M., Shahriar, K. and Pourrahimian, Y. (2018). A 3D approximate hybrid algorithm for stope boundary optimization. *Computers and Operations Research*, *https://doi.org/10.1016/j.cor.2018.05.012* 1–9.

[33] Ovanic, J. and Young, D. S. (1995). Economic Optimisation of Stope Geometry Using Separable Programming with Special Branch and Bound Techniques in *Third Canadian Conference on Computer Applications in the Mineral Industry, Rotterdam, Balkema*, pp. 129–135.

[34] Pferschy, U. and Schauer, J. (2009). The Knapsack Problem with Conflict Graph. *Journal of Graph Algorithms and Applications*, *vol. 13, no. 2* 233–249.

[35] Plotcube Retrieved from: https://www.mathworks.com/matlabcentral/fileexchange/15161-plotcube

[36] Pourrahimian, Y. (2013). Mathematical programming for sequence optimization in block cave mining.Thesis, University of Alberta,

[37] Riddle, J. M. (1977). A Dynamic Programming Solution of A Block-Caving Mine Layout. *Application of Computer Methods in the Mineral Industry: Proceedings of the Fourteenth Symposium*, 767–780.

[38] Sandanayake, D. S. S. (2014). Stope Boundary Optimisation in Underground Mining Based on A Heuristic Approach.Thesis, Curtin University

[39] Sandanayake, D. S. S., Topal, E. and Asad, M. W. A. (2015a). A Heuristic Approach to Optimal Design of An Underground Mine Stope Layout. *Applied Soft Computing*, *1568-4946*

[40] Sandanayake, D. S. S., Topal, E. and Asad, M. W. A. (2015b). Designing An Optimal Stope Layout for Underground Mining Based on A Heuristic Algorithm. *International Journal of Mining Science and Technology*, *25* (5), 767-772.

[41] Sens, J. J. (2011). Stope mine design optimisation using various algorithms for the Randgold Kibali project.Thesis, Delft University of Technology,

[42] Silverprice Retrieved from: https://silverprice.org/silver-price-canada.html

[43] Sotoudeh, F., Kakaie, R. and Ataei, M. (2017). Development of a computer program for underground mine stope optimisation using a heuristic algorithm. in *First International Conference on Underground Mining Technology*. Sudbury, Canada: Australian Centre for Geomechanics, Perth, ISBN 978-0-9924810-7-0

[44] Topal, E. (2008). Early start and late start algorithms to improve the solution time for long-term underground mine production scheduling. *The Southern African Institute of Mining and Metallurgy*, *108* 99-1 079.

[45] Topal, E. and Sens, J. (2010). A New Algorithm for Stope Boundary Optimization. *Journal of Coal Science and Engineering (China)*, *16* (2), 113-119.

[46] Villaecusa, E. (2003). Global extraction sequences in sublevel stoping. *The AusIMM Bulletin, The Australasian Institute of Mining and Metallurgy: Melbourne*, 9-17.

[47] Villaecusa, E. (2014). Sublevel Stoping Geometery. in *Geothecnical Design for Sublevel Open Stoping*, Vol. 13: 978-1-4822-1189-4, CRC Press, Taylor & Francis Group,

[48] Villalba. M, M. E. and Kumral, M. (2017). Heuristic Stope Layout Optimisation Accounting for Variable Stope Dimensions and Dilution Management. *Int. J. Mining and Mineral Engineering*, *Vol. 8, No. 1* 1–18.

[49] Villalba. M, M. E. and Kumral, M. (2018). Underground mine planning: stope layout optimisation under grade uncertainty using genetic algorithms. *International Journal of Mining, Reclamation and Environment*,

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

# APPENDIX A

This section includes codes of the presented models for stopes layout optimization based on total blocks (LOT). The order of these codes is as followed.

- A1) Import the block model data set.
- A2) Get economic parameters; calculate Cut-off and blocks economic value.
- A3) Get stope dimensions.
- A4) Create initial stopes and positive stopes.
- A5) Find Stopes without Overlap.
- A6) Create the objective function.
- A7) Create the constraint matrix and its upper bound.
- A8) Run the optimization model.
- A9) Plot the result.

**A1) Import the block model data set.**

```
function F1_ReadBM1
[f,p]= uigetfile('*.xlsx');
pf= [p,f];
[num,txt,raw]=xlsread(pf);
DataBase.Excel.General = raw;
DataBase.Excel.BN = xlsread(pf,'Data','BN');          %Number of Blocks
DataBase.Excel.XI = xlsread(pf,'Data','XI');          %X Index
DataBase.Excel.YI = xlsread(pf,'Data','YI');          %Y Index
DataBase.Excel.ZI = xlsread(pf,'Data','ZI');          %Z Index
DataBase.Excel.Grade = xlsread(pf,'Data','Grade');    %Block Average Grade
DataBase.Excel.BTon = xlsread(pf,'Data','BTon');      %Block Tonnage
DataBase.EconomicPar.Pri= xlsread(pf,'Data','Pri');
DataBase.EconomicPar.CoS= xlsread(pf,'Data','CoS');
DataBase.EconomicPar.CoM= xlsread(pf,'Data','CoM');
DataBase.EconomicPar.CoP= xlsread(pf,'Data','CoP');
DataBase.EconomicPar.Rec= xlsread(pf,'Data','Rec');
DataBase.GetDim.LenX= xlsread(pf,'Data','LenX');
DataBase.GetDim.LenY= xlsread(pf,'Data','LenY');
DataBase.GetDim.LenZ= xlsread(pf,'Data','LenZ') ;
DataBase.Others.Minb= xlsread(pf,'Data','Minb');
save('DataBase','DataBase');
msgbox(sprintf('The Number of Imported Blocks are:%d', N),'Stope Optimizer');
```

**A2) Get economic parameter; calculate Cut-off and blocks economic value.**

```
function F2_EconomicPar
load DataBase.mat
Pri=DataBase.EconomicPar.Pri;
CoS=DataBase.EconomicPar.CoS;
CoM=DataBase.EconomicPar.CoM;
CoP=DataBase.EconomicPar.CoP;
Rec=DataBase.EconomicPar.Rec;
CutOffGrade = (CoP+CoM)/((Pri-CoS)*Rec/100)
DataBase.EconomicPar.CutOffGrade= CutOffGrade;
%Calculate Block Economic Value
Grade=DataBase.Excel.Grade;
BTon=DataBase.Excel.BTon;
BN=DataBase.Excel.BN;
N = length(BN);
BEV= zeros(N,1);
for iloop=1:N
    if   Grade(iloop,1)<CutOffGrade
         BEV(iloop,1)=-CoM*BTon(iloop,1);
    else
BEV(iloop,1)=(((Pri-CoS)*Grade(iloop,1)*(Rec/100))-(CoM+CoP))*BTon(iloop,1);
    end
end
DataBase.EconomicPar.BEV = BEV;
save('DataBase','DataBase');
msgbox('Cut_off Grade & Block Economic Values Were Calculated','Stope
Optimizer');
```

**A3) Get Stope Dimensions.**

```
function F3_GetDim
load DataBase.mat
LenX= DataBase.GetDim.LenX;
LenY= DataBase.GetDim.LenY;
LenZ= DataBase.GetDim.LenZ;
save('DataBase','DataBase');
msgbox('Stopes Dimensions Were Determined','Stope Optimizer');
```

**A4) Create initial stopes and positive stopes.**

```
function F4_CreateStopes
load('DataBase','DataBase');
BN=DataBase.Excel.BN;
LenX=DataBase.GetDim.LenX;
LenY=DataBase.GetDim.LenY;
LenZ=DataBase.GetDim.LenZ;
Minb= DataBase.Others.Minb;
XI = [DataBase.Excel.XI];
YI = [DataBase.Excel.YI];
ZI = [DataBase.Excel.ZI];
Grade = [DataBase.Excel.Grade];
BTon = [DataBase.Excel.BTon];
BEV = DataBase.EconomicPar.BEV;
MaxX=max(XI);
MaxY=max(YI);
MaxZ=max(ZI);
MinX=min(XI);
MinY=min(YI);
MinZ=min(ZI);
SizeX=max(XI) - min(XI) + 1;
SizeY=max(YI) - min(YI) + 1;
SizeZ=max(ZI) - min(ZI) + 1;
TotalNomberOfBlockS= SizeX*SizeY*SizeZ;
SingleZeroBlock.BTN=0;
SingleZeroBlock.ZIndex=0;
SingleZeroBlock.YIndex=0;
SingleZeroBlock.XIndex=0;
TotalBlocks = repmat(SingleZeroBlock,[TotalNomberOfBlockS 1]);
for ZILoop=1:SizeZ
    for YILoop=1:SizeY
        for XILoop = 1:SizeX
            RowNumber = (ZILoop-1)*SizeY*SizeX + (YILoop-1)*SizeX + XILoop;
            TotalBlocks(RowNumber).BTN=RowNumber;
            TotalBlocks(RowNumber).XIndex = XILoop + MinX - 1;
            TotalBlocks(RowNumber).YIndex = YILoop + MinY - 1;
            TotalBlocks(RowNumber).ZIndex = ZILoop + MinZ - 1;
            G = Grade(XI == TotalBlocks(RowNumber).XIndex &...
            YI == TotalBlocks(RowNumber).YIndex &...
            ZI == TotalBlocks(RowNumber).ZIndex);
            if ~isempty(G)
               TotalBlocks(RowNumber).Grade = G;
            end
            T = BTon(XI == TotalBlocks(RowNumber).XIndex &...
                YI == TotalBlocks(RowNumber).YIndex &...
                ZI == TotalBlocks(RowNumber).ZIndex);
            if ~isempty(T)
               TotalBlocks(RowNumber).BTon = T;
            end
                EV = BEV(XI == TotalBlocks(RowNumber).XIndex &...
                YI == TotalBlocks(RowNumber).YIndex &...
                ZI == TotalBlocks(RowNumber).ZIndex);
            if ~isempty(EV)
                TotalBlocks(RowNumber).BEV = EV;
            end

        end
```

```
    end
end
DataBase.TotalBlocks=TotalBlocks;
N = SizeX*SizeY*SizeZ;
SN(N).SBN=0;
M=SizeX*SizeY;
C=M-((LenY-1)*SizeX)-(LenX-1);
A=C+((SizeZ-LenZ)*M);
for i=1:A
   n=floor(i/M);
   b=i-(n*M);
    if b<=C
        for j=0:(LenZ-1)
           if  (SizeX-rem(i,SizeX)>= (LenX-1))&& (rem(i,SizeX)~=0)
                for k=0:(LenY-1)
                    for e=0:(LenX-1)
                       if TotalBlocks((n*M)+b+(j*M)+(k*SizeX)+e).BTon ~= 0
                          SN(i).SBN=[SN(i).SBN
,TotalBlocks((n*M)+b+(j*M)+(k*SizeX)+e).BTN];
                       end
                    end
                end
           end
       end
     end
end

Result.IniStopes=SN;
for iloop=N:-1:1
    if isempty(Result.IniStopes(iloop).SBN) | Result.IniStopes(iloop).SBN==0|
length(Result.IniStopes(iloop).SBN)<=(Minb-1)
       Result.IniStopes(iloop)=[];
    end
    Result.IniStopes=Result.IniStopes;
end
N = length(Result.IniStopes);
for iloop=1:N

Result.IniStopes(iloop).StopeValue=sum([TotalBlocks(Result.IniStopes(iloop).S
BN).BEV]);
end
DataBase.TotalBlocks=TotalBlocks;
%Find Positive Stopes and Save
N = length(Result.IniStopes);
jloop=1;
for iloop=1:N
        if  Result.IniStopes(iloop).StopeValue> 0
            Result.PositiveStopes(jloop).PSBN=Result.IniStopes(iloop).SBN;

Result.PositiveStopes(jloop).PositiveStopeValue=Result.IniStopes(iloop).Stope
Value;
            jloop=jloop+1;
        end
end
save('DataBase','DataBase');
save('Result','Result');
msgbox('Positive Value Stopes Were Determined','Stope Optimizer');
```

**A5) Find Stopes without Overlap.**

```
function F5_OverLapFinder
load Result.mat;
N = length(Result.PositiveStopes);
HaveOverLap=sparse(N,N);
for i=1:N
     for j=1:N

ConcatenateToFinOverLaps=[Result.PositiveStopes(i).PSBN,Result.PositiveStopes
(j).PSBN];
          EleminateSameBlocks=unique(ConcatenateToFinOverLaps);
            if length(ConcatenateToFinOverLaps)>length(EleminateSameBlocks)
                HaveOverLap(i,j)=1;
            end
     end
end
if exist('OPT_DataBase.mat') == 2
    load OPT_DataBase
    OPT_DataBase.OverLap=HaveOverLap;
else
    OPT_DataBase.OverLap=HaveOverLap;
end
save('OPT_DataBase','OPT_DataBase')
msgbox('The Overlaps Were Found');
```

**A6) Create the objective function.**

```
function F6_OPT1_OBJfunc
load Result.mat
PS=[Result.PositiveStopes];
Optimization1.PosStopes=[PS.PositiveStopeValue]';
Optimization1.OBJfun = (-1).*(Optimization1.PosStopes)';
save('Optimization1','Optimization1')
h = msgbox('Objective function was created successfully');
```

**A7) Create the constraint matrix and Upper bound.**

```
function F7_OPT2_Cons_Overlap
load OPT_DataBase.mat
numStopes = size(OPT_DataBase.OverLap,1);
A = sparse(numStopes*numStopes,numStopes);
A(1,:) = ones(1,numStopes);
rowLoop = 1;
for iLoop = 1:numStopes
    index = find(OPT_DataBase.OverLap(iLoop,:));
    numindex = numel(index);
      if isempty(numindex) == 0
        for jLoop =1:numindex
            A(rowLoop,index(jLoop)) = 1;
                A(rowLoop,iLoop) = 1;
                rowLoop = rowLoop + 1;
        end
    end
end
A(rowLoop:end,:) = [];
A = sparse(A);
Optimization2.A = A;
b = ones(size(A,1),1);
b(1,1) = numStopes;
Optimization2.b = b;
save('Optimization2','Optimization2')
h = msgbox('Constraint(s)was created successfully');
```

**A8) Run the optimization model.**

```
function F8_OPT3_Run_bintprog
warning off
load 'Optimization1'.mat
load 'Optimization2'.mat
f = Optimization1.OBJfun;
A = Optimization2.A;
b = Optimization2.b;
addpath 'C:\Program Files\IBM\ILOG\CPLEX_Studio1271\cplex\matlab\x64_win64';
options = cplexoptimset;
options.Display = 'on';
   [x, fval, exitflag, output] = cplexbilp (f, A, b, [], [],[ ], options);
fprintf ('\nSolution status = %s\n', output.cplexstatusstring);
fprintf ('Solution value = %d\n', fval);
Optimization3.Result.x = x;
Optimization3.Result.fval = fval;
rmpath 'C:\Program Files\IBM\ILOG\CPLEX_Studio1271\cplex\matlab\x64_win64';
save('Optimization3','Optimization3')
```

**A9) Plot the result.**

```
function Plot_sec
load OPT_DataBase.mat
load('Result','Result');
load('DataBase','DataBase');
load Optimization3.mat
StopesinBestCombination=find(Optimization3.Result.x);
LenX= DataBase.GetDim.LenX;
LenY= DataBase.GetDim.LenY;
LenZ= DataBase.GetDim.LenZ;
figure
for iLoop=1:length(StopesinBestCombination)

    X =
[DataBase.TotalBlocks(Result.PositiveStopes(StopesinBestCombination(iLoop)).P
SBN).XIndex];
    Y =
[DataBase.TotalBlocks(Result.PositiveStopes(StopesinBestCombination(iLoop)).P
SBN).YIndex];
    Z =
[DataBase.TotalBlocks(Result.PositiveStopes(StopesinBestCombination(iLoop)).P
SBN).ZIndex];
    Cl = [rand(1) rand(1) rand(1)];
    for jLoop=1:length(X)
        plotcube([1 1 1],[X(jLoop) Y(jLoop) Z(jLoop)],.8,Cl);
    end
end
set(gca,'fontsize',11)
ax = gca;
ax.ZDir = 'reverse';
XL = ax.XLim;
YL = ax.YLim;
ZL = ax.ZLim;
LimMin = min([XL(1),YL(1),ZL(1)]);
LimMax = max([XL(2),YL(2),ZL(2)]);
ax.XLim = [LimMin LimMax];
ax.YLim = [LimMin LimMax];
ax.ZLim = [LimMin LimMax];
xlabel('Easting-X')
ylabel('Northing-Y')
zlabel('Elevation-Z')
axis equal
end
```

# APPENDIX B

This section includes codes of the presented models for stopes layout optimization based on the level (LOL). The order of these codes is as followed.

- B1) Create Initial and positive stopes and find the best stopes combination for each level.
- B2) Create objective function to find the best levels.
- B3) Create the constraint and Upper bound.
- B4) Run the optimization model to find the best levels set.
- B5) Plot the result.

**B1) Create Initial and positive stopes and find the best stopes combination for each level.**

```
function F4Level_1
load('DataBase','DataBase');
BN=DataBase.Excel.BN;
LenX=DataBase.GetDim.LenX;
LenY=DataBase.GetDim.LenY;
LenZ=DataBase.GetDim.LenZ;
XI = [DataBase.Excel.XI];
YI = [DataBase.Excel.YI];
ZI = [DataBase.Excel.ZI];
Grade = [DataBase.Excel.Grade];
BTon = [DataBase.Excel.BTon];
BEV = DataBase.EconomicPar.BEV;
MaxX=max(XI);
MaxY=max(YI);
MaxZ=max(ZI);
MinX=min(XI);
MinY=min(YI);
MinZ=min(ZI);
SizeX=max(XI) - min(XI) + 1;
SizeY=max(YI) - min(YI) + 1;
SizeZ=max(ZI) - min(ZI) + 1;
Result=repmat(struct('IniStopes',[],'PositiveStopes',[],'LevelValue',[],'Have
OverLap',[]),[MaxZ-(LenZ-MinZ) 1]);
for Lloop=1:MaxZ-(LenZ-MinZ)
    Level=(LenZ-MinZ) + Lloop;
    M=SizeX*SizeY;
    C=M-((LenY-1)*SizeX)-(LenX-1);
    SN=repmat(struct('SBN',[]),[C 1]);
    StopeNumber = 1;
           for i=((M*(Level-1))+1):((M*(Level-1))+C)
         if  (SizeX-rem(i,SizeX)>= (LenX-1))&& rem(i,SizeX)~= 0
            for j=0:(LenY-1)
                for k=0:(LenX-1)
                    for m=0:(LenZ-1)
                        SN(StopeNumber).SBN=[SN(StopeNumber).SBN
,BN((SizeX*j)+i+k-(m*M))];
                    end
                end
            end
        end
         StopeNumber = StopeNumber + 1;
       end
  Result(Lloop).IniStopes=SN;
    N = length(Result(Lloop).IniStopes);
    for iloop=1:N

Result(Lloop).IniStopes(iloop).StopeValue=sum(DataBase.EconomicPar.BEV(Result
(Lloop).IniStopes(iloop).SBN));

Result(Lloop).IniStopes(iloop).StopeTonnage=sum(DataBase.Excel.BTon(Result(Ll
oop).IniStopes(iloop).SBN));

Result(Lloop).IniStopes(iloop).AvgGrade=(sum((DataBase.Excel.Grade(Result(Llo
op).IniStopes(iloop).SBN)).*(DataBase.Excel.BTon(Result(Lloop).IniStopes(iloo
p).SBN)))/(sum(DataBase.Excel.BTon(Result(Lloop).IniStopes(iloop).SBN))));
```

```
    end    N = length(Result(Lloop).IniStopes);
    jloop=1;
    for iloop=1:N
            if  Result(Lloop).IniStopes(iloop).StopeValue> 0

Result(Lloop).PositiveStopes(jloop).PSBN=Result(Lloop).IniStopes(iloop).SBN;
Result(Lloop).PositiveStopes(jloop).PositiveStopeValue=Result(Lloop).IniStope
s(iloop).StopeValue;

Result(Lloop).PositiveStopes(jloop).PositiveStopeTonnage=Result(Lloop).IniSto
pes(iloop).StopeTonnage;

Result(Lloop).PositiveStopes(jloop).PositiveStopeGrade=Result(Lloop).IniStope
s(iloop).AvgGrade;
                jloop=jloop+1;
            end
    end

Result(Lloop).LevelValue=sum([Result(Lloop).PositiveStopes.PositiveStopeValue
]');
N = length(Result(Lloop).PositiveStopes);
HaveOverLap=sparse(N,N);
for i=1:N
     for j=1:N

ConcatenateToFinOverLaps=[Result(Lloop).PositiveStopes(i).PSBN,Result(Lloop).
PositiveStopes(j).PSBN];
         EleminateSameBlocks=unique(ConcatenateToFinOverLaps);
           if length(ConcatenateToFinOverLaps)>length(EleminateSameBlocks)
               HaveOverLap(i,j)=1;
           end
     end
end
Result(Lloop).HaveOverLap=HaveOverLap;
% Create Objective Function for Each Level
PS=[Result(Lloop).PositiveStopes];
PS=[PS.PositiveStopeValue]';
Result(Lloop).OBJfun = (-1).*(PS)';
% Create Cons and bond for Each Level
numStopes = size(Result(Lloop).HaveOverLap,1);
A = sparse(numStopes*numStopes,numStopes);
A(1,:) = ones(1,numStopes);
rowLoop = 1;
for iLoop = 1:numStopes
    index = find(Result(Lloop).HaveOverLap(iLoop,:));
    numindex = numel(index);
    if isempty(numindex) == 0
        for jLoop =1:numindex
            A(rowLoop,index(jLoop)) = 1;
                A(rowLoop,iLoop) = 1;
                rowLoop = rowLoop + 1;
        end
    end
end
A(rowLoop:end,:) = [];
A = sparse(A);
b = ones(size(A,1),1);
```

```
b(1,1) = numStopes;
Result(Lloop).A=A;
Result(Lloop).b=b;
% Run Optimization for Each Level
f = Result(Lloop).OBJfun;
addpath 'C:\Program Files\IBM\ILOG\CPLEX_Studio1271\cplex\matlab\x64_win64';
options = cplexoptimset;
options.mip.tolerances.mipgap = 0.01;
options.Display = 'on';
[x, fval, exitflag, output] = cplexbilp (f, A, b, [], [],[ ], options);
fprintf ('\nSolution status = %s\n', output.cplexstatusstring);
fprintf ('Solution value = %d\n', fval);
Result(Lloop).x = x;
Result(Lloop).fval = fval;
rmpath 'C:\Program Files\IBM\ILOG\CPLEX_Studio1271\cplex\matlab\x64_win64';
%calculate Final Level Value
Result(Lloop).FinalLevelValue=(-1).*(Result(Lloop).fval);
end
save('Result','Result');
%Create Message Box
msgbox('Level Information Were Determined','Stope Optimizer');
```

**B2) Create objective function.**

```
function F4Level_2
load Result.mat
LevelOptimization.Values=[Result.FinalLevelValue]';
LevelOptimization.OBJfun = (-1).*(LevelOptimization.Values)';
save('LevelOptimization','LevelOptimization')
h = msgbox('Objective function was created successfully');
```

**B3) Create the constraint and Upper bound.**

```
function F4Level_3
load Result.mat
load('DataBase','DataBase');
FLV = [Result.FinalLevelValue];
LenZ=DataBase.GetDim.LenZ;
numLevels=length(FLV);
A1(1,:) = ones(1,numLevels);
b1(1,1) = numLevels;
A2 = zeros(numLevels,numLevels);
 for rowloop = 1:numLevels
     for colloop = 1:numLevels
         if rowloop==colloop
              for zloop=0:(LenZ-1)
                if colloop+zloop <=numLevels
                  A2(rowloop,colloop+zloop) = 1;
                  end
              end
          end
    end
 end
b2 = ones(size(A2,1),1);
A=[A1;A2];
b=[b1;b2];
ConsLevelOptimization.A= A;
ConsLevelOptimization.b= b;
save('ConsLevelOptimization','ConsLevelOptimization')
h = msgbox('Constraint(s)was created successfully');
```

**B4) Run the optimization model to find the best levels set.**

```
function F4Level_4
warning off
load 'ConsLevelOptimization'.mat
load 'LevelOptimization'.mat
f = LevelOptimization.OBJfun;
A = ConsLevelOptimization.A;
b = ConsLevelOptimization.b;
addpath 'C:\Program Files\IBM\ILOG\CPLEX_Studio1271\cplex\matlab\x64_win64';
options = cplexoptimset;
options.mip.tolerances.mipgap = 0.01;
options.Display = 'on';
[x, fval, exitflag, output] = cplexbilp (f, A, b,[],options);
fprintf ('\nSolution status = %s\n', output.cplexstatusstring);
fprintf ('Solution value = %d\n', fval);
LevelOptimization.Result.x = x;
LevelOptimization.Result.fval = fval;
rmpath 'C:\Program Files\IBM\ILOG\CPLEX_Studio1271\cplex\matlab\x64_win64';
save('LevelOptimization','LevelOptimization')
```

**B5) Plot the result.**

```
function Plot_layout
load('Output','Output');
load('DataBase','DataBase');
Bloc=Output.BlocksinBestCombination;
figure(1)
for iloop=1:length(Bloc)

    X = [DataBase.Excel.XI(Bloc(iloop))];
    Y = [DataBase.Excel.YI(Bloc(iloop))];
    Z = [DataBase.Excel.ZI(Bloc(iloop))];
          if  Z<14
           Cl = [1 1 0];
          end
          if Z>=14 && Z<26
           Cl = [0.14 0.36 0.87];
          end
          if Z>=26 && Z<38
           Cl = [0 1 0.14];
          end
          if Z>=38 && Z<50
            Cl = [1 0 0];
          end
          if Z>=50 && Z<63
          Cl = [0.44 0.18 0.63];
          end
        plotcube([1 1 1],[X Y Z],.8,Cl);
 end
 set(gca,'fontsize',12)
ax = gca;
ax.ZDir = 'reverse';
XL = ax.XLim;
YL = ax.YLim;
ZL = ax.ZLim;
LimMin = min([XL(1),YL(1),ZL(1)]);
LimMax = max([XL(2),YL(2),ZL(2)]);
axis equal
xlabel('Easting-X')
ylabel('Northing-Y')
zlabel('Elevation-Z')

end
```

# APPENDIX C

This section includes codes of the presented models for production scheduling optimization based on the total stopes (SOT). The order of these codes is as followed.

- C1) Prepare the inputs.
- C2) Import the scheduling data.
- C3) Generate the objective function.
- C4) Create the grade blending constraints.
- C5) Create mining capacity constraints.
- C6) Create only one time mining constraint.
- C7) Generate stopes adjacent constraint.
- C8) Generate the connection between levels and their stopes constraints.
- C9) Create concurrent levels constraint
- C10) Create delay between levels constraint
- C11) Run the optimization model.
- C12) Plot the result.

**C1) prepare inputs.**

```
function OutPutOptimization
load('Result','Result');
load('DataBase','DataBase');
load('LevelOptimization','LevelOptimization');
LevelsinBestCombination=find(LevelOptimization.Result.x);
LL = [LevelsinBestCombination]';
PosStop = [];
PosBloc = [];
StVal= [];
StTon= [];
StGra= [];
for iLoop=LL
    PosStop=[PosStop,Result(iLoop).PositiveStopes(logical(Result(iLoop).x))];
PosBloc=[PosBloc,Result(iLoop).PositiveStopes(logical(Result(iLoop).x)).PSBN]
;
StVal=[StVal,Result(iLoop).PositiveStopes(logical(Result(iLoop).x)).PositiveS
topeValue];
StTon=[StTon,Result(iLoop).PositiveStopes(logical(Result(iLoop).x)).PositiveS
topeTonnage];
StGra=[StGra,Result(iLoop).PositiveStopes(logical(Result(iLoop).x)).PositiveS
topeGrade];
end
 Output.BlocksinBestCombination=[PosBloc]';
 Output.StopesinBestCombination=PosStop;
 Output.StVal=StVal;
 Output.StTon=StTon;
 Output.StGra=StGra;
 save('Output','Output')
end
```

**C2) Import the scheduling data.**

```
function S1_input_parameters
load Output.mat
load Result.mat
load DataBase.mat
[f,p]= uigetfile('*.xlsx');
pf= [p,f];
[num,txt,raw]=xlsread(pf);
Scheduling.InputParameters.T = xlsread(pf,'SchedulingData','T');
Scheduling.InputParameters.MinGra = xlsread(pf,'SchedulingData','MinG');
Scheduling.InputParameters.MaxGra = xlsread(pf,'SchedulingData','MaxG');
Scheduling.InputParameters.MinCap = xlsread(pf,'SchedulingData','MinC');
Scheduling.InputParameters.MaxCap = xlsread(pf,'SchedulingData','MaxC');
Scheduling.InputParameters.I = xlsread(pf,'SchedulingData','IRR');
Scheduling.InputParameters.R = xlsread(pf,'SchedulingData','Rec');
Scheduling.InputParameters.DelayBetweenLayers =
xlsread(pf,'SchedulingData','Delay');
Scheduling.InputParameters.NumberOfConcurrentLayers=xlsread(pf,'SchedulingDat
a','Concurrent');
%create stope information
T=Scheduling.InputParameters.T;
SelectedStopes=Output.StopesinBestCombination;
StVal=Output.StVal;
StTon=Output.StTon;
StGra=Output.StGra;
NumOfStops = length(StVal);
NumOfPeriods=length(T);
Scheduling.InputParameters.StVal=StVal;
Scheduling.InputParameters.StTon=StTon;
Scheduling.InputParameters.StGra=StGra;
Scheduling.InputParameters.NumOfStops=NumOfStops;
Scheduling.InputParameters.NumOfPeriods=NumOfPeriods;
Scheduling.InputParameters.SelectedStopes=SelectedStopes;
StopeXCordinatePoint=[zeros(size(SelectedStopes))]';
StopeYCordinatePoint=[zeros(size(SelectedStopes))]';
StopeZCordinatePoint=[zeros(size(SelectedStopes))]';
   for iloop=1:NumOfStops
     StopeXCordinatePoint(iloop)=
min([DataBase.Excel.XI(SelectedStopes(iloop).PSBN)]);
     StopeYCordinatePoint(iloop)=
min([DataBase.Excel.YI(SelectedStopes(iloop).PSBN)]);
     StopeZCordinatePoint(iloop)=
min([DataBase.Excel.ZI(SelectedStopes(iloop).PSBN)]);
   end
Scheduling.InputParameters.CordinatePoint.X=StopeXCordinatePoint;
Scheduling.InputParameters.CordinatePoint.Y=StopeYCordinatePoint;
Scheduling.InputParameters.CordinatePoint.Z=StopeZCordinatePoint;
save('Scheduling','Scheduling')
end
```

**C3) Generate the objective function.**

```
function S2_ObjectiveFunction
load Scheduling.mat
load LevelOptimization.mat
T =Scheduling.InputParameters.T;
StVal=Scheduling.InputParameters.StVal;
NumOfStops=Scheduling.InputParameters.NumOfStops;
NumOfPeriods=Scheduling.InputParameters.NumOfPeriods;
NumOfLevels=length(find(LevelOptimization.Result.x));
I=Scheduling.InputParameters.I;
ObjectiveFunction1 = zeros(1,NumOfStops*NumOfPeriods);
  for tloop = 1:NumOfPeriods
       for sloop = 1:NumOfStops
                 ObjectiveFunction1(1,(((tloop-1)*NumOfStops)+sloop)) =
StVal(sloop)/((1+I)^(tloop-1)');
       end
  end
ObjectiveFunction2 = zeros(1,NumOfLevels*NumOfPeriods);
ObjectiveFunction=[ObjectiveFunction1 ObjectiveFunction2];
Scheduling.f =(-1).*(ObjectiveFunction);
save('Scheduling','Scheduling')
end
```

**C4) Create the grade blending constraints.**

```
function S3_Const_Grade
load Scheduling.mat
load LevelOptimization.mat
T =Scheduling.InputParameters.T;
MinG=Scheduling.InputParameters.MinGra;
MaxG=Scheduling.InputParameters.MaxGra;
StTon=Scheduling.InputParameters.StTon;
StGra=Scheduling.InputParameters.StGra;
R=Scheduling.InputParameters.R;
NumOfStops=Scheduling.InputParameters.NumOfStops;
NumOfPeriods=Scheduling.InputParameters.NumOfPeriods;
NumOfLevels=length(find(LevelOptimization.Result.x));
%Frist part
GradeConstMax1 = zeros(NumOfPeriods,NumOfStops*NumOfPeriods);
for rowloop = 1:NumOfPeriods
    for tloop = 1:NumOfPeriods
      for sloop = 1:NumOfStops
         if    rowloop == tloop
    GradeConstMax1(rowloop,(((tloop-1)*NumOfStops)+sloop)) =
StTon(sloop).*((StGra(sloop))-(MaxG(tloop))).*(R(tloop))';
         end
      end
    end
end
%Second part
GradeConstMax2=zeros(NumOfPeriods,NumOfLevels*NumOfPeriods);
GradeConstMax=[GradeConstMax1 GradeConstMax2];
Scheduling.Constraints.A1_GradeConstMax = GradeConstMax;
GradeConstMaxb=zeros(size(GradeConstMax,1),1);
Scheduling.RHSides.b1_GradeConstMaxb = GradeConstMaxb;
GradeConstMin1 = zeros(NumOfPeriods,NumOfStops*NumOfPeriods);
for rowloop = 1:NumOfPeriods
    for tloop = 1:NumOfPeriods
      for sloop = 1:NumOfStops
         if    rowloop == tloop
                GradeConstMin1(rowloop,(((tloop-1)*NumOfStops)+sloop)) =(-
1).*( StTon(sloop).*((StGra(sloop))-(MinG(tloop))).*(R(tloop))') ;
         end
      end
    end
end
% second part
GradeConstMin2=zeros(NumOfPeriods,NumOfLevels*NumOfPeriods);
GradeConstMin=[GradeConstMin1 GradeConstMin2];
Scheduling.Constraints.A2_GradeConstMin = GradeConstMin;
GradeConstMinb=zeros(size(GradeConstMin,1),1);
Scheduling.RHSides.b2_GradeConstMinb = GradeConstMinb;
save('Scheduling','Scheduling')
end
```

**C5) Create mining capacity constraints.**

```
function S4_Const_Capacity
load Scheduling.mat
load LevelOptimization.mat
T =Scheduling.InputParameters.T;
MinCap=Scheduling.InputParameters.MinCap;
MaxCap=Scheduling.InputParameters.MaxCap;
StVal=Scheduling.InputParameters.StVal;
StTon=Scheduling.InputParameters.StTon;
NumOfStops=Scheduling.InputParameters.NumOfStops;
NumOfPeriods=Scheduling.InputParameters.NumOfPeriods;
NumOfLevels=length(find(LevelOptimization.Result.x));
CapacityConstMax1 = zeros(NumOfPeriods,NumOfStops*NumOfPeriods);
for rowloop = 1:NumOfPeriods
    for tloop = 1:NumOfPeriods
      for sloop = 1:NumOfStops
         if    rowloop == tloop
    CapacityConstMax1(rowloop,(((tloop-1)*NumOfStops)+sloop)) = StTon(sloop);
          end
      end
    end
end
CapacityConstMax2=zeros(NumOfPeriods,NumOfLevels*NumOfPeriods);
CapacityConstMax=[ CapacityConstMax1  CapacityConstMax2];
Scheduling.Constraints.A3_CapacityConstMax= CapacityConstMax;
CapacityConstMaxb=MaxCap;
Scheduling.RHSides.b3_CapacityConstMaxb= CapacityConstMaxb;
CapacityConstMin1 = zeros(NumOfPeriods,NumOfStops*NumOfPeriods);
for rowloop = 1:NumOfPeriods
    for tloop = 1:NumOfPeriods
      for sloop = 1:NumOfStops
         if    rowloop == tloop
    CapacityConstMin1(rowloop,(((tloop-1)*NumOfStops)+sloop)) = (-
1).*StTon(sloop);
          end
      end
    end
end
CapacityConstMin2=zeros(NumOfPeriods,NumOfLevels*NumOfPeriods);
CapacityConstMin=[ CapacityConstMin1  CapacityConstMin2];
Scheduling.Constraints.A4_CapacityConstMin= CapacityConstMin;
CapacityConstMinb=(-1).*MinCap;
Scheduling.RHSides.b4_CapacityConstMinb= CapacityConstMinb;
save('Scheduling','Scheduling')
end
```

**C6) Create only one time mining constraints.**

```
function S5_Cons_onlyonce
load Scheduling.mat
load LevelOptimization.mat
NumOfLevels=length(find(LevelOptimization.Result.x));
T =Scheduling.InputParameters.T;
StTon=Scheduling.InputParameters.StTon;
NumOfStops=Scheduling.InputParameters.NumOfStops;
NumOfPeriods=Scheduling.InputParameters.NumOfPeriods;
OnlyonceConst1 = zeros(NumOfStops,NumOfStops*NumOfPeriods);
for rowloop = 1:NumOfStops
    for tloop = 1:NumOfPeriods
      for sloop = 1:NumOfStops
         if    rowloop == sloop
    OnlyonceConst1(rowloop,(((tloop-1)*NumOfStops)+sloop)) = 1;
         end
      end
    end
end
OnlyonceConst2=zeros(NumOfStops,NumOfLevels*NumOfPeriods);
OnlyonceConst=[OnlyonceConst1 OnlyonceConst2];
Scheduling.Aeq.OnlyonceConst= OnlyonceConst;
OnlyonceConstb=ones(size(OnlyonceConst,1),1);
Scheduling.beq.OnlyonceConstb= OnlyonceConstb;
save('Scheduling','Scheduling')
end
```

**C7) Generate stopes adjacent constraints.**

```
function S6_Const_adjucent
load('Output','Output');
load('Result','Result');
load('DataBase','DataBase');
load('Scheduling','Scheduling');
load LevelOptimization.mat
T =Scheduling.InputParameters.T;
StVal=Scheduling.InputParameters.StVal;
NumOfStops=Scheduling.InputParameters.NumOfStops;
NumOfPeriods=Scheduling.InputParameters.NumOfPeriods;
SelectedStopes=Scheduling.InputParameters.SelectedStopes;
LenX=DataBase.GetDim.LenX;
LenY=DataBase.GetDim.LenY;
LenZ=DataBase.GetDim.LenZ;
NumBlockInOneStope=LenX*LenY*LenZ;
BlocksinBestCombination=Output.BlocksinBestCombination;
TotalBlocks=DataBase.Excel.BN;
X=DataBase.Excel.XI;
Y=DataBase.Excel.YI;
Z=DataBase.Excel.ZI;
SelectedBlocks=reshape(BlocksinBestCombination,[NumBlockInOneStope,NumOfStops
]);
NumOfSelectedBlocks=length(BlocksinBestCombination);
NumOfTotalBlocks=length(TotalBlocks);
NumOfLevels=length(find(LevelOptimization.Result.x));
MinAcceptableBlockDistance=sqrt(3);
XIVector = single(X);
YIVector = single(Y);
ZIVector = single(Z);
clear 'X' 'Y' 'Z';
TempX1 = XIVector(:,ones(NumOfTotalBlocks,1));
TempX2 = TempX1';
TempY1 = YIVector(:,ones(NumOfTotalBlocks,1));
TempY2 = TempY1';
TempZ1 = ZIVector(:,ones(NumOfTotalBlocks,1));
TempZ2 = TempZ1';
BlockDistancesmatrix  = sqrt((TempX1-TempX2) .^ 2 + (TempY1-TempY2) .^
2+(TempZ1-TempZ2) .^ 2);
clear 'TempX1' 'TempX2' 'TempY1' 'TempY2' 'TempZ1' 'TempZ2';
AdjucentBlocks = BlockDistancesmatrix < MinAcceptableBlockDistance;
clear 'BlockDistancesmatrix';
 AdjucentStopes = zeros(NumOfStops,NumOfStops);
for iLoop = 1:NumOfStops
  for jLoop = iLoop:NumOfStops
    AdjucentStopes(iLoop,jLoop)=
max(max(AdjucentBlocks(SelectedBlocks(:,iLoop),SelectedBlocks(:,jLoop)')));
    AdjucentStopes(jLoop,iLoop)=AdjucentStopes(iLoop,jLoop);
  end
end
clear 'AdjucentBlocks';
AdjucentStopesBasicCell=zeros(NumOfStops*NumOfStops,NumOfStops);
for rowloop = 1
     for Rowloop = 1:NumOfStops
          for Colloop = 1:NumOfStops
                if AdjucentStopes(Rowloop,Colloop)==1
                       AdjucentStopesBasicCell(rowloop,Rowloop)=1;
```

```
                    AdjucentStopesBasicCell(rowloop,Colloop)=1;
                end
                rowloop=rowloop+1;
            end
        end
end
AdjucentStopesBasicCell( all(~AdjucentStopesBasicCell,2), : ) = [];
EachStopeWithItself = (sum(AdjucentStopesBasicCell, 2) == 1);
AdjucentStopesBasicCell(EachStopeWithItself,:) = [];
AdjucentStopesCell = repmat({AdjucentStopesBasicCell},1,NumOfPeriods);
AdjucentConst1= blkdiag(AdjucentStopesCell{:});
NumOfAdjucent=length(AdjucentConst1);
AdjucentConst2=zeros(NumOfAdjucent,NumOfLevels*NumOfPeriods);
AdjucentConst=[AdjucentConst1 AdjucentConst2];
Scheduling.Constraints.A5_AdjucentConst= sparse(AdjucentConst);
AdjucentConstb=ones(size(AdjucentConst,1),1);
Scheduling.RHSides.b5_AdjucentConst= AdjucentConstb;
save('Scheduling','Scheduling')
save('Output','Output')
end
```

**C8) Generate the connection between levels and their stopes.**

```
function  S7_Cons_ConnectionStopsAndLayer
load Scheduling.mat
load LevelOptimization.mat
load DataBase.mat
Z=Scheduling.InputParameters.CordinatePoint.Z;
NumOfStops=Scheduling.InputParameters.NumOfStops;
NumOfPeriods=Scheduling.InputParameters.NumOfPeriods;
NumOfLevels=length(find(LevelOptimization.Result.x));
LenZ=DataBase.GetDim.LenZ;
FirstPlaceINZ=min(Z);
ZBasedONPlace=zeros(NumOfStops,1);
for  i=1:NumOfStops
     n=(((Z(i,1)-FirstPlaceINZ)/LenZ)+1);
     ZBasedONPlace(i,1)=n;
end
ConnectionStopesAndLayer1= zeros(NumOfLevels,NumOfStops);
for  iloop=1:NumOfStops
     Level=ZBasedONPlace(iloop,1);
     ConnectionStopesAndLayer1(Level,iloop)=1;
     iloop=iloop+1;
end
ConnectionConst1 = zeros(NumOfLevels*NumOfPeriods,NumOfStops*NumOfPeriods);
ConnectionCons1Cell = repmat({ConnectionStopesAndLayer1},1,NumOfPeriods);
ConnectionConst1= blkdiag(ConnectionCons1Cell{:});
ConnectionStopesAndLayer2= zeros(NumOfLevels,NumOfLevels);
NumberOFStopesINLevel=sum(ConnectionStopesAndLayer1');
for jloop=1:NumOfLevels
    ConnectionStopesAndLayer2(jloop,jloop)= (NumberOFStopesINLevel(1,jloop)*-
1);
    jloop=jloop+1;
end
  ConnectionConst2 = zeros(NumOfLevels,NumOfLevels);
  ConnectionCons1Cell2 = repmat({ConnectionStopesAndLayer2},1,NumOfPeriods);
  ConnectionConst2= blkdiag(ConnectionCons1Cell2{:});
ConnectionConst = [ConnectionConst1 ConnectionConst2];
Scheduling.Constraints.A6_ConnectionConst= ConnectionConst;
ConnectionConstb=zeros(size(ConnectionConst,1),1);
Scheduling.RHSides.b6_ConnectionConst= ConnectionConstb;
save('Scheduling','Scheduling')
end
```

**C9) Create concurrent levels constraint.**

```
function S8_Cons_ConcurrentLayers
load Scheduling.mat
load LevelOptimization.mat
NumOfPeriods=Scheduling.InputParameters.NumOfPeriods;
NumOfLevels=length(find(LevelOptimization.Result.x));
NumOfStops=Scheduling.InputParameters.NumOfStops;
NumberOfConcurrentLayers=Scheduling.InputParameters.NumberOfConcurrentLayers;
ConcurrentLayersCons1=zeros(NumOfPeriods,NumOfStops*NumOfPeriods);
AllLayesr=ones(1,NumOfLevels);
ConcurrentLayersCons2 = zeros(NumOfPeriods,NumOfLevels*NumOfPeriods);
ConcurrentLayers1Cell = repmat({AllLayesr},1,NumOfPeriods);
ConcurrentLayersCons2= blkdiag(ConcurrentLayers1Cell{:});
ConcurrentLayersCons = [ConcurrentLayersCons1 ConcurrentLayersCons2];
Scheduling.Constraints.A7_ConcurrentLayersCons= ConcurrentLayersCons;
ConcurrentLayersConsb=(NumberOfConcurrentLayers*ones(NumOfPeriods,1));
Scheduling.RHSides.b7_ConcurrentLayersCons= ConcurrentLayersConsb;
save('Scheduling','Scheduling')
end
```

**C10) Create delay between levels constraint.**

```
function S9_Const_DelayLayers
load Scheduling.mat
load LevelOptimization.mat
NumOfPeriods=Scheduling.InputParameters.NumOfPeriods;
NumOfLevels=length(find(LevelOptimization.Result.x));
NumOfStops=Scheduling.InputParameters.NumOfStops;
DelayBetweenLayers=Scheduling.InputParameters.DelayBetweenLayers;
DelayLayers1=zeros(NumOfLevels*NumOfPeriods,NumOfStops*NumOfPeriods);
DelayLayers2=zeros(NumOfLevels*NumOfPeriods,NumOfLevels*NumOfPeriods);
DelayLayers2_1=zeros(NumOfLevels*NumOfPeriods,NumOfLevels*NumOfPeriods);
LeftSide=zeros(NumOfLevels,NumOfLevels);
  for iloop=1:NumOfLevels
    for    jloop=1:NumOfLevels
        if iloop==jloop
        LeftSide(iloop,jloop)=1;
        end
        if iloop==NumOfLevels && jloop==NumOfLevels
        LeftSide(iloop,jloop)=0 ;
        end
    end
  end
DelayLayersCell2_1 = repmat({LeftSide},1,NumOfPeriods);
DelayLayers2_1= blkdiag(DelayLayersCell2_1{:});
RightSide=zeros(NumOfLevels,NumOfLevels);
  for iloop=1:NumOfLevels
    for    jloop=1:NumOfLevels
        if iloop+1==jloop
        RightSide(iloop,jloop)=-1;
        end
        if iloop==NumOfLevels && jloop==NumOfLevels
        RightSide(iloop,jloop)=0 ;
        end
    end
  end
DelayLayersCell2_2=zeros(NumOfPeriods,NumOfPeriods);
for i=(DelayBetweenLayers+1):NumOfPeriods
    for j=1:(i-DelayBetweenLayers)
        DelayLayersCell2_2(i,j)=1;
    end
end
DelayLayers2_2=kron(DelayLayersCell2_2,RightSide);
DelayLayers2 = plus(DelayLayers2_1,DelayLayers2_2);
DelayLayers = [DelayLayers1 DelayLayers2];
Scheduling.Constraints.A8_DelayLayersCons= DelayLayers;
DelayLayersConsb=zeros(size(DelayLayers,1),1);
Scheduling.RHSides.b8_DelayLayersCons= DelayLayersConsb;
save('Scheduling','Scheduling')
end
```

**C11) Run the optimization model.**

```
function S10_run_scheduling
load Scheduling.mat
f = Scheduling.f;
A1= Scheduling.Constraints.A1_GradeConstMax;
A2= Scheduling.Constraints.A2_GradeConstMin;
A3= Scheduling.Constraints.A3_CapacityConstMax;
A4= Scheduling.Constraints.A4_CapacityConstMin;
A5= Scheduling.Constraints.A5_AdjucentConst;
A6= Scheduling.Constraints.A6_ConnectionConst;
A7= Scheduling.Constraints.A7_ConcurrentLayersCons;
A8=Scheduling.Constraints.A8_DelayLayersCons;
b1= Scheduling.RHSides.b1_GradeConstMaxb;
b2= Scheduling.RHSides.b2_GradeConstMinb;
b3= Scheduling.RHSides.b3_CapacityConstMaxb;
b4= Scheduling.RHSides.b4_CapacityConstMinb;
b5= Scheduling.RHSides.b5_AdjucentConst;
b6= Scheduling.RHSides.b6_ConnectionConst;
b7= Scheduling.RHSides.b7_ConcurrentLayersCons;
b8=Scheduling.RHSides.b8_DelayLayersCons;
Aeq= Scheduling.Aeq.OnlyonceConst;
beq= Scheduling.beq.OnlyonceConstb;
A=[A1;A2;A3;A4;A5;A6;A7;A8];
b=[b1;b2;b3;b4;b5;b6;b7;b8];
addpath 'C:\Program Files\IBM\ILOG\CPLEX_Studio1271\cplex\matlab\x64_win64';
options = cplexoptimset;
options.mip.tolerances.mipgap=0.06;
options.Display = 'on';
[x, fval, exitflag, output] = cplexbilp (f, A, b,Aeq,beq,[],options);
fprintf ('\nSolution status = %s\n', output.cplexstatusstring);
fprintf ('Solution value = %d\n', fval);
Scheduling.Result.x = x;
Scheduling.Result.fval = fval;
rmpath 'C:\Program Files\IBM\ILOG\CPLEX_Studio1271\cplex\matlab\x64_win64';
save('Scheduling','Scheduling')
end
```

**C12) Plot the result.**

```
function Plot
load('Scheduling','Scheduling');
load('DataBase','DataBase');
LenX= DataBase.GetDim.LenX;
LenY= DataBase.GetDim.LenY;
LenZ= DataBase.GetDim.LenZ;
Origin_X=Scheduling.InputParameters.CordinatePoint.X ;
Origin_y=Scheduling.InputParameters.CordinatePoint.Y;
Origin_Z=Scheduling.InputParameters.CordinatePoint.Z;
figure(1)
EleminateLevelPart=Scheduling.Result.x(1:2046);
SelectedStopesinLevel=reshape(EleminateLevelPart,[93 22]);
[~,m]=max(SelectedStopesinLevel,[],2);
Cl = [0 112 192;255 255 0;255 0 0;255 0 255;146 208 80;255 192 0;248 203
173;128 96 0;192 0 0;204 102 255;251 251 183;117 113 113;0 128 0;112 48 160;0
176 240;55 86 35;0 32 96;244 210 250;189 215 238;0 0 0;0 255 255;102 255 51]
/ 255;
 for iLoop=1:length(Origin_X)
        plotcube([LenX LenY LenZ],[Origin_X(iLoop) Origin_y(iLoop)
Origin_Z(iLoop)],.8,Cl(m(iLoop),:));
end
set(gca,'fontsize',11)
ax = gca;
ax.ZDir = 'reverse';
XL = ax.XLim;
YL = ax.YLim;
ZL = ax.ZLim;
LimMin = min([XL(1),YL(1),ZL(1)]);
LimMax = max([XL(2),YL(2),ZL(2)]);
axis equal
xlabel('Easting-X')
ylabel('Northing-Y')
zlabel('Elevation-Z')
end
```

# APPENDIX D

This section includes codes of the presented models for production scheduling optimization based on the levels (SOL). The order of these codes is as followed.

- D1) Generate the objective function, constraints and Runs the optimization model.
- D2) Plot the result.

**D1) Generate the objective function, constraints and Runs the optimization model.**

```
function Scheduling_inEachLevel
load Output.mat
load Result.mat
load Scheduling.mat
load DataBase.mat
[f,p]= uigetfile('*.xlsx');
pf= [p,f];
[num,txt,raw]=xlsread(pf);
Scheduling.InputParameters.MinG = xlsread(pf,'SchedulingData','MinG');
Scheduling.InputParameters.MaxG = xlsread(pf,'SchedulingData','MaxG');
Scheduling.InputParameters.MinC = xlsread(pf,'SchedulingData','MinC');
Scheduling.InputParameters.MaxC = xlsread(pf,'SchedulingData','MaxC');
Scheduling.InputParameters.Rec = xlsread(pf,'SchedulingData','Rec');
Scheduling.InputParameters.IRR = xlsread(pf,'SchedulingData','IRR');
Scheduling.InputParameters.Period = xlsread(pf,'SchedulingData','t');
Scheduling.InputParameters.NumberOfPreiod =
xlsread(pf,'SchedulingData','Nop');
Scheduling.InputParameters.NumberOfLevel =
xlsread(pf,'SchedulingData','NoL');
MinGra=Scheduling.InputParameters.MinG;
MaxGra=Scheduling.InputParameters.MaxG;
MinCap=Scheduling.InputParameters.MinC;
MaxCap=Scheduling.InputParameters.MaxC;
I=Scheduling.InputParameters.IRR;
R=Scheduling.InputParameters.Rec;
NumLevel=Scheduling.NumLevel;
LenX=DataBase.GetDim.LenX;
LenY=DataBase.GetDim.LenY;
LenZ=DataBase.GetDim.LenZ;
SelectedStopes=Output.StopesinBestCombination;
NumberOfPreiod=Scheduling.InputParameters.NumberOfPreiod;
NumBlockInOneStope=LenX*LenY*LenZ;
TotalBlocks=DataBase.Excel.BN;
X=DataBase.Excel.XI;
Y=DataBase.Excel.YI;
Z=DataBase.Excel.ZI;
NumberOfPreiod=Scheduling.InputParameters.NumberOfPreiod;
Scheduling.CordinatePoint=repmat(struct('Xin',[],'Yin',[],'Zin',[]),[NumLevel
1]);
for iLoop=1:NumLevel
    Output(iLoop).NumberOfPreiod=NumberOfPreiod(iLoop);
end
%AdjucentBlocks
NumOfTotalBlocks=length(TotalBlocks);
MinAcceptableBlockDistance=sqrt(2);
XIVector = single(X);
YIVector = single(Y);
ZIVector = single(Z);
TempX1 = XIVector(:,ones(NumOfTotalBlocks,1));
TempX2 = TempX1';
TempY1 = YIVector(:,ones(NumOfTotalBlocks,1));
TempY2 = TempY1';
TempZ1 = ZIVector(:,ones(NumOfTotalBlocks,1));
TempZ2 = TempZ1';
```

```
BlockDistancesmatrix  = sqrt((TempX1-TempX2) .^ 2 + (TempY1-TempY2) .^
2+(TempZ1-TempZ2) .^ 2);
AdjucentBlocks = BlockDistancesmatrix < MinAcceptableBlockDistance;
% create stope information
NextStartPeriod=0;
for Lloop=NumLevel:-1:1
NumberOfPreiod=Output(Lloop).NumberOfPreiod;
StVal=Output(Lloop).ValueinBestCombination;
SelectedStopes=Output(Lloop).StopesinBestCombination;
StTon=Output(Lloop).TonnageinBestCombination;
StGra=Output(Lloop).GradeinBestCombination;
NumOfStops= length(StVal);
NumberOfPreiod=Output(Lloop).NumberOfPreiod;
BlocksinBestCombination=Output(Lloop).BlocksinBestCombination;
StopeXCordinatePoint=[zeros(size(SelectedStopes))]';
StopeYCordinatePoint=[zeros(size(SelectedStopes))]';
StopeZCordinatePoint=[zeros(size(SelectedStopes))]';
   for iloop=1:NumOfStops
     StopeXCordinatePoint(iloop)=
min([DataBase.Excel.XI(SelectedStopes(iloop).PSBN)]);
     StopeYCordinatePoint(iloop)=
min([DataBase.Excel.YI(SelectedStopes(iloop).PSBN)]);
     StopeZCordinatePoint(iloop)=
min([DataBase.Excel.ZI(SelectedStopes(iloop).PSBN)]);
     Scheduling.CordinatePoint(Lloop).Xin=StopeXCordinatePoint;
     Scheduling.CordinatePoint(Lloop).Yin=StopeYCordinatePoint;
     Scheduling.CordinatePoint(Lloop).Zin=StopeZCordinatePoint;
   end
Xin=(Scheduling.CordinatePoint(Lloop).Xin)';
Yin=(Scheduling.CordinatePoint(Lloop).Yin)';
Zin=(Scheduling.CordinatePoint(Lloop).Zin)';
Output(Lloop).coordinate.Xin =Xin;
Output(Lloop).coordinate.Yin =Yin;
Output(Lloop).coordinate.Zin =Zin;
% Create Objective Function for Each Level
ObjectiveFunction = zeros(1,NumOfStops*NumberOfPreiod);
for tloop=1:NumberOfPreiod
      for sloop = 1:NumOfStops
    ObjectiveFunction(1,(((tloop-1)*NumOfStops)+sloop)) =
StVal(sloop)/((1+I)^(tloop+NextStartPeriod-1));
      end
 end
Output(Lloop).ObjectiveFunction =(-1).*(ObjectiveFunction);
%Create Const Max & Min Grade
A_MaxGrade = zeros(NumberOfPreiod,NumOfStops*NumberOfPreiod);
for rowloop = 1:NumberOfPreiod
    for tloop = 1:NumberOfPreiod
      for sloop = 1:NumOfStops
         if    rowloop == tloop
    A_MaxGrade(rowloop,(((tloop-1)*NumOfStops)+sloop)) =
StTon(sloop).*((StGra(sloop))-(MaxGra(tloop))).*(R(tloop))';
         end
      end
    end
end
b_MaxGrade=zeros(size(A_MaxGrade,1),1);
A_MinGrade = zeros(NumberOfPreiod,NumOfStops*NumberOfPreiod);
```

```
for rowloop = 1:NumberOfPreiod
    for tloop = 1:NumberOfPreiod
      for sloop = 1:NumOfStops
         if    rowloop == tloop
               A_MinGrade(rowloop,(((tloop-1)*NumOfStops)+sloop)) =(-1).*(
StTon(sloop).*((StGra(sloop))-(MinGra(tloop))).*(R(tloop))') ;
         end
      end
    end
end
b_MinGrade=zeros(size(A_MinGrade,1),1);
Output(Lloop).A_MaxGrade = A_MaxGrade;
Output(Lloop).b_MaxGrade =b_MaxGrade;
Output(Lloop).A_MinGrade =A_MinGrade;
Output(Lloop).b_MinGrade =b_MinGrade;
%Create Const Max & Min Capacity
A_MaxCapacity = zeros(NumberOfPreiod,NumOfStops*NumberOfPreiod);
for rowloop = 1:NumberOfPreiod
    for tloop = 1:NumberOfPreiod
      for sloop = 1:NumOfStops
         if    rowloop == tloop
    A_MaxCapacity(rowloop,(((tloop-1)*NumOfStops)+sloop)) = StTon(sloop);
          end
      end
    end
end
b_MaxCapacity=MaxCap(1:NumberOfPreiod);
A_MinCapacity = zeros(NumberOfPreiod,NumOfStops*NumberOfPreiod);
for rowloop = 1:NumberOfPreiod
    for tloop = 1:NumberOfPreiod
      for sloop = 1:NumOfStops
         if    rowloop == tloop
    A_MinCapacity(rowloop,(((tloop-1)*NumOfStops)+sloop)) = (-
1).*StTon(sloop);
          end
      end
    end
end
b_MinCapacity=(-1).*MinCap(1:NumberOfPreiod);
Output(Lloop).A_MaxCapacity = A_MaxCapacity;
Output(Lloop).b_MaxCapacity =b_MaxCapacity;
Output(Lloop).A_MinCapacity =A_MinCapacity;
Output(Lloop).b_MinCapacity =b_MinCapacity;
% Create Const Only Once
Aeq_Onlyonce = zeros(NumOfStops,NumOfStops*NumberOfPreiod);
for rowloop = 1:NumOfStops
    for tloop = 1:NumberOfPreiod
      for sloop = 1:NumOfStops
         if    rowloop == sloop
    Aeq_Onlyonce(rowloop,(((tloop-1)*NumOfStops)+sloop)) = 1;
         end
      end
    end
end
beq_Onlyonce=ones(size(Aeq_Onlyonce,1),1);
Output(Lloop).Aeq_Onlyonce = Aeq_Onlyonce;
Output(Lloop).beq_Onlyonce =beq_Onlyonce;
```

```
SelectedBlocks=reshape(BlocksinBestCombination,[NumBlockInOneStope,NumOfStops
]);
NumOfSelectedBlocks=length(BlocksinBestCombination);
AdjucentStopes = zeros(NumOfStops,NumOfStops);
for iLoop = 1:NumOfStops
  for jLoop = iLoop:NumOfStops
    AdjucentStopes(iLoop,jLoop)=
max(max(AdjucentBlocks(SelectedBlocks(:,iLoop),SelectedBlocks(:,jLoop)')));
    AdjucentStopes(jLoop,iLoop)=AdjucentStopes(iLoop,jLoop);
  end
end
   AdjucentStopesBasicCell=zeros(NumOfStops*NumOfStops,NumOfStops);
for rowloop = 1
     for Rowloop = 1:NumOfStops
          for Colloop = 1:NumOfStops
               if AdjucentStopes(Rowloop,Colloop)==1
                    AdjucentStopesBasicCell(rowloop,Rowloop)=1;
                    AdjucentStopesBasicCell(rowloop,Colloop)=1;
               end
               rowloop=rowloop+1;
          end
     end
end
AdjucentStopesBasicCell( all(~AdjucentStopesBasicCell,2), : ) = [];
EachStopeWithItself = (sum(AdjucentStopesBasicCell, 2) == 1);
AdjucentStopesBasicCell(EachStopeWithItself,:) = [];
AdjucentStopesCell = repmat({AdjucentStopesBasicCell},1,NumberOfPreiod);
A_Adjucent= blkdiag(AdjucentStopesCell{:});
b_Adjucent=ones(size(A_Adjucent,1),1);
Output(Lloop).A_Adjucent = A_Adjucent;
Output(Lloop).b_Adjucent =b_Adjucent;
% Run Scheduling for Each Level
f=Output(Lloop).ObjectiveFunction;
A=[A_MaxGrade;A_MinGrade;A_MaxCapacity;A_MinCapacity;A_Adjucent];
b=[b_MaxGrade;b_MinGrade;b_MaxCapacity;b_MinCapacity;b_Adjucent];
Aeq=Aeq_Onlyonce;
beq=beq_Onlyonce;
addpath 'C:\Program Files\IBM\ILOG\CPLEX_Studio1271\cplex\matlab\x64_win64';
options = cplexoptimset;
options.Display = 'on';
[x, fval, exitflag, output] = cplexbilp (f, A, b,Aeq,beq,[],options);
fprintf ('\nSolution status = %s\n', output.cplexstatusstring);
fprintf ('Solution value = %d\n', fval);
Output(Lloop).Result_x = x;
Output(Lloop).Result_fval = fval;
rmpath 'C:\Program Files\IBM\ILOG\CPLEX_Studio1271\cplex\matlab\x64_win64';
NextStartPeriod=NextStartPeriod+NumberOfPreiod;
save('Output','Output')
save('Scheduling','Scheduling')
end
end
```

**D2) Plot the result.**

```
function Plot
load('Scheduling','Scheduling');
load('Output','Output');
load('DataBase','DataBase');
NumLevel=Scheduling.NumLevel;
LenX= DataBase.GetDim.LenX;
LenY= DataBase.GetDim.LenY;
LenZ= DataBase.GetDim.LenZ;
    for Lloop=5
 figure
Origin_X=Scheduling.CordinatePoint(Lloop).Xin ;
Origin_y=Scheduling.CordinatePoint(Lloop).Yin;
Origin_Z=Scheduling.CordinatePoint(Lloop).Zin;
figure(1)
 selstope=Output(Lloop).Result_x;
SelectedStopesinLevel=reshape (selstope,[18 4]);
[~,m]=max(SelectedStopesinLevel,[],2);
Cl = [0 112 192;255 255 0;255 0 0;255 0 255] / 255;
for iLoop=1:length(Origin_X)
    plotcube([LenX LenY LenZ],[Origin_X(iLoop) Origin_y(iLoop)
Origin_Z(iLoop)],.8,Cl(m(iLoop),:));
set(gca,'fontsize',10)
ax = gca;
ax.ZDir = 'reverse';
XL = ax.XLim;
YL = ax.YLim;
ZL = ax.ZLim;
LimMin = min([XL(1),YL(1),ZL(1)]);
LimMax = max([XL(2),YL(2),ZL(2)]);
axis equal
xlabel('Easting-X')
ylabel('Northing-Y')
zlabel('Elevation-Z')
title('L5')
end
```