OPEN–PIT MINE PRODUCTION SCHEDULING USING THE GENETIC ALGORITHM

Pathy Musema Muke

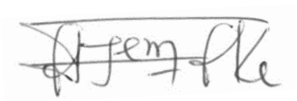
A research report submitted to the Faculty of Engineering and the Built Environment, University of the Witwatersrand, Johannesburg, in partial fulfilment of the requirements for the Degree of Master of Science in Engineering.

Johannesburg, 2020

DECLARATION

I declare that this report is my own, unaided work. I have read the University Policy on Plagiarism and hereby confirm that no plagiarism exists in this report. I also confirm that there is no copying nor is there any copyright infringement. I willingly submit to any investigation in this regard by the School of Mining Engineering and I undertake to abide by the decision of any such investigation.

Signed:



Pathy Musema Muke

This 08 day of December 2020

ABSTRACT

The determination of an optimum open-pit mine production scheduling (OMPS) plan is an important part of the planning of any open-pit mine. The related optimisation problem for OMPS is an intractable problem. This is because it involves large datasets and, multiple hard and soft constraints, which make it a large combinatorial optimisation problem. Therefore, the use of deterministic techniques has demonstrated their limitation towards solving the OMPS problem. This impracticability has led to the use of metaheuristic techniques suitable to handle problems of such a nature. In order to solve the OMPS long-term problem, this research study applied the genetic algorithm (GA), which is regarded as an evolutionary technique to address difficult and complex problems in a much-reduced amount of time. The concern of the OMPS problem is to find the best extraction layout that generates the maximum net present value (NPV) of the project, out of thousands of feasible options.

A proposed GA model was developed and coded in Python programming environment to solve the OMPS optimisation problem. Python is an interpreted, interactive and object-oriented programming language and is one of the most used in data mining and machine learning. The research demonstrated that the GA run in Python, can be effectively used to find a near-optimal mine production schedule for small- and large-scale block models in three-dimensional (3D) space. The GA algorithm was applied on two mining case studies, namely: Newman1 and Zuck Small, which were downloaded from MineLib datasets that are freely available in the public domain for research purposes. The production plans obtained using the GA algorithm defined the appropriate time at which each block should be extracted from the mine on a long-term horizon. In order to validate the proposed GA model, a numerical result comparison was done against the best-known solutions provided in MineLib.

The two OMPS long-term plans generated for the two case studies yielded NPVs, which were by about 7.0% and 8.3%, respectively compared to the best-known optimal feasible solutions obtained using the TopoSort approach. Although the proposed GA algorithm demonstrated its efficiency to deal with the OMPS long-

term problem, further research such as applying the proposed GA on a real-world active open-pit mine needs to be conducted in order to improve and generalise the GA to other optimisation problems in mining.

ACKNOWLEDGEMENTS

Above everything, I am so grateful to my Lord, the Creator of the Universe for granting me wisdom and knowledge to achieve this degree and for guiding me in this scientific adventure, which I undertook. However, different individuals and organisations contributed towards the realisation of the research study, to whom I would like to extend the following special acknowledgements:

- My supervisors Mr Sihesenkosi Nhleko and Professor Cuthbert Musingwini (Head- School of Mining Engineering, University of the Witwatersrand) for their insightful contributions in initiating this research study. Their academic guidance and advice inspired me to achieve what initially seemed almost impossible;
- The CORY Scholarship Fund through the School of Mining Engineering for providing financial assistance for my studies;
- The School of Mining Engineering staff members and all my postgraduate colleagues for their words of encouragement and support especially in moments of despair;
- The Director of ISTA (Institut Supérieur des Techniques Appliquées), in Kolwezi, Professor Flory Kiseya, and all the staff members for allowing me to pursue this degree;
- Mr Robert Nkateko Ngobeni for his undeniable assistance during the coding session in Python programming language;
- My lovely family for their prayers on my behalf and trusting in me to achieve my master's degree. Here is the fruit of their investment in me; and
- My brother, Professor Moses Mukepe, for his relentless financial support and advice to make me succeed in my studies. I saw a true brotherly love in him.

DEDICATION

This research report is dedicated to my late niece, Gerassy Namulwisa, whose premature death left me heartbroken. I would have liked to share with you this exciting moment of my success.

May your soul rest in peace, Gerassy!

CONTENTS **Page**

LIST OF FIGURES

LIST OF TABLES

1 INTRODUCTION

1.1 Chapter overview

This chapter explains the context of the research topic being conducted and highlights the major operations involved along the mining value chain for the extraction of the Mineral Resource as detailed in Section 1.2. It also presents in Section 1.3 the importance of the research and the problem statement, including details on the relevant constraints that make the problem intractable. Section 1.4 presents the main research question and the nature of the problem as well as the general background on the potential techniques suitable for solving the problem of the research focus. The scope and objectives of the research are outlined in Section 1.5, while the nature of required data and data sources are presented in Section 1.6. Finally, Section 1.7 describes the structure of the remaining chapters of the report.

1.2 Research context

Mining operations consist of extracting mineral deposits of economic interest from the earth's crust by means of blasting and/or excavation activities using surface or underground mining methods (Darling, 2011). Blasting allows breaking the hard rock into small-sized blocks that are easy to handle. The broken ore material is subsequently transported using a well-established transport network to the processing plant for further treatment. Processing operations aim at retrieving the mineral element of economic interest through metallurgical and chemical processes, saleable to the market in order to derive a profit.

Mineral deposits originate from natural accumulations of mineral substances in a region resulting in an ore deposit or orebody, which is economically mineable (Newman *et al*., 2010). The formation of mineral deposits takes millions to billions of years of geological processes in the presence of higher temperatures and higher pressures (Darling, 2011). The types of mineral deposits differ in accordance with the nature of the substance and can be metallic, non-metallic, fossil fuel, liquids and gas resources (Samavati, 2017). In alignment with the research objective, this report considers two case studies of metallic ore deposits of gold extraction by means of surface mining.

Surface mining is a complex and long activity encompassing five major successive stages namely: prospecting, exploration, development, exploitation and reclamation (Newman *et al*., 2010). The activities start with the search of indices of an eventual accumulation of Mineral Resource using visual observation and physical measurement on the surface of a region during the first stage. The second stage, the exploration stage, consists of investigating and collecting below the ground, sample data essential for estimating the quantity and quality of the mineral deposit. The geological data regroups the information related to the lithology formations (rock types), mineral elements and value measurement attributes (e.g. ore grade, specific gravity). The set of data collected in this stage should be sufficient to construct a three-dimensional (3D) model of the mineral deposit and surrounding rocks, for use during mine planning and optimisation studies.

The mine planning process is based on the Mineral Resource, which is considered as the company's primary asset. The economic sustainability of a mining project is highly dependent upon careful mine planning and management of a Mineral Resource subject to continuous evaluation and planning over the life of the mine (Caccetta and Hill, 2003). Mining as a business activity converts Mineral Resources into Mineral Reserves using mining, metallurgical, economic, marketing, legal, environmental, social and governmental factors (Rendu, 2014). A Mineral Reserve represents the portion of the deposit that can be mined at a profit. In open-pit mining, a Mineral Reserve is included in the ultimate pit limits, upon which mine planners develop a production schedule for the extraction of mineral material.

The third stage in the surface mining activities is the development stage, which consists of translating the mine planning and optimisation studies into mine design. The translation proceeds along with the determination of the mining method, equipment selection, production capacity and the construction of facilities (Samavati, 2017). In fact, development operations result in removing overlying material and creating access to the ore material. The fourth stage is the exploitation stage, which involves the mining of the ore from the ground, hauling the broken rock to either the processing plant, stockpile or waste dump. Finally, the reclamation stage consists of ending the mining operations to enable the restoration and rehabilitation of the site.

Between the exploration and the development stages, there exists several optimisation problems with the objective of maximising the value of the mine and minimising the mining cost. These problems include cut-off grade optimisation, pit optimisation, pushback sequencing, stockpiling and recovery optimisation. Most often, optimisation problems are conducted during the mine planning process using operations research (OR) techniques. Therefore, the focus of this research study is to find the optimum production scheduling in order to maximise the net present value (NPV) of the mine. In this regard, the genetic algorithm (GA) was adopted for solving the identified problem, as justified later in Chapter 4.

1.3 Problem statement

Open-pit mine production scheduling (OMPS) consists of developing the extraction sequence of a mineral deposit across the entire life of the mine. It is designed upon portions of the deposit that have been proved to be economically mineable. These valuable portions constitute the Mineral Reserve and are included in the ultimate pit limits of the project. For the purpose of mine planning studies, the mineral deposit is whittled into a regular array of small blocks often of the same size displayed in a 3D space known as an orebody block model. As such, the OMPS plan determines the time at which each block in the ultimate pit should be mined and how a mined block should be treated afterwards (Chicoisne *et al*., 2012). There exists the best time for the extraction of each individual block within the block model to generate the maximum possible profit for the mine. The sooner an ore block is mined, processed and sold, the higher the NPV generated. Hence, the OMPS long-term optimisation determines the best extraction time of each individual block. Additionally, the OMPS plan determines a destination for each block according to the quality (grade) of the block. A block is classified as ore if the grade is equal or greater than the cut-off grade; a block with a grade less than the cut-off grade is considered as waste, which is unable to generate profit. Waste blocks are sent to the waste dump while ore blocks are either stockpiled or sent to the processing plant.

Finding the best sequence for extracting blocks from the pit has never been a straightforward task to resolve. The question has drawn the attention of many researchers since the 1960s. The determination of the optimum sequence is reputed to be complex in the mine planning process. In view of the number of blocks included in the pit that might be in the order of several thousands to millions of blocks, the long-term OMPS is mostly considered as a difficult combinatorial optimisation problem. Moreover, the time horizon of the long-term plan, which forecasts an extraction plan of 5 to 30 years into the future, contributes to the complexity of the problem. The configuration of the block model and the distribution of ore blocks inside induce the following constraints in the modelling of the OMPS long-term plan: precedence block constraints, grade control, mining capacity, processing capacity and reserve capacity. These constraints are discussed in more details in the next sections.

1.3.1 Precedence block constraints

The extraction of each block is dependent upon its position in the orebody block model. Hence, the extraction of blocks that are located on top must precede that of blocks underneath. Figure 1.1 illustrates the number of direct block predecessors of a block respectively in two-dimensional (2D) and 3D block models. The mining of any block must proceed if and only if all blocks above it, have been previously removed. For instance, the mining of Block 1 in 2D (see Figure 1.1 (a)) will require the prior removal of Blocks 2, 3 and 4; while in 3D the mining of Block 1 (see Figure 1.1 (b)) will require the prior removal of Blocks 2, 3, 4, 5 and 6. Therefore, Blocks 2, 3 and 4 constitute the set of block predecessors of Block 1 in (a) and, Blocks 2, 3, 4, 5 and 6 for Block 1 in (b). The determination of the set of block predecessors must comply with the pit slope angle for bench stability keeping the shape of the pit. An optimum OMPS plan is to deliver the best extraction sequence of blocks, while respecting the block precedence constraints that account for physical constraints such as geotechnical constraints on safe pit slope angle.

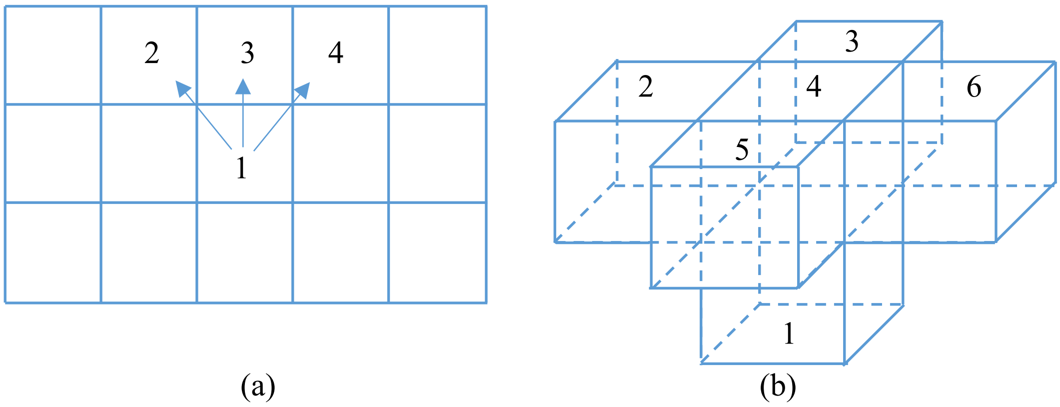


Figure 1.1 Graphical view of block predecessors of Block 1 in 2D (a) and 3D (b) (Source: Gaupp, 2008; Nogholi, 2015)

1.3.2 Grade control for ore blocks

The mineral grade distribution is not homogenous across the deposit. Therefore, it is important to define a cut-off grade that distinguishes ore blocks from waste blocks, because the block model consists of a mixture of ore and waste blocks. This configuration makes the selection and schedule of ore blocks for OMPS a combinatorial optimisation problem. Ore blocks contain amounts of mineral of economic interest equal to or higher than the cut-off grade. In contrast, waste blocks are those that contain less amounts of the mineral of economic interest, which are unable to generate profit at current market conditions. The quality of ore blocks delivered at each period must fit within the required limits. Consequently, the OMPS plan should find the best block sequence combination that satisfies all the requirements.

1.3.3 Reserve control

The total quantity of Mineral Reserve represents the company's asset available for extraction. Each individual block is scheduled once for its extraction in the OMPS plan and must be removed from the model thereafter. The formulation of OMPS should be free of block duplication in the execution of the block extraction sequence. The output of the OMPS plan, in terms of the tonnage, must be equal to the total amount of material (ore and waste) contained in the ultimate pit limits.

1.3.4 Mining capacity

The equipment used for the extraction of blocks determines the mining capacity of the operations set between the minimum and maximum capacities per period. The total tonnage of material (ore and waste blocks) scheduled per period, must meet the predefined production rate of the mine. Each mining period must contain a selection of blocks to mine of which the total tonnage meets the mining capacity requirements.

1.3.5 Processing capacity

The quality of ore in the ultimate pit limits varies from place to place. Nevertheless, from the miner's point of view, a certain fixed quality of ore has to be supplied to the plant. A predesigned processing plant comes with the required quality of material to process. The processing capacity is defined to process material of a certain quality and quantity between the minimum and maximum capacities. The OMPS plan must, therefore, deliver a product of the quality and quantity in accordance with the plant requirements. The surplus ore material is sent to stockpiles for later processing.

1.4 Research question

The presence of precedence block, grade control, reserve control, mining capacity and processing capacity constraints along the mining value chain makes the development of a production schedule plan a more challenging and complex task, whose optimal solution would require the use of advanced techniques. The need for defining an optimal OMPS plan delivering an efficient extraction sequence over the life of mine is at the centre of the mine planning process. Moreover, an optimal production plan is vital for generating the maximum NPV of the project. The OMPS can be mathematically formulated with an objective function of maximising the NPV.

The ultimate pit consists of a mixture of ore blocks and waste blocks in a 3D space. These blocks are distributed unevenly along X, Y and Z directions. The extraction of blocks should not proceed in a random way to maximise the profit that can be obtained from the mine. There should exist a best time for each block to be mined regardless of its quality (grade). Of course, the extraction sequence of blocks must comply with all the constraints, which include precedence block, grade control, reserve control, mining and processing capacity constraints. These constraints initiate the search for the optimum time for each block to be mined from the pit. This raises the question: *"When and how should each block be extracted from the pit in such a way that the NPV of the mine is maximised?"*

Despite having several optimisation techniques that have been applied for solving the OMPS, the problem of modelling an optimum extraction scheduling remains a challenge in the mine planning process (Caccetta and Hill, 2003). The production scheduling problem has mostly been resolved using exact algorithms since the work of Lerchs and Grossmann in 1965. However, the implementation of exact algorithms faces some limitations in dealing with complex problems. For more than two decades, the mining industry has integrated the use of heuristic algorithms, which are inspired by the natural behaviour of species to solve optimisation problems in mining. One of the heuristic algorithms is the genetic algorithm (GA), which is regarded as a good tool in solving difficult and complex problems. This raises the question: "*Can GA generate optimum open-pit mine production schedule?"*

1.5 Research objectives

The main objective of this research was to generate an optimum OMPS long-term plan that maximises the NPV of the mine, under given constraints. The optimum production plan seeks to provide the block sequence in the presence of the identified constraints. To do this, the research makes use of the GA, which is a heuristic algorithm that deals with problems of such nature. The research was motivated by the growing needs of using advanced techniques (heuristic algorithms) that have direct application to the mining industry. However, the application of the GA for the long-term OMPS is to contribute to the small amount of research work done using heuristic approaches in mine planning. More specifically, this research carries the following objectives:

- Apply the GA to an orebody block model (ultimate pit limits) to produce an optimum long-term production schedule for an open-pit mine;

- Discuss the efficiency of the GA in solving the production scheduling problem and assess its practicability in terms of complexity and flexibility of the model in integrating OMPS constraints; and
- Through the identification of the strengths and weaknesses of the GA, recommend areas of improvement for the performance of the GA for future work on OMPS optimisation.

1.6 Input data type

The long-term OMPS is a large and complex problem in mine planning. It involves constraints that affect the mining extraction plan to be optimised. It requires some predefined features ahead of the formulation of the production schedule. Therefore, the input data required for the OMPS long-term plan includes the following:

- A block model description file that contains information relative to the block-ID, block coordinates, ore grade, block tonnage, block economic value;
- Block precedence file containing the direct block predecessors for each block in the model;
- Target minimum and maximum mining capacities defining the amount of material to be mined per time period. This includes ore and waste blocks;
- Target minimum and maximum processing capacities that represent the amount of ore to be supplied to the processing plant; and
- A definition of the duration of time period for the long-term plan in years.

Data input was freely downloaded from MineLib (http://mansci-web.uai.cl/minelib/), which is a publicly available website providing open-pit mine instances for research experiments.

1.7 Outline of chapters

This research report comprises six chapters of which the contents of the next five chapters are outlined below.

Chapter 2 covers the literature review on the traditional open-pit mine planning where optimisation techniques are implemented, and the decision-making process is taking place for the planning of subsequent activities along the mining value chain. The chapter discusses the optimisation techniques for the determination of the optimal cut-off grade and the ultimate pit limits. The chapter also explains the concepts of Mineral Resource and Mineral Reserve, then discusses the geological modelling of the mineral deposit, which is considered an important aspect in solving the open-pit mine production scheduling. The chapter also discusses a generic open-pit mining including the pit geometry and how the pit should be expanded throughout the excavation operation.

Chapter 3 reviews the background work on the optimisation of open-pit mine production scheduling. The production schedule is often conducted over three time-horizons that include long-term, medium-term and short-term plans; while the focus of the report is on the long-term plan. Thus, the chapter develops the mathematical formulation of the long-term production schedule of an open-pit mine. Furthermore, it describes the most popular techniques that have been applied to solve the OMPS problem. These techniques are grouped into two groups namely: deterministic algorithms and heuristic algorithms. Additionally, the chapter describes the available commercial software packages used for the production schedule in mining.

Chapter 4 explains the research methodology adopted in this report to solve the long-term open-pit mine production scheduling. In this chapter, the genetic algorithm is described in detail, followed by a discussion about the genetic algorithm terminology and key concepts and how this is suitable for solving the problem being investigated. Therefore, a proposed genetic algorithm model that adapts with the long-term mine production scheduling in open-pit mining is developed.

Chapter 5 demonstrates the implementation of the genetic algorithm on two case studies freely downloaded from MineLib. The proposed genetic algorithm model is written in Python programming language for testing. The chapter provides a discussion on the results and assesses the compliance of the long-term production plan generated with the required constraints. The validation of the model is given through measuring the results obtained against the benchmark solutions proposed in MineLib using TopoSort Heuristic approach.

Finally, Chapter 6 provides the general conclusion of the research. It briefly restates the matter being investigated and presents the research findings. It suggests some recommendations for future work that will improve the proposed genetic algorithm and impact on the open-pit mine production scheduling problem.

2 LITERATURE REVIEW ON TRADITIONAL OPEN-PIT MINE PLANNING

2.1 Chapter overview

Traditional open-pit mine planning is a system of connected activities that can be broken up and understood in order to assess the importance and efficiency that each activity brings to the system. The interactive nature of the system derives from the use of the outcome of an activity as an input for subsequent activities in a successive and circular way. This chapter briefly provides a broad view of open-pit mine planning and the interconnection of its component tasks towards the overall goal. The aim is to contextualise the production schedule as a subset of the planning process and highlight the major tasks that impact on the realisation of an optimum open-pit mine production schedule. The description of an open-pit mine and pit expansion are also discussed in this chapter.

2.2 Introduction

Mine planning is a series of activities that require decision-making to be made for the realisation of a realistic and feasible plan to profitably extract Mineral Resources from the ground. It is a multi-disciplinary exercise starting with geological data collection and ending up with a financial valuation of the business case. The practice of mine planning involves several activities that include exploration of the mineral deposit, extraction of the mineral of concern and ore processing. Geologists build a geological block model illustrating the geological characteristics of the deposit. Mining engineers take over from the geological block model built by geologists, to proceed with mine design and production plan. Metallurgists receive the run of mine material supplied to them by miners to extract the commodity of economic interest through metallurgical and chemical processes. Financial experts work on project investment, cost and revenue generated from the business. These personnel work together in an integrated mine planning team towards the overall company goal that aims to maximise the value of the deposit.

The execution of planning activities follows a certain directive in which the optimum solutions of early activities might serve as inputs for subsequent activities.

However, solutions obtained from subsequent activities can also be used to adjust the previous decisions made at earlier stages. For example, a change in the cut-off grade evaluation along the mine planning process would trigger the readjustment of the pit limits (Mugwagwa, 2017). Regarding the sequence of tasks occurrence, mine planning activities are conducted at different levels in terms of planning horizons and level of detail on the mineral deposit. Figure 2.1 illustrates a schematic guide for a conventional mine planning process as proposed by Thorley (2012), highlighting the major tasks under the responsibility of mine planners. These planning tasks are executed over three planning horizons namely: strategic planning, tactical planning and operational planning, in an increasing level of confidence in the mineral deposit.

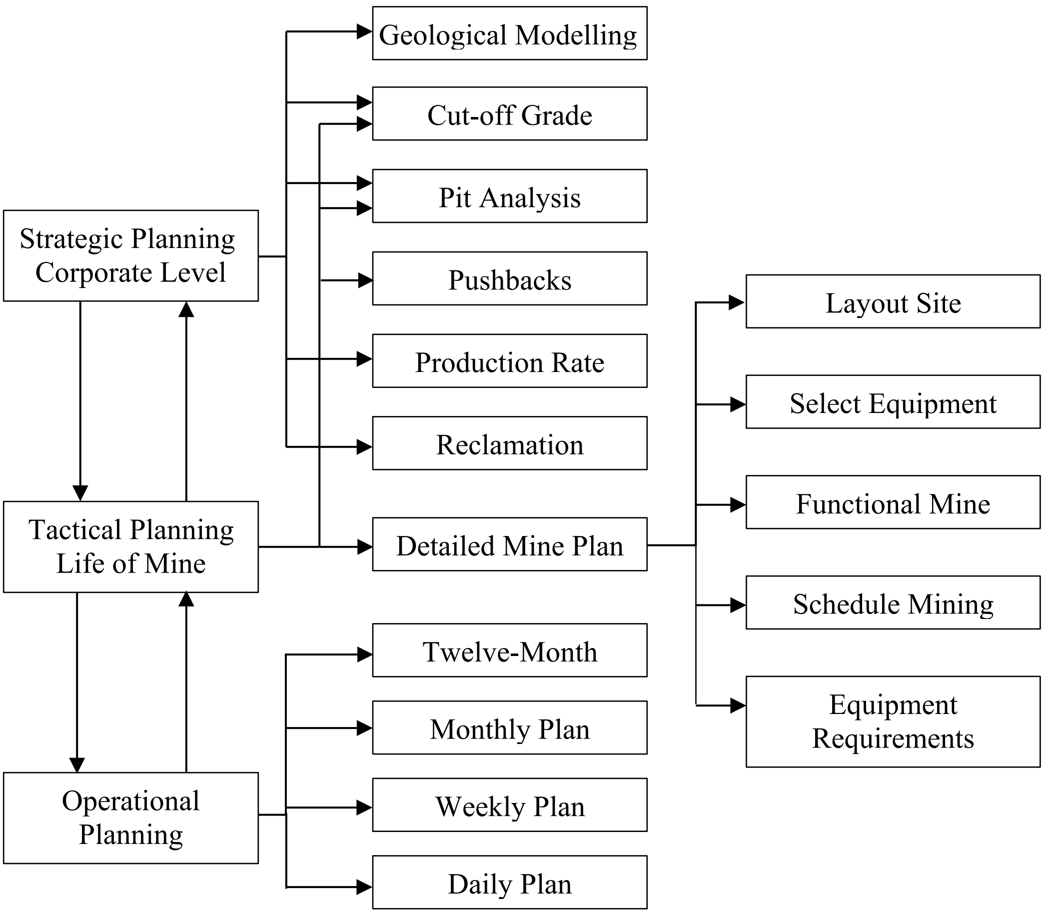


Figure 2.1 Generic open-pit mine planning process (Adapted from: Thorley, 2012)

Strategic planning focusses on a company's long-range objective that stands for the entire mining activities and adjusts the organisation's direction on major activities along the mine value chain. Strategic decisions can be made on the following planning activities: geological modelling, pit optimisation, determination of the production rate, reclamation policy, cut-off grade and setting the direction for mine scheduling (Thorley, 2012). Tactical planning establishes actionable tasks of which results must cover a long-term period in alignment with strategic goals. In addition to the strategic planning activities such as pit analysis and cut-off grade, which are established for a period of more than a year, tactical planning seeks to develop a detailed production plan (Kloppers *et al*., 2015). This plan contains the layout site, select equipment, functional mine design and equipment requirements (Thorley, 2012). Operational planning consists of setting up smaller and intermediate milestones to achieve short-term objectives, which combined meet the overall company goal. On the operational level, the goal is often to get a production plan that stands for twelve months, one month, one week and one day.

Despite the significant advances recorded in the development of optimisation techniques in the open-pit planning process, there exists no approach for global optimisation of the entire open-pit mine planning. Instead, researchers have developed techniques that can optimise sub-tasks individually or co-optimise with one or two other tasks at once. One of these sub-tasks is the long-term open-pit mine production plan, which in some instances can be optimised together with the ultimate pit limit.

The planning of the long-term open-pit mine scheduling aims to find the best mining production plan under constraints imposed by geological and physical conditions and operational mining parameters. Optimisation of long-term production schedule plan involves various steps such as development of geological and economic block models, optimisation of economic pit limits and life of mine production schedule plan. Therefore, the following sub-sections describe the major tasks that precede the establishment of the open-pit mine production schedule.

2.3 Mineral Resource

International standards have been adopted for the definition and reporting of Mineral Resources and Reserves to facilitate communication between mining companies and parties involved in the exploitation of minerals (Rendu and Miskelly, 2001). In 1971, the Joint Ore Reserves Committee (JORC) was founded in order to establish codes on public reporting and ore reserves classification. Over the years, the JORC code has been used as the basis on which, countries like South Africa, Australia, the United Kingdom, Canada and the United States of America have defined the standard classification of exploration results and definition of Mineral Resources and Mineral Reserves (Hustrulid *et al*., 2013).

The South African Mineral Reporting Code (SAMREC, 2016:18) defined a Mineral Resource as “a concentration or occurrence of solid material of economic interest in or on the earth’s crust in such form, quality and quantity that there are reasonable prospects for eventual economic extraction.” The estimation of Mineral Resources is based on the geological evidence and knowledge of sparsely located samples collected from the mineral deposit usually in a specified grid pattern. The location, quality, grade, geological characteristics and continuity of a Mineral Resource must be known or estimated at a reasonable level of confidence (Hustrulid *et al*., 2013). Uncertainty always exists in the evaluation of Mineral Resources as these are always estimated from drill-hole samples. Several methods are used in the estimation of grade and Mineral Resources by providing grade-tonnage curves measuring the potential benefit of extracting the orebody (Newman *et al*., 2010). The most popular estimation methods are Kriging, inverse distance squared and the nearest distance (Ortiz and Emery, 2006). The quality of grade estimation mainly depends upon data spacing used to estimate unmeasured points. The closer the data spacing is, the more confident will be the resource estimation.

Mineral Resources are classified into three distinct categories namely: inferred, indicated and measured in regard to the level of confidence in the estimation measurement. Figure 2.2 illustrates the different categories of Mineral Resources in terms of increased level of geological confidence and knowledge. The level of geological confidence in the inferred resource estimation is not sufficient for the conversion into Mineral Reserves. Nevertheless, the indicated and measured Mineral Resources can be converted into Mineral Reserves as they assure the continuity of the mineralisation. The latter is a vital factor in the validation of the Mineral Reserve (Rendu and Miskelly, 2001).

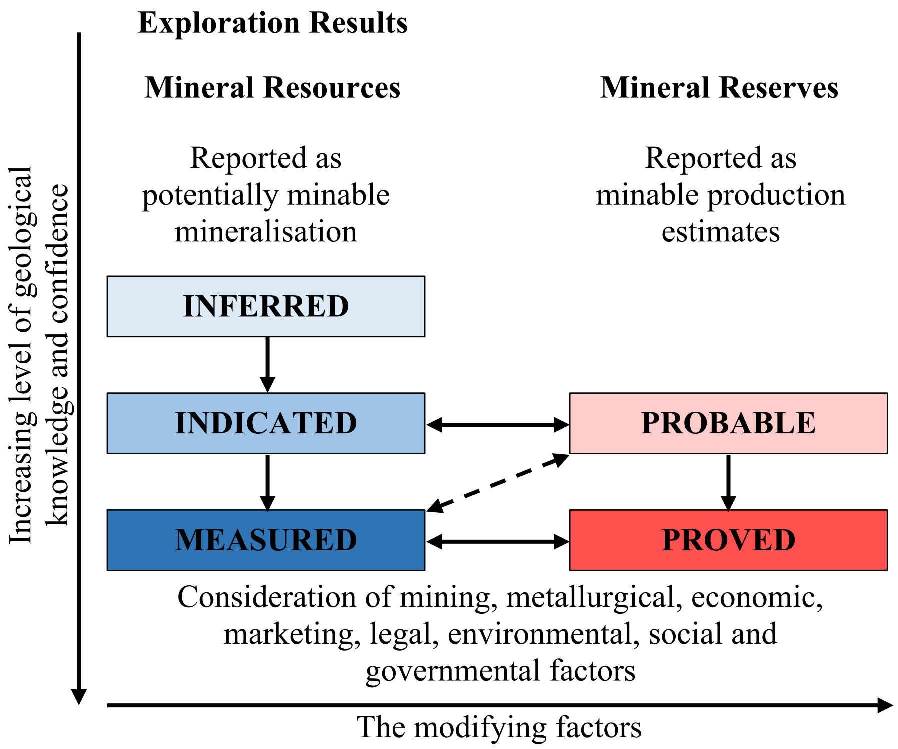


Figure 2.2 Relationship between Mineral Resources and Mineral Reserves in the exploration results (Source: Rendu, 2014)

2.4 Mineral Reserve

Mineral Reserves contain the economically minable portions of the Mineral Resource that yields the maximum profit or return on investment. According to the SAMREC (2016), Mineral Reserve is obtained from the conversion of measured or indicated resources using modifying factors as illustrated in Figure 2.2. A Mineral Reserve is also categorised into probable and proved Mineral Reserve. The evaluation of Mineral Reserve comes along with the determination of the pit limits.

Practically, Mineral Reserves represent the amount of material included in the ultimate pit limits subject to a continuous review along the mine planning process. The upgrading of the Mineral Reserves might either increase or decrease the total tonnage according to the current market conditions and planned dilution. The increase in the geological level of confidence can also initiate the translation of mineral portions from Mineral Resource to Mineral Reserve or vice versa. The key parameter for separating Mineral Resources from Mineral Reserves is the definition of the cut-off grade. In other words, the cut-off grade helps to define the boundaries that distinguish valuable portions from uneconomical portions. Material below the cut-off grade should remain within the Mineral Resource domain until a change occurs in the market price that enables economic extraction (Birch, 2019).

2.5 Modelling the deposit

During the mine planning process, the prior task to the ultimate pit limit and long-term production scheduling is the construction of models describing the mineral deposit. The model depicts the orebody in 3D space containing small regular and manageable blocks. For better visualisation and more convenience for some studies, the 3D model can be taken in cross sections resulting in 2D space representation of the block model. The model presents blocks along X (Easting), Y (Northing) and Z (Elevation) axes. The coordinates of a block refer to the centre point of the block. Figure 2.3 illustrates the representation of two block models in 2D and 3D configurations.

The definition of a block's dimensions are guided by the sample spacing, geological formations of the orebody, grade variability within the deposit, mining method and mining equipment size (Birch, 2019; Nogholi, 2015). Based on the drill pattern measurements, block dimensions must be within the range of one quarter to one-half of the average drill-hole interval (Hustrulid *et al.*, 2013). Small blocks increase error estimation while big blocks reduce detailed perception of the mineral deposit. Preferably, a block should include one geological characteristic enabling a better selectivity and reduce dilution. In an open-pit mine, the height of blocks is equivalent to the bench height previously determined according to the geotechnical parameters of the rock in the deposit and mining equipment size planned to be used for production.

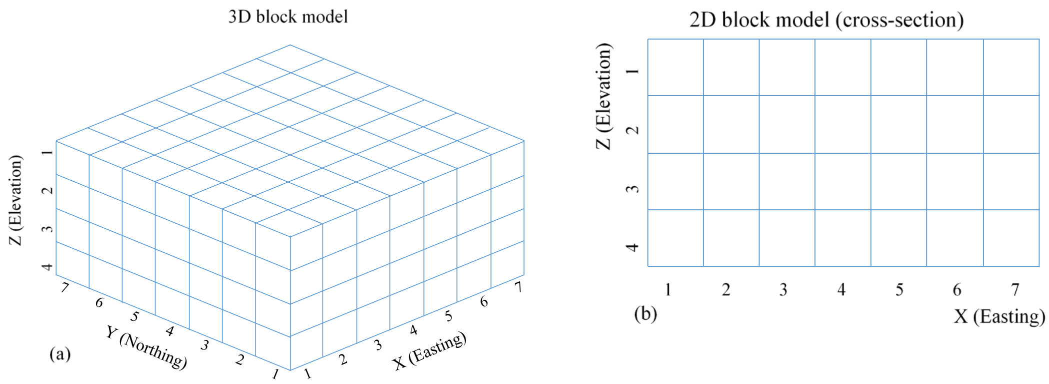


Figure 2.3 Graphical representation of the block model in 3D (a) and 2D (b)
(Source: Nogholi, 2015)

The mineral deposit can be modelled as a geological block model and economic block model. The geological block model consists of visualising the orebody features such as geological formations, rock types, and grade variability of the mineral of interest. These features are defined from the analysis of the drilling samples collected from the deposit. The size of the blocks in the geological block model is mostly driven by the exploration drill holes grid pattern (Birch, 2019). Ideally, the size of blocks in a geological block model and in an economic block model should be the same for easier geo-spatial referencing.

The economic block model consists of blocks assigned with their economic values obtained from the estimate values of the amount of the mineral of interest contained in the block. The block economic value is equivalent to the difference between the value estimate of the block in the market and the estimated cost associated with its extraction (Hustrulid *et al*., 2013). The value of the block is driven by the ore grade contained in the block and the current mineral price in the market. The total cost of a block involves the mining, processing and refinery costs. The economic block model is a critical data input in the formulation of OMPS in order to maximise the net present value of the mine.

2.6 Cut-off grade

During the mine planning process, the distinction between ore and waste materials is important for the optimisation of the open-pit mining limits and the establishment of the production schedule. This distinction is achieved by using a break-even cut-off grade. The grade is one of the estimate attributes for blocks during the block modelling of the deposit. It is the amount of the valuable element contained in a metric tonne of material, sometimes expressed in percentage (Rendu, 2014). Within the deposit block model, any block that has a grade, which is equivalent to or above the threshold grade, is deemed economically valuable ore and might be economic to be mined and processed at a profit. Blocks with a grade below the threshold grade are deemed as waste blocks as they are likely to be uneconomic to mine and process. Waste blocks may either be left *in-situ* or removed from the ground and sent to the waste dumps. Waste blocks that are sent to the waste dumps are included in the ultimate pit limits because they have to be removed to gain access to ore blocks, while blocks left behind fall outside of the final pit limit.

The determination of the optimum cut-off grade has an economic aspect and a direct effect on the monetary value realised. An optimum cut-off grade maximises the economic value of the deposit. The work of Lane (1964) on the economic cut-off grade is viewed as the pioneer reference on the definition and calculation of cut-off grades based on the overall maximisation of the NPV of future cash flows. Several studies have expended the knowledge on the topic to improve Lane's model on the optimisation of the cut-off grade. Hustrulid *et al*. (2013) gave the definition of the cut-off grade in the simplest term as a grade applied to attribute a destination label to a parcel of material.

The optimisation of the cut-off grade remains an important element in the construction of the OMPS long-term plan. During the conception of the block extraction plan, the cut-off grade should be used to identify ore blocks and waste blocks distinctly in a non-homogeneous block model. With regards to ore grade, it has been demonstrated that high cut-off grade in the early years shortens the payback period on the initial capital invested and maximises the NPV for an open-pit mine operation (Githiria and Musingwini, 2018). Additionally, Rendu (2014) established the benefit of operating at a cut-off grade greater than the break-even cut-off grade early in the life of the mine.

The break-even cut-off grade is the grade required for a block to return the total cost related to its extraction. Mathematically, the break-even cut-off grade is obtained by equalising the unit total cost and the revenue generated through the sale of a unit material. The unit total cost (UTC) depends on the total fixed cost (TFC), unit variable cost (UVC) and the quantity of material mined (X). The unit revenue (UR) is a product of the mine recovery factor (MRF), the selling price and the grade. Consequently, the MRF represents a resulting product of the mine call factor (MCF) and the plant recovery factor (PRF) (Birch, 2019). The optimisation of the cut-off grade requires a continuous review on a yearly basis over the life of the mine due to variations that occur in an unstable market. Equations 2.1 to 2.5 express the procedure of the computation of the break-even cut-off grade (Birch, 2019):

$$UTC[US\$/tonne] = \left(\frac{TFC[US\$/tonne]}{X[tonnes]}\right) + UVC[US\$/tonne] \tag{2.1}$$

At break-even value, the unit revenue is equal to the unit total cost of mining:

$$UR = UTC \tag{2.2}$$

Where,

$$UR\left[\frac{US\$}{tonne}\right] = Break - even\ grade[g/tonne] \times MRF[\%] \times Price[US\$/gram] \tag{2.3}$$

Therefore,

$$Break - even\ grade \times MRF \times Price = \left(\frac{TFC}{X}\right) + UVC \tag{2.4}$$

From which,

$$Break - even\ grade = \frac{\left(\frac{TFC}{X}\right) + UVC}{Price \times MRF} \tag{2.5}$$

2.7 Ultimate pit limit

The fundamental problem in open-pit mine design optimisation consists of determining the ultimate pit limits, which defines the economic material to remove from the ground. The ultimate pit can be divided into manageable small pits often referred to as phases, pushbacks, or cutbacks (Meagher *et al*., 2014). The economic block model is one of the requirements for the determination of the optimum pit limits defining the contour that provides the maximum net present value, whilst complying with the operational constraints. Blocks within the ultimate pit represent the set of blocks that can be mined economically. These blocks include ore and waste blocks.

Waste blocks included in the ultimate pit limits represent blocks that must be removed to give access to the ore blocks. Practically, a block is ready to be extracted only when all its overlying blocks have been mined. The number of predecessor blocks is determined based on the slope angle of the mine respecting the geometry of the pit as dictated by geotechnical requirements for a stable and safe pit slope angle. The extraction of ore blocks contained in the final pit limits must ensure the payback of costs associated not only with their mining, processing and marketing, but also with the stripping of the overlapping waste blocks (Akbari *et al*., 2008).

Mine planning and pit optimisation are dynamic activities that are continuously upgraded to integrate changes that may occur in the geology of the orebody. For example, the size of the pit might be reviewed with more exploration drilling because of the under/over-estimation of the resource from limited samples. Traditionally, the ultimate pit limit is reached when the revenue generated through the extraction of a unit of material becomes less than the cost of its extraction. Despite the availability of many techniques used for determining the ultimate pit limits, there are two principal classical methods namely: Floating Cone and Lerchs and Grossmann algorithms (Newman *et al*., 2010). These methods form part of an integrated process in many commercial software packages such as Whittle®, Vulcan® and MineSight® (Cuba and Deutsch, 2011). The following sub-sections briefly review these two techniques.

2.7.1 Ultimate pit limits using the Lerchs and Grossmann algorithm

Lerchs and Grossmann (1965) proposed using dynamic programming (DP) for solving the ultimate pit limits on a two-dimensional section of a block model. The Lerchs and Grossmann (LG) algorithm yields the optimum pit limits under slope angle constraint represented by precedence relations between blocks (Marcotte and

Caron, 2012). The aim of the pit optimisation activity is the design of the boundary of a pit, which maximises the total profit of the mine. The profit implies the difference between the total value of mineable ore blocks and the total cost of mining ore and waste blocks (Lerchs and Grossmann, 1965). The essence of the LG algorithm is to split the economic block model into parallel vertical sections or 2D space, then determine the contour of the pit on each section that yields the maximum profit. In order to demonstrate the precedence block constraint, the method converts the 2D economic block model into a directed graph consisting of a set of nodes (blocks) connected to each other by a connecting arc (Khalokakaie *et al*., 2000). The original LG method assumes a slope angle of 45° to define the number of overlying blocks (Lerchs and Grossmann, 1965). The general configuration of the pit contour on a cross-section (2D) consists of three sides: two walls inclined at a certain slope angle and the bottom level of the pit.

Analytically, the LG seeks to maximise the objective function Z shown in Equation 2.6 to design an optimum pit:

$$Max\ Z = \sum_{i,j=1}^{n} BEV_{ij} \tag{2.6}$$

$$BEV_{ij} = v_{ij} - c_{ij} \tag{2.7}$$

Where:

Z is the objective function of the ultimate boundaries on a cross-section;

BEV_{ij} is the block economic value (BEV) of block x_{ij};

v_{ij} is the gross sale revenue of block x_{ij}; and

c_{ij} is the total extraction cost of block x_{ij}.

The first step is to compute the cumulative profits M_{ij} realised after the extraction of a single column of blocks having at its base the block x_{ij}. The calculation of the M_{ij} within a column is independent of other columns as shown in Equation 2.8 where the index l identifies all blocks involved.

$$M_{ij} = \sum_{l=1}^{i} BEV_{lj} \tag{2.8}$$

The next step is the introduction of a dummy row of zeros on top of the block section, as demonstrated in Figure 2.4, followed with the calculation of the P_{ij} (overall cumulative values) across the section moving from left to right, a column at a time. The P_{ij} values result also from the contribution of one of three blocks in column $(j-1)$ located directly above $(k=-1)$, adjacent $(k=0)$, and below $(k=1)$ to the left-hand side of the block. Over the three blocks, an arrow is drawn (see Figure 2.5) to indicate the direction from the block x_{ij} to the one that gives the maximum P_{ij} value against the other two blocks. The P_{ij} solution replaces the M_{ij} value and will be used for the subsequent calculations based on Equation 2.9.

$$P_{ij} = M_{ij} + max\{P_{i+k,j-1}\} \tag{2.9}$$

i\j	*0*	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	
0	0	0	0	0	0	0	0	0	0	0	→ Dummy row
1		-1	-1	-1	-1	-1	-1	-1	-1		
2		-2	1	3	-1	0	-1	1	-2		
3		-4	-1	1	-3	-2	-3	-1	-4		
4		-6	-3	-1	5	-4	-5	-3	-6		

Figure 2.4 Introduction of a dummy row on a cross-section block model of M_{ij} values

The process continues until the last column generating overall cumulative values P_{ij} on each block. The optimum pit contour on the section is that which yields the maximum cumulative value, thus the maximum net value. Once the maximum value is found on row 1, the pit outline proceeds through the direction indicated by the arrows beginning from the block with the maximum value to the left-hand side of the section. Any feasible pit obtained on the section must contain at least one block from the first row. If none of the P_{ij} values on the first row are positive, there is no boundary that can be mined at a profit (Hustrulid *et al.*, 2013). The sketch of the outline determines the optimum pit limit on the section that complies with the precedence block constraint through pit slope angle.

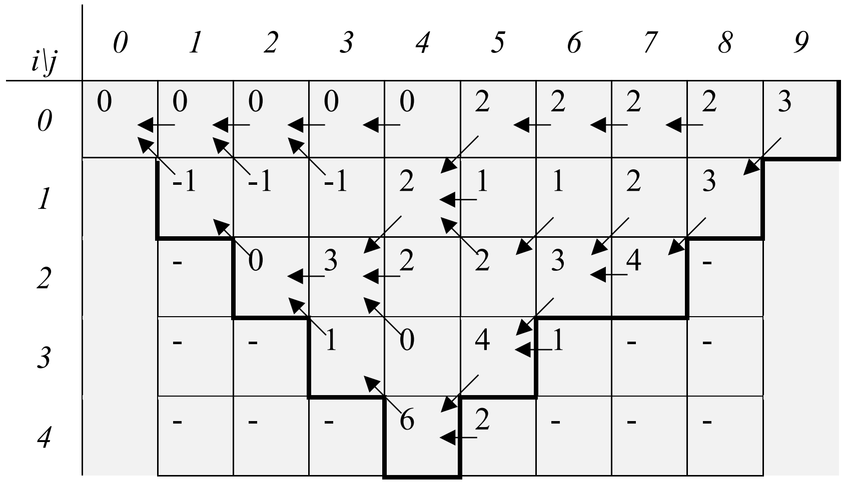


Figure 2.5 Overall cumulative values of P_{ij} ***on a block model cross-section***

Although the LG approach optimises the pit limit in 2D by maximising the net value on the section, the approach suffers some drawbacks such as (Hustrulid *et al.*, 2013):

- Failure to ensure that the optimum limits on the sections fit with one another; and
- Difficulty to implement the optimum limits in a 3D space.

However, several pieces of research have been conducted to improve the LG approach in a 3D space. Johnson and Sharp (1971) extended a 2D to a 21/2D dynamic programming algorithm by applying the LG on both cross-section and longitudinal section, but still requires some smoothing adjustment to fit sections one another. Most of the ultimate pit limit algorithms tried to maximise the undiscounted cash flow regardless of the production schedule until Erarslan and Celebi (2001) proposed a dynamic programming algorithm that simultaneously optimises the ultimate pit limits and production schedule.

2.7.2 Ultimate pit limits using the Floating Cone algorithm

Pana (1965), cited in Hustrulid *et al*. (2013), developed the Floating Cone approach to find the optimum pit limits on a 2D space block model by searching the cone in which the cumulative value of all the blocks is positive. This approach is one of the alternatives to LG dynamic programming due to its flexibility and time efficiency. The approach is based on a heuristic idea whereby multiple cones float over a 2D block model to find the pit contour of the maximum value on the section. The slope angle restricts the process allowing the removal of overlying blocks prior to the extraction of ore blocks. Each ore block is assigned a cone in which the cumulative value must be positive to make its extraction valuable. The cone is laid on an upside-down position on the blocks, having at its apex the ore block with a positive BEV as shown in Figure 2.6.

The Floating Cone method proceeds in a top-down direction row after row. The cone floats from left to right and positions its apex on ore blocks to include overlying blocks that have to be removed in order to extract the ore block. Obviously, ore blocks on the first row do not have overlying blocks; therefore, their extraction does not require the removal of block predecessors. The extraction of each block depends on the cumulative value of the cone, if this is positive or zero the extraction can proceed if not the extraction of the ore block is uneconomic. The process continues until no more ore blocks can be economically mined. The total cumulative value on the cross-section results from summing up the BEVs of blocks within the final cone, which is the combination of all the removal cones. Figure 2.6 illustrates the cone generated from block x_{23} of which the cone cumulative value is one. However, the outline of the final cone on the section determines the ultimate pit limits through the Floating Cone method in 2D space.

i\j	1	2	3	4	5	6	7	8
1	-1	-1	-1	-1	-1	-1	-1	-1
2	-1	2	4	0	1	0	2	-1
3	-2	-2	-2	-2	-2	-2	-2	-2
4	-2	-2	-2	8	-2	-2	-2	-2

$$C_{23} = -1 - 1 - 1 + 4 = 1$$

Figure 2.6 Illustration of the cumulative value of a cone positioned on Block x_{23}

The implementation of the method faces some drawbacks that prevent the Floating Cone to find an optimum pit on a section. This often happens when using the simple cone analysis whereby the BEV of a single ore block is not enough to satisfy the removal of the overburden waste blocks in the cone. These cones should normally not take part in the optimum pit. However, the Floating Cone often includes non-profitable blocks in the pit reducing the net value of the pit. Although the method is simple and easy to understand, it fails to provide true optimum pit limits in some specific instances (Elahi zeyni *et al.*, 2011). Newman *et al.* (2010) highlighted the following drawbacks of the Floating Cone approach:

- Determination of the ultimate pit depends on the sequence in which the reference blocks are designated; and
- Existence of some reference blocks that may need to be designated in advance to produce a pit practically feasible, yet not always optimal.

To overcome these shortcomings, Wright (1999) cited in Hustrulid (2013), proposed an improved moving cone II that computes the economic values of the cones of ore blocks on each row, from which only cones with positive values will be included in the final optimum pit. The process continues finding positive cone economic values row after row. However, the Floating Cone II might fail to produce an optimum pit, which can be found by combining two or more uneconomical cones on a cross-section of a block model. Figure 2.7 illustrates two uneconomic separated cones that will be rejected but putting them all together will generate a valuable pit with a positive cumulative value as shown in Figure 2.8.

i\j	*1*	*2*	*3*	*4*	*5*	*6*	*7*
1	-1	-1	-1	-1	-1	-1	-1
2	-2	-1	-1	-1	-1	-1	-2
3	-3	-3	6	-2	7	-3	-3

$C_{33} = -2$ (a)

i\j	*1*	*2*	*3*	*4*	*5*	*6*	*7*
1	-1	-1	-1	-1	-1	-1	-1
2	-2	-1	-1	-1	-1	-1	-2
3	-3	-3	6	-2	7	-3	-3

$C_{33} = -1$ (b)

Figure 2.7 Two separated uneconomic cones rejected in Floating Cone II

Elahi zeyni *et al.* (2011) developed a Floating Cone III that modifies and improves the previous versions of Floating Cone method in a way to tackle its inability of generating an optimum pit in some specific cases. To do so, in his approach, cones that have common overlying blocks are dependent on one another and cones that have no common blocks are classified as independent. Therefore, dependent cones may generate a positive value through mutual support between cones. An example of this is given in Figure 2.8. Effective cones are cones with positive cumulative value and ineffective cones are those with negative cumulative value. The final optimum pit solution forms a set of effective cones.

i\j	*1*	*2*	*3*	*4*	*5*	*6*	*7*
1	-1	-1	-1	-1	-1	-1	-1
2	-2	-1	-1	-1	-1	-1	-2
3	-3	-3	6	-2	7	-3	-3

$Pit\ value = 1$

Figure 2.8 Ultimate pit limits generated from two uneconomic cones in Floating Cone III

2.8 Generic open-pit mining

2.8.1 Open-pit geometry

The main geometric elements of an open-pit mine that have to accommodate mining equipment and mining operations are benches, haul roads and the disposal areas for stockpile and waste dumps (Wetherelt and Van der Wielen, 2011). Benches are designed to extract material from the pit and serve as working space; they also accommodate space for blasting and excavation operations. In line with mining activities in the pit, there exist two types of benches: working benches and inactive benches. Working benches are those being mined, and the inactive benches maintain pit slope stability (Hustrulid *et al*., 2013). Figure 2.9 illustrates a typical open-pit mine showing the important geometric elements. For additional safety reasons, the faces of the benches are inclined by an angle defined according to the geo-mechanical quality of the rock in place (Hustrulid *et al*., 2013). During excavation operations, the bench angle is adhered to because of its critical contribution on the overall slope angle of the pit. In addition, the pit slope angle delineates the pit on the sides to define the portion of the deposit, which can be economically mined. The overall pit slope angle is one of the most important input parameters for the optimisation of the ultimate pit limits as described in the previous section.

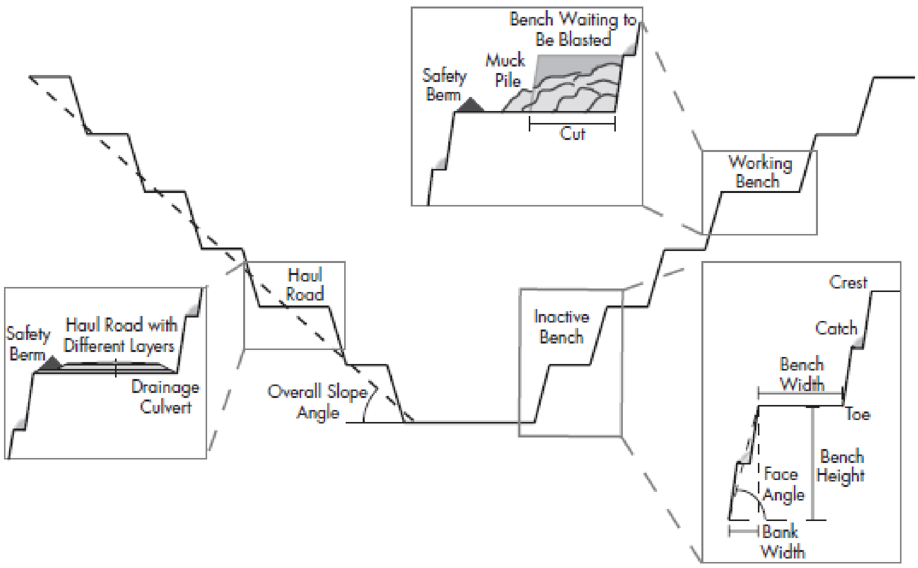
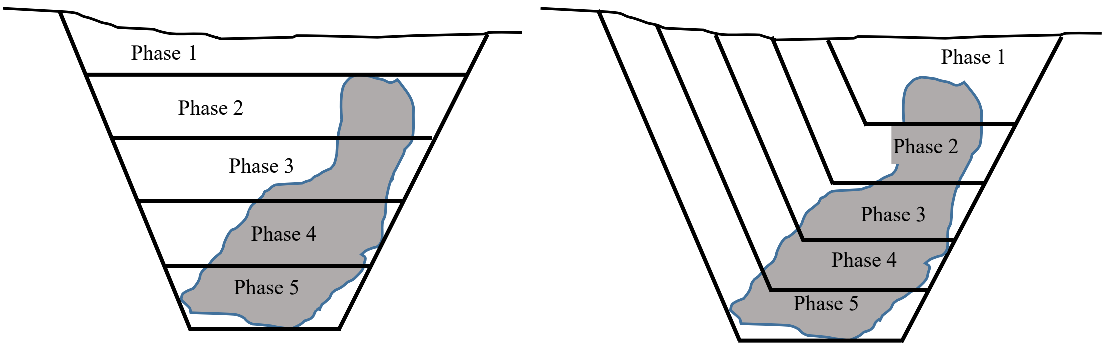


Figure 2.9 Typical open-pit mine geometry (Source: Wetherelt and Van der Wielen, 2011)

Ramps or haulage roads are created in a truck and shovel system for the haulage purpose to transport the material from the mine and give access to different benches. Roads in an open-pit mine wind-up from the bottom of the pit to the vicinity of the pit entrances. The steepness and the wideness of the roads are defined according to the type of trucks used, traffic intensity, lifespan of the ramp and costs related to the construction (Wetherelt and Van der Wielen, 2011). The quality of haulage roads is of great importance in the realisation of the target production capacity, thus, must be considered with much attention. The road design integrates the geometry of the road, the structural material for road construction and the functional design. Good management of haulage roads must ensure continuous road maintenance to keep smooth haulage of the mined product along the production chain (Thompson and Visser, 2006). It is important to mention that haul roads are the lifeline of the production system; consequently, bad road design and poor maintenance increase the operational costs and reduce the productivity of the mine.

2.8.2 Open-pit expansion

Open-pit mining operations start with a small pit at the top bench expanding to a larger pit in a series of phases called pushbacks (Nogholi, 2015). The pushback sequences are pre-established during the mine planning in order to guide the production plan on a yearly basis (Osanloo *et al.*, 2008). In the pursuit of optimal pushbacks, some factors should be considered such as the grade of *in-situ* material, the costs of development, mining and processing capacities and the market conditions (Wetherelt and Van der Wielen, 2011). However, pushback open-pit expansion is aiming at generating high cash flow in the early years of the mining, while keeping the stripping ratio at its lowest level. The development of pushbacks can be done either using the conventional method or the sequential method (Wetherelt and Van der Wielen, 2011). Both methods divide the main pit into small nested pits that represent different pushback zones of exploitation as illustrated in Figure 2.10. The conventional method (see Figure 2.10 (a)) proceeds in a bench-by-bench extraction allowing one active bench at a level until the pit contours are reached, this results having all activities concentrated on one level. From the planning point of view, the conventional method is less complex to schedule and offers a wider active bench that may accommodate more working places on one level. However, the sequential method (see Figure 2.10 (b)) extends the pit through multiple active benches pushing on different levels creating a more complex schedule planning. Although a complex schedule, the sequential method offers the advantage of being flexible during excavation and enables the mining of different quality of ore essential for ore blending. Additionally, the sequential method generates high NPV compared to the conventional method because of its lower stripping ratio and the ability to select high ore grade in the early life of mine.



(a) (b)

Figure 2.10 Open-pit mine expansion methods: (a) conventional method, (b) sequential method (Source: Whittle, 2011)

2.9 Chapter summary

This chapter defined the context of mine planning in which production scheduling takes place. Between the orebody modelling and the mining extraction operation, several optimisation problems require optimal solutions to maximise the value of the mine. The most popular techniques used for the optimisation of the ultimate pit limits are the Lerchs and Grossman algorithm and the Floating Cone currently added as built-in components in some mining software tools such as Whittle®, MineSight® and Vulcan®. The orebody block modelling (geological and economic block models) and grade estimates enable the implementation of these optimisation techniques on the orebody. A geological block model illustrates the geological features of the orebody including rock formation, rock type, grade estimate and geographical location. The economic block model delivers the monetary value of each block in the model. The optimisation of cut-off grade remains an important element not only for the ultimate pit determination, but also for the construction of the OMPS long-term plan. Cut-off grade distinguishes ore blocks from waste blocks resulting in determining the destination to which blocks must be sent either the crushing plant, stockpile or waste dump.

With respect to the open-pit mine expansion, there exists two methods namely: conventional method and sequential method. The latter provides many advantages compared to the former as it allows a flexible excavation with many working benches. Also, the sequential method is suitable for selecting higher grade portions within the early years of operation essential to maximising the NPV and to shorten the payback period on the initial capital invested. Therefore, a good production plan would adopt the sequential method to meet the overall goal of maximising the profit out of the mine. The next chapter discusses the open-pit mine production scheduling.

3 LITERATURE REVIEW ON OPEN-PIT MINE PRODUCTION SCHEDULING

3.1 Chapter overview

This chapter discusses the focus of the research that is open-pit mine production scheduling. Mine production scheduling is known to be a difficult and complex problem along the mine planning process and requires optimisation techniques for its resolution. To begin with, the chapter introduces the scope of the problem and the activities that are involved in the construction of the production schedule. Considering the duration of the period covered and planning details, the production plan is usually divided into three planning horizons namely: long-term, medium-term and short-term planning horizons. With regard to the aim of this research to finding the optimal long-term production plan, this chapter highlights the importance of the production schedule towards the overall goal of mine planning. Furthermore, it develops the formulation of the open-pit mine production scheduling optimisation subject to geological, operational and physical constraints. It also reviews the techniques and methods that have been used in the literature to tackle the problem. Finally, the chapter presents a brief summary of the commercial software available in the mining software market referring to the production scheduling optimisation.

3.2 Introduction

Open-pit mine production scheduling consists of establishing the plan of mineral extraction throughout the life of the mine. It is a very complex task at the same time very important for the mining company to meet the production target satisfying the economic expectation of the project. The OMPS plan is established over the ultimate pit, which contains a mixture of ore and waste blocks that yields the highest value of the deposit under pit constraints. The production plan generated through the OMPS plan defines the sequence in which blocks within the ultimate pit should be extracted in order to maximise the undiscounted cash flow of the mine.

An ideal ultimate pit limit should contain only ore blocks, but because of the non-homogeneous grade distribution, the pit contains a mixture of ore and waste blocks.

Waste blocks within the ultimate pit are those blocks that require their removal to access ore blocks. The removal of waste blocks constitutes an extra cost on the cost of extracting ore material. The cost associated with the removal of waste blocks must be minimised and redeemed through the selling of the final product. Objectively, the revenue generated from each ore block must be enough to pay back not only the cost of its extraction, but also for the extraction of overlying waste blocks. The production schedule should therefore define the best time at which each block whether ore or waste has to be mined based on physical, technical and economic constraints in order to maximise the NPV. Additionally, OMPS plan should define the destination (processing plant, stockpile or waste dump) to which blocks should be sent.

Thorley (2012) proposed a diagram (see Figure 3.1) that illustrates the activities involved in the establishment of the OMPS plan. His diagram starts off by setting the production capacities and evaluating the production costs relating to the mining cost and processing cost. However, production capacities (mining and processing capacities) would constrain the long-term plan later in the process to meet the expected production rate of the mine. In the same way, production cost constrains the short-term plan with the objective of minimising the mining cost. Then comes the pit optimisation process to define the ultimate pit with the maximum profit, in which a number of pushbacks or nested pits must be designed. Before defining the production plan, an optimum cut-off grade must be defined to distinguish ore from waste. Thorley's diagram considers any change that would occur during the mine planning process to readjust previous tasks as they interact.

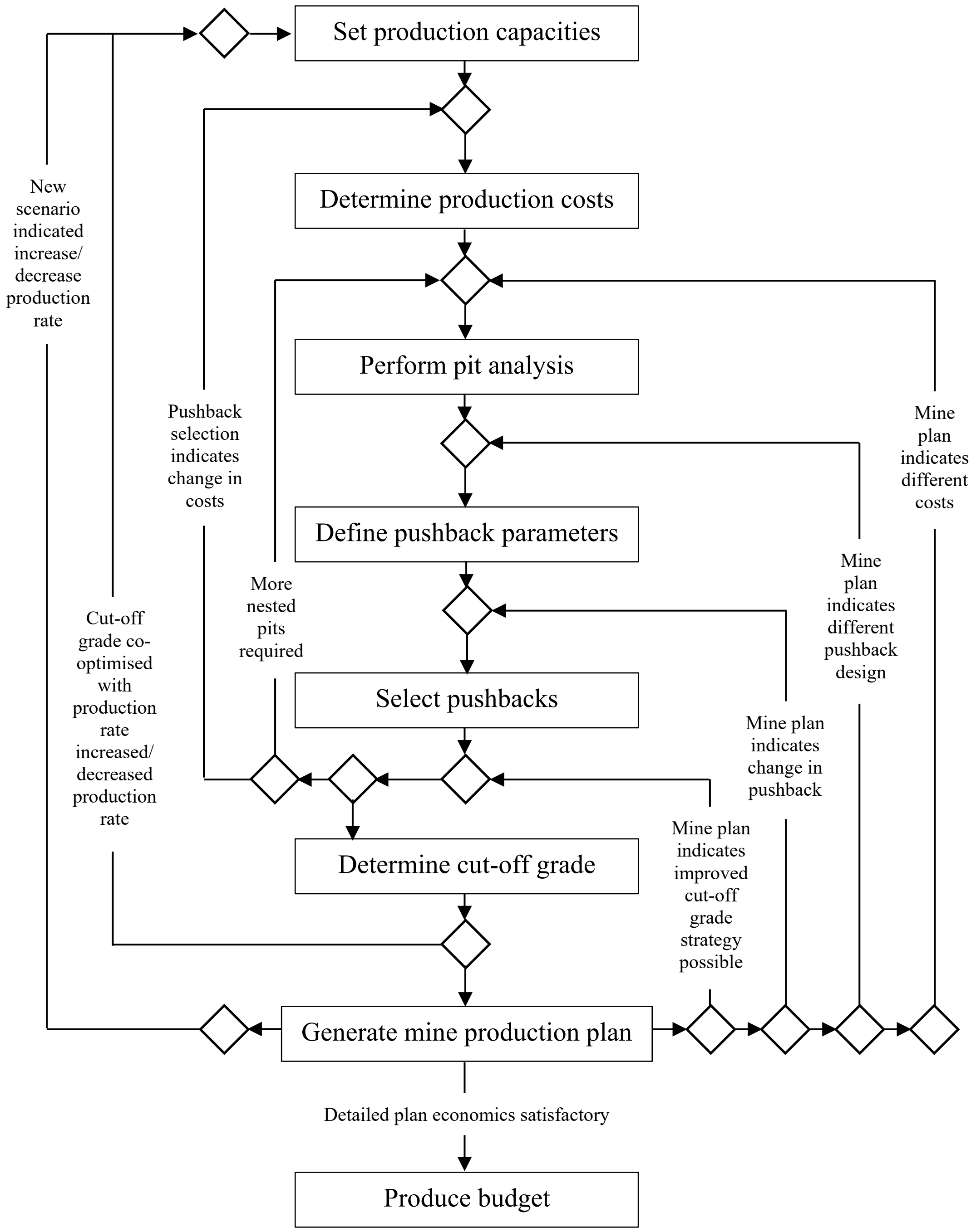


Figure 3.1 Activities involved in the establishment of the open-pit mine production plan (Adapted from: Thorley, 2012)

Following the production plan diagram proposed by Thorley, the OMPS plan must be divided into three-time horizons, which are long-term plan, medium-term plan and short-term plan (Osanloo *et al*., 2008). These time planning horizons differ in several key dimensions including the time-span of planning forecast, the type of block model used, the objectives pursuit, the optimisation constraints used, and the level of detail to which the production schedule is modelled (Blom *et al*., 2018). Dimitrakopoulos and Jewbali (2013) added that the OMPS long-term plan focuses on maximising the NPV of the project while the short-term plan intends to meet the daily, weekly and monthly production targets and minimise the mining costs.

3.2.1 Long-term OMPS plan

During the mine planning process, the elaboration of the OMPS long-term plan that covers the life of the mine is critical in the realisation of the maximum economic value of the mine. The OMPS long-term plan anticipates the production figures for many years in the future and plays a major role in reporting the outcome of the feasibility studies of a mining project. The feasibility report should establish whether to proceed with the investment or to discard the project (Chicoisne *et al*., 2012). If the project is deemed viable, a production rate per time period should be determined until the completion of the extraction of the Mineral Reserve. In OMPS long-term plan, blocks are scheduled up to 5 years or more in the future (Osanloo *et al*., 2008) creating pushbacks in the pit. The OMPS long-term plan is subject to a continuous review to implement changes that may occur in the geology of the mineral deposit or in the market (Shonts and Nettleton, 2011).

The OMPS long-term plan considers some constraints that include ore grade control, block precedence constraint, mining capacity, processing capacity and Mineral Reserve constraint (Osanloo *et al*., 2008). The importance of the long-term production schedule is not only to determine the NPV of the project and the life of the mine, but it is also to serve as the foundation for the medium-term and short-term plans (Blom *et al*., 2018).

3.2.2 Medium-term and short-term OMPS plans

Considering the complexity of delivering the production plan for a long-term horizon involving a large number of blocks to schedule across many years, it is essential to split the long-term plan into shorter time-ranges. That is why the OMPS medium-term plan breaks down the OMPS long-term plan into plans of a minimum of one year-interval (Osanloo *et al*., 2008). The portions of blocks defined in the long-term plan are considered individually in the medium-term whereby material is scheduled on a yearly basis.

The medium-term plan identifies the annual ore advance (tonnes and location) to meet the plant demand in terms of the tonnage and grade supplied. Since waste blocks are present in the pit, the plan must also identify the waste advance in order to maintain the stripping ratio of the mine. Preferably, the stripping ratio should adopt an increasing trend as the pit deepens and widens. As being operational than the long-term plan, the medium-term plan would seek to minimise the loader relocation, meet stockpile and dump construction needs. The plan should forecast the availability of utilisation of the mining equipment as well as the future replacement of equipment.

Annual determination of production quantity is essential for the calculation of the cash flows during the financial studies. The determination of cash flows validates the financial viability of the project with a year-to-year production rate. The time horizon in the medium-term plan is not yet to an absolute plan to schedule a daily, weekly and monthly plan for the extraction of selected blocks in the pit. Therefore, it is essential to introduce an OMPS short-term plan with the goal of breaking down the one-year production schedule into months.

In a short-term production plan, the time at which each block must be mined is defined in a period of less than a month. The OMPS short-term plan takes into consideration the following constraints: mining extraction sequence, production capacity, grade control, processing demand, stockpiling and blending (Nogholi, 2015). Typically, the objectives being optimised in the short-term plan consist of maintaining the grade within the optimum cut-off grade to meet production targets, maximising equipment utilisation, and minimising re-handling and costs of production (Blom *et al*., 2018). The short-term plan outlines extraction stages in terms of months, weeks or days.

A monthly schedule plan is a breakdown of the annual production capacity into regular planning horizons spread on a monthly basis (Osanloo *et al*., 2008). The plan evaluates major stoppages of the operations during times of equipment unavailability or maintenance, impracticability due to weather conditions and other unpredictable factors. Monthly sequence of extraction is guided by the annual production schedule required to be executed and completed in order to meet the annual production target. The objective function should be based on minimising the deviation from the annual production plan, which is deemed optimal during the financial valuation studies (Matamoros, 2014). During monthly scheduling, it is sometimes necessary to recalculate equipment hours and the numbers of truckloads in order to assess ore and waste schedule changes. The details provided in monthly production plans also include weekly and daily execution schedules, production rates and planned tasks.

A weekly plan selects production polygons to be mined in each week of the horizon, and the destination to which this material should be sent, to meet weekly quality and quantity targets on production. The plan establishes the scheduling of blasting, which determines when certain production polygons would become available for mining (Nogholi, 2015). The weekly objectives vary and may include maximising equipment utilisation, minimising material re-handling, maintaining the grade of production and meet the daily production rate (Blom *et al*., 2018). The daily production plan captures a range of day-to-day and shift-to-shift activities such as drilling, blasting, excavation and haulage. Therefore, it identifies drill patterns and blast patterns to maintain inventory, and determines the crusher set to receive the mined material (Matamoros, 2014). It also determines waste shovel priority and the number of trucks to maintain the stripping ratio of the mine. The main focus of the weekly and daily extraction sequence is to design a series of production polygons to be extracted on the weekly and daily planning horizons (Blom *et al*., 2018).

3.3 Problem formulation

The OMPS problem is a large-scale combinatorial optimisation problem that involves several thousands to millions of blocks and a set of multiple constraints. The size of the dataset, long time range and multiple constraints involved in the optimisation process make the OMPS problem a complex and difficult task to solve. The set of constraints considered in the OMPS long-term optimisation problem include grade control, mining capacity, processing capacity, reserve constraint and slope constraint. These constraints are considered in order to deliver an optimum production schedule that maximises the discounted profit or the NPV for the mine.

The optimisation of the OMPS problem succeeding the pit optimisation problem in mine planning activities must schedule blocks within the pit for mining extraction over the life of the mine. The optimum solution to this problem must define the time at which each block should be mined in accordance with all the constraints and the destination to which it should be sent after its removal from the pit. Solving the OMPS problem aims at maximising the discounted value of the mine expressed as the objective function of the model. Mathematically, the OMPS problem can be formulated through a mixed inter-programming structure as demonstrated in the subsections below. The mathematical model of the OMPS problem involves notations, which are described in Table 3.1.

Table 3.1 Description of notations used for OMPS optimisation problem

Notation	Description
$\boldsymbol{i}$	Block index, $i = 1,2, \dots, N$.
$\boldsymbol{t}$	Time period index, $t = 1,2, \dots, T$.
$\boldsymbol{r}$	Time period index anterior or equivalent to time period t.
$\boldsymbol{T}$	Total duration over which blocks are scheduled.
$\boldsymbol{N}$	Total number of blocks within the block model.
$\boldsymbol{x_i}$	Binary variable that takes 0 for blocks outside the ultimate pit limit and 1 for blocks inside the pit.
$\boldsymbol{P_i}$	Set of block predecessors of block i.
$\boldsymbol{l}$	Block index of block predecessors.
$\boldsymbol{V_i}$	Block economic value of block i.
$\boldsymbol{d}$	Discount rate.
$\boldsymbol{m}$	Material type, where waste takes 1 and ore takes $2, \dots, M$ according to the grade value.
$\boldsymbol{Z}$	Objective function
$\boldsymbol{v_i^{tm}}$	Discounted economic value obtained by mining a unit weight of material type *m* from block i during period t.
$\boldsymbol{x_i^{tm}}$	Proportion of material of type *m* from block i to be mined during period t.
$\boldsymbol{x_l^{rm}}$	Proportion of material of type *m* from block *l* mined during period *r*.
$\boldsymbol{G_{min}^{tm}}$	Minimum average grade of ore material of type m during period t.
$\boldsymbol{G_{max}^{tm}}$	Maximum average grade of ore material of type m during period t.
$\boldsymbol{g_i}$	Average grade of block i.
$\boldsymbol{Q_i}$	Total quantity of material in block i, in tonnes.
$\boldsymbol{M_{min}^{tm}}$	Minimum mining capacity of all material of type m during period t.
$\boldsymbol{M_{max}^{tm}}$	Maximum mining capacity of all material of type m during period t.
$\boldsymbol{P_{min}^{tm}}$	Minimum processing capacity of ore material type m sent to the plant during period t.
$\boldsymbol{P_{max}^{tm}}$	Maximum processing capacity of ore material type m sent to the plant during period t.

3.3.1 Objective function

The fundamental objective function for the OMPS problem seeks to maximise the total undiscounted cash flow generated from mining the mineral material throughout the life of the mine. In other words, the total mining value represents the sum economic values of ore and waste blocks inside the ultimate pit limit. Mathematically, the total economic value can be expressed as in Equation 3.1 applied on an economic block model (Khan, 2018).

$$Maximise \sum_{i=0}^{N} V_i \, x_i \tag{3.1}$$

Where:

N is the total number of blocks within the block model;

V_i is the block economic value of block i; and

$$x_i = \begin{cases} 0, & for\ blocks\ outside\ the\ pit\ limit \\ 1, & for\ blocks\ inside\ the\ pit\ limit \end{cases}$$

Equation 3.1 does not provide the time horizon and the extraction sequence of blocks over the life of the mine. However, it discards blocks from the block model that should not be relevant in the planning process of an open-pit mine. It is known that all the mathematical formulations of exact models for the optimisation of the OMPS long-term problem are developed based on Equation 3.1. Osanloo *et al.* (2008) proposed a general objective function for solving the OMPS problem based on the work of Johnson (1968) that can be formulated as shown in Equation 3.2. The calculation of the discounted economic value (v_i^{tm}) is shown in Equation 3.3.

$$Maximise\ Z = \sum_{t=1}^{T} \sum_{m=1}^{M} \sum_{i=1}^{N} v_i^{tm} * x_i^{tm} \tag{3.2}$$

$$v_i^{tm} = \frac{V_i^m}{(1+d)^t} \tag{3.3}$$

Where:

v_i^{tm} is the discounted economic value obtained by mining a unit weight of material type m from block i during period t;

x_i^{tm} is the proportion of material of type m from block i to be mined during period t; and

d is the discount rate.

The objective function in Equation 3.2 is to maximise the NPV of the cash flows of the mine by providing the time horizon through which blocks in the pit should be mined. Considering the constraints involved in the OMPS problem making the problem more complex and difficult to solve, it is vital to optimise the extraction sequence with respect to all the constraints applied to the OMPS long-term problem. The following sub-sections describe the long-term OMPS constraints as described by Osanloo *et al.* (2008) and Khan (2018).

3.3.2 Grade control

The ore grade distribution is subject to variation within the ultimate pit limits. The quality of ore mined must fall between the grade range required at the processing plant. During the planning process, grade control is critical for obtaining a realistic production scheduling. The grade is a key factor in determining the mining reserve available and the profitability of the mine. Operating at a low grade increases the Mineral Reserve; hence extend the life of mine. A higher grade reduces the Mineral Reserve and maximises the short-term profitability of the mining cash flow (Mugwagwa, 2017). An optimal open-pit mine production plan should allow high grade in the early years to reduce the payback period on the initial capital invested. The consideration of the grade control during OMPS solution enables a smooth process at the subsequent stages in the mine value chain. Mathematically, the mining grade control can be expressed as in Equation 3.4.

$$G_{min}^{tm} \leq \left(\sum_{i=1}^{N} g_i \,.\, Q_i \,.\, x_i^{tm} \middle/ \sum_{i=1}^{N} Q_i \,.\, x_i^{tm} \right) \leq G_{max}^{tm} \qquad (3.4)$$
$$\forall\ t = 1,2, \dots, T; m = 2,3, \dots, M$$

Where:

G_{min}^{tm} represents the minimum average grade of ore material of type m during period t;

G_{max}^{tm} represents the maximum average grade of ore material of type m during period t;

Q_i is the total quantity of material in block i, in tonnes; and

g_i is the average grade of block i.

3.3.3 Mining capacity

The mining capacity represents the total tonnage of material mined per period for the depletion of the Mineral Reserve across the life of the mine. The volume of material required is defined according to both the set of equipment used and the size of the deposit (Godoy and Dimitrakopoulos, 2004). The total mining capacity includes ore and waste blocks. There are minimum and maximum production capacities that must not be exceeded when planning an OMPS long-term plan. During the mine planning process, the production rate is expressed in tonnes per year. The mining capacities (see Equation 3.5) constrain the OMPS problem in such a way that the tonnage of ore blocks scheduled for extraction per period meets the required production rate of the mine.

$$M_{min}^{tm} \leq \sum_{i=1}^{N} Q_i \,.\, x_i^{tm} \leq M_{max}^{tm} \;\; \forall\, t = 1,2, \dots, T; m = 1,2, \dots, M \qquad (3.5)$$

Where:

M_{min}^{tm} represents the minimum mining capacity of all material of type m during period t; and

M_{max}^{tm} represents the maximum mining capacity of all material of type m during period t.

3.3.4 Processing capacity

The processing plant operates on a range of tonnage capacity predefined according to the size of the plant. In this way, the total tonnage of ore material delivered to the plant must not exceed the maximum capacity of the plant or must not be less than the minimum tonnage required. Moreover, the quality of ore material must also meet the requirement because it is not only the quantity that matters, but also the quality to enable a good functioning of the plant. The processing capacities (see Equation 3.6) constrain the OMPS objective function by setting the upper and lower limits of supplied ore in order to produce a compromising schedule.

$$P_{min}^{tm} \leq \sum_{i=1}^{N} Q_i \,.\, x_i^{tm} \leq P_{max}^{tm} \;\forall\; t = 1,2,\dots,T; m = 2,3,\dots,M \tag{3.6}$$

Where:

P_{min}^{tm} represents the minimum processing capacity of ore material type m sent to the plant during period t; and

P_{max}^{tm} represents the maximum processing capacity of ore material type m sent to the plant during period t.

3.3.5 Reserve constraint

The ultimate pit comprises the set of blocks that allow for an economic mining extraction. Each block must be mined fully to yield the expected value. The reserve constraint controls the production scheduling model to avoid any rescheduling for the same block. The sum of removing portions of a block in one period t or more must not exceed the unit to avoid block duplication in the model. Equation 3.7 expresses reserve constraint to the OMPS objective function.

$$\sum_{i=1}^{N} x_i^{tm} \leq 1 \;\forall\; t = 1,2,\dots,T; m = 1,2,\dots,M \tag{3.7}$$

3.3.6 Slope constraint

Slope angle is an essential parameter for the optimisation of the pit limits and the production scheduling in the mine planning process. During the production scheduling, the slope angle guides the block extraction sequence and serves to determine the set of block predecessors for each individual block. A block as a reference point for expanding the pit requires the removal of its predecessor blocks all the way to surface. The set of predecessor blocks constrains the extraction of the concerned block and forces the production scheduling to proceed with overlying blocks before its extraction. Equation 3.8 helps to find the set of block predecessors of each block in the model. The scheduling time period t for any block except the blocks that have direct contact with the surface comes after the scheduling time of the block predecessors.

$$\sum_{i=1}^{N} x_i^{tm} - \sum_{r=1}^{t} x_l^{rm} \leq 0 \;\; \forall\, l \in P_i; t = 1,2,\ldots,T;\; m = 1,2,\ldots,M \qquad (3.8)$$

Where:

x_l^{rm} is the proportion of material of type *m* from block *l* mined during period *r*;

P_i is the set of block predecessors of block i; and

l identifies all block predecessors P_i of block i, extracted in period r, anterior or equivalent to time period *t*.

It is noticeable that solving such a problem in an optimal way can be very challenging and difficult. The OMPS is not a straightforward problem to solve. However, it requires extensive computational capacity especially for an ultimate pit containing thousands to millions of blocks that have to be scheduled over a long-term plan for extraction.

3.4 Optimisation techniques used for solving the OMPS problem

3.4.1 Introduction

Regarding the importance of the mine planning process before the start of the mineral exploitation, production scheduling has drawn much attention of many researchers to come up with approaches and techniques to optimise the OMPS plan. There exist techniques to solve the OMPS problem that are grouped into two categories namely: exact or deterministic methods, and heuristic methods (Osanloo *et al*., 2008). These techniques belong to the professional scientific discipline called operations research (OR). The OR techniques use mathematical models and algorithms for managerial decision-making and optimisation in organisations.

3.4.2 Brief historical overview of operations research

OR provides enough mathematical tools nowadays incorporated in many software packages used to address real-world problems in different fields. Although the application of OR has been broadly acknowledged during World War II by the

British Royal Air Force and Army (Musingwini, 2016), the roots of OR are traced even back to the early 1800s (Lyeme and Selemani, 2012). The starting point was the application of scientific analysis to the methods of production and management. Satisfactory success of OR applications in military operations was achieved where the objective was to find the best way for the allocation of limited resources to the military operations and activities (Lyeme and Selemani, 2012). The OR techniques have attracted the attention of many fields to aid decision-making and solving complex problems. The implementation of OR techniques quickly spread from military to government, social, industrial and economic planning (Lyeme and Selemani, 2012).

In the mining industry, OR techniques were introduced in the early 1960s to optimise diverse problems that arise in the mine planning activity. Many researchers acknowledge the work of Lerchs and Grossmann in 1965 as a pioneer work for the use of OR in the mining industry. Their work uses dynamic programming to resolve the open-pit mine limits problem under slope constraints. Other optimisation problems that have shown exceptional interest in the use of OR include mine design, cut-off grade, production scheduling, truck dispatching and equipment selection (Newman *et al.*, 2010). Kwiri and Genc (2017) acknowledged the use of optimisation techniques as being crucial to the profitability of mining operations.

3.5 Deterministic algorithms applied on OMPS problem

OMPS is one of the most complex problems along the mine value chain that have benefitted from the application of many optimisation techniques. These techniques have considerably improved mining operations which are characterised by uncertainties and market volatilities. Since the introduction of OR techniques in mining, it has been known that production scheduling can be modelled using integer programming (Chicoisne *et al.*, 2012). The work of Johnson (1968) is acknowledged as the first attempt to solve the OMPS problem through the application of exact algorithms. Models developed by Johnson seek to maximise the objective function while taking into consideration a set of assumptions including grade control, mining capacity, processing capacity, reserve constraint and slope angle constraint.

However, real-world instances of OMPS problem have caused difficulties in the implementation of exact algorithms for the optimisation of OMPS long-term plans. These challenges have led to the use of heuristic approaches. Samavati *et al*. (2018) emphasised that the impracticability of exacts algorithms is mainly due to the large number of constraints and the large number of blocks within the pit to be scheduled for a period of more than 5 years.

Section 3.6 describes and highlights the applications of well-known deterministic algorithms used to solve the OMPS problem. These techniques include linear programming, integer programming and mixed integer programming. A summary of these mathematical programming approaches is given in Table 3.2, which presents a descriptive, but not exhaustive list of deterministic methods.

Table 3.2 Summary of deterministic algorithms applied in solving the OMPS problem

Researcher	Year	Algorithm	Solution method	Constraints	Advantages	Main drawbacks
Johnson	1968	Linear programming	Dantzig-Wolf decomposition method	• Grade control; • Mining capacity; • Processing capacity; • Reserve constraint; and • Slope angle constraints.	• Considers dynamic cut-off grade; • Can hold large deposits; and • Generate optimum results for each period individually.	• Fails to solve completely the OMPS long-term problem; • Some blocks will be extracted partially, creating suspension in the air of un-extracted part of the block; • Has very limited applicability; and • Generates too many extended variables.
Roman	1974	Dynamic programming	Complete enumeration	• Mining capacity; • Milling capacity; and • Slope angle constraints.	• Optimises block sequencing and pit limit simultaneously.	• Does not satisfy mining and milling constraints; and • Cannot be implemented on large deposit.
Gershon	1983	Mixed integer programming		• Precedence block constraint; • Mining capacity; • Plant capacity; and • Grade control.	• More practical than Johnson's model in block sequencing; • Its mathematical formulation has the potentiality to reduce computer time; and • Only one constraint per block is required.	• Develop difficulties on large deposit instances; and • Shows inability to handle dynamic cut-off grade.
Dagdelen and Johnson (cited in Osanloo *et al.*, 2008)	1986	Integer programming	Lagrangian relaxation and sub-gradient multiplier	• Production capacity; and • Blending policy.	• Does not lead to fractional block extraction; • Exploits the structure of the block-sequencing problem; • Reduces the size of problem through relaxation; and • Adjusts the Lagrangian multipliers for blocks on the sides of the pit.	• Faces difficulties when implemented on large deposits; • Inability to handle dynamic cut-off grade concept; • Prevailing gap problem; • Demonstrates a need for creating pushbacks prior to block extraction sequence; • Uses a discrete approach to solve the OMPS problem, which is a continuous approach; and • More than one iteration required for the creation of the initial fundamental tree.

Researcher	Year	Algorithm	Solution method	Constraints	Advantages	Main drawbacks
Tolwinski and Underwood	1992	Dynamic programming	Dynamic programming, Artificial Intelligence and heuristic rules.	• Operational constraints; and • Slope angle constraints.	• Delivers practical solutions; and • Ability to incorporate many constraints.	• Cannot be implemented on large deposits; • Inability to handle dynamic cut-off grade; and • Does not guarantee optimum solution.
Tachefine and Soumis	1997	Maximal closure graph	Lagrangian relaxation	• Resource constraints: resource availability and consumption of resource; and • Precedence arcs constraints.	• Delivers solutions close to optimum; and • Applies heuristic to transform unfeasible closures into feasible closures.	• Unfeasible solutions obtained in later iterations violate relaxed constraints; and • Generates too many subproblem variables.
Akaike and Dagdelen	1999	Integer programming	4D-network relaxation and sub-gradient	• Grade control; • Production capacity; and • Blending policy.	• Takes into account the dynamic cut-off grade concept; and • Demonstrates the ability to handle stockpile option.	• Prevailing gap problem; and • Faces difficulties when implemented on large deposits.
Erarslan and Celebi	2001	Dynamic programming algorithm	Directed graph, simulative optimisation approach and heuristic rule	• Mining extraction sequence; • Stockpile and waste dump grade control; • Head grade control; • Mining and milling capacities; and • Mining width.	• Delivers simultaneously optimum pit limits and production scheduling; • Provides realistic estimate of the mining cost; • Considers all kinds of operational constraints; and • Considers the effect of pit volume on the unit cost.	• Encounters difficulties when applied to large sized instances; and • Does not reach optimal solution.
Caccetta and Hill	2003	Integer programming	Branch-and-cut	• Volume of material extracted per period; • Mill feed and mill capacity; • Processing different ore types; • Blending constraint; • Logistic constraints; • Stockpile related constraints; • Maximum vertical depth; and • Minimum pit bottom width and stockpiles.	• Considers all kinds of operational constraints; • Obtains a good solution in medium-size deposits; and • Provides mining engineers confidence with the results delivered.	• It cannot be implemented on large deposits; and • Inability to handle dynamic cut-off grade concept.
Ramazan *et al.*	2005	Integer programming	Fundamental tree generation	• Production capacity; and • Blending policy.	• Elimination of gap problem;	• Generating pushback is required before scheduling;

Researcher	Year	Algorithm	Solution method	Constraints	Advantages	Main drawbacks
					• Reduces the binary variables; and • Delivers higher NPV compared to any other deterministic algorithms.	• Generates fundamental trees after more than one iteration; • The difficulties faced in the implementation reduces its popularity; and • The optimality of the solution depends on the optimality of generated pushback.
Gleixner	2008	Mixed integer programming	Lagrangian relaxation	• Reserve constraint; • Precedence feasibility; • Mining activities; and • Processing activities.	• Attempts to aggregate processing variables in the LP; and • Reduce the OMPS problem size while preventing any loss in the NPV.	• No explicit formulation of the MIP has been proposed; and • Large number of variables is needed to model the production plan.
Chicoisne *et al*.	2012	Linear programming	Knapsack	• Block predecessor constraint; • Extraction time period; • Processing constraints; and • Extraction constraint.	• Can tackle real-sized instances in practical time; and • Can lead to new production scheduling systems, which does not need to schedule bench-phases.	• Cannot accurately capture block precedence; • Blocks scheduled in the same period may be scattered throughout the mine; and • May lead to an impractical solution from an engineering point of view.
Behrang *et al*.	2014	Linear programming	Clustering technique	• Processing capacity; • Mining capacity; • Block extraction precedence constraint • Mining processing portion; • Reserve constraint; • Over and under production constraint; and • Upper and lower limit for stockpile.	• Finds the block extraction sequence and their respective destination under grade uncertainty; • Introduces the cost of uncertainty in the OMPS long-term problem; • Reduces the number of variables to make the problem tractable; and • Allows the plan to extract extra ore at early stages of the life of the mine.	• The use of a large number of mining cut does not improve the NPV; and • Faces weaknesses for real planning applications.
Moosavi and Gholamnejad	2015	Mixed integer programming	Stochastic model	• Mining slope; • Grade blending;	• Integrates tonnage uncertainty;	• Cannot eliminate the tonnage uncertainty; and

Researcher	Year	Algorithm	Solution method	Constraints	Advantages	Main drawbacks
				• Ore production capacity; and • Mining capacity.	• Reduces geological risk in the OMPS long-term plan; • Delivers a better result compared to that of the LP; and • Targets higher ore grade in the first periods of mining extraction.	• Tonnage and grade uncertainty can lead to considerable deviation from production target.
Samavati	2017	Linear programming	Adding cut technique	• Precedence constraints; • Operational constraints; • Capacity limitation; and • Resource constraint.	• Converges quickly to a near optimum feasible solution; • Can enhance the production schedule plan obtained with LP; relaxation approach; • Shows a remarkable flexibility compared to other successful algorithms; and • Can potentially be modified to contain minimum resource requirements.	• Difficulties in solving extremely large instances of OMPS problems.

3.6 Description of some deterministic algorithms applied on OMPS problem

3.6.1 Linear programming

Linear programming can be regarded as one of the early mathematical descriptions using the block concept for the OMPS solution developed by Johnson in 1968. The formulation of the model proceeds through the partition of the large production plan into sub-problems then elementary problems for a better understanding of the model. The principle goal that governs the OMPS problem is embodied in the main objective function to maximise. The factors that constrain the block extraction sequence represent the sub-problems. Moreover, the properties of each sub-problem represent the elementary problems for which the model is given. For the production scheduling problem, solving all sub-problems consist of maximising the objective function in the presence of constraints (Johnson, 1968). Therefore, the LP model selects the best scenario plan making it the most feasible schedule that yields the optimal solution under constraints. Johnson's model has had much application due to its flexibility and simplicity. However, at the time of implementing the model, Johnson lacked the computational experience to evaluate the computation time of the model despite the reduction of the problem size (Djilani, 1997).

3.6.2 Mixed integer programming

The mine scheduling problem can also be solved using the mixed integer programming (MIP) model. Gershon (1983) first developed the formulation of the MIP for solving the long-term production plan. The MIP is meant to improve the LP by adding new additional decision variables in Johnson's model in order to allow the mining of partial blocks. It includes the following variables: percentage of the block mined in the mining period, the portion of the block left at the beginning of a given period, the percentage of the precedent blocks mined at the beginning of a given period and a constraint that shows whether all precedent blocks have been mined completely in a given period. These decision variables try to resolve the overlying blocks prevailing in the LP method and thus, allow a more practical extraction sequence than that generated with LP. The model may be applied to solve only aggregate block models that reduce the number of blocks, hence the number of constraints. Despite improvement, Gleixner (2008) admitted that no explicit formulation of the MIP has been proposed in the resolution of the long-term mine production scheduling problem.

3.6.3 Integer programming

The integer programming (IP) problem is a mathematical modelling that restricts all the decision variables to be an integer. It aims to discard the fractional part of the LP variables to make them integers. The first use of IP for the optimisation of block scheduling problem can be traced back the work of Dagdelen and Johnson in 1986 as cited in Osanloo *et al*., 2008. The implementation of the IP was motivated by the directed graph in which the block model can be represented. In this representation, nodes correspond to the blocks while arcs refer to the precedence relationship of blocks. The graph denotes a discrete approach, which fits best for the implementation of the IP (Rosas *et al*., 2016). The model solves a production sequence problem providing the complete extraction of a block during one period. This representation aligns with the aggregated block model but faces difficulties in solving the original sized block model. Consequently, the IP becomes more and more impractical with the increase in size simultaneously of the open-pit mine, the block dimensions and the block aggregate (Gleixner, 2008).

3.7 Heuristic algorithms applied on OMPS problem

Johnson (1968) introduced the application of exact algorithms to optimise the long-term production plan. The optimal solution produced by these exact algorithms may maximise the objective function as stated in the previous section but fail to define the mining extraction sequence of a large-scale mine. The mine production schedule optimisation must comply with a set of physical, operational and geological constraints (Khan and Niemann-Delius, 2015). For a large number of thousands to millions of blocks inside the final pit limit, these exact models may contain millions of constraints and integer variables that reduce their practicability (Khan, 2018). This has been one of the general issues among others faced using the mathematical formulations. The set of weaknesses of the exact methods have led to the use of heuristic methods that deal with such complex problems by mimicking the natural behaviour of some creatures like birds, ants and rabbits (Chicoisne *et al.*, 2012).

The following sub-section briefly describes the most popular heuristic techniques that have been applied in mining for the optimisation of the OMPS plans. These techniques include ant colony optimisation, simulated annealing and particle swarm optimisation. A summary on heuristic techniques is presented in Table 3.3.

Table 3.3 Summary of heuristic techniques applied in solving the OMPS problem

Researcher	Year	Algorithm	Solution method	Constraints	Advantages	Main drawbacks
Rovenscroft	1993	Risk analysis using deterministic algorithms	Conditional simulation technique and deterministic algorithms	• Grade control; • Mining capacity; • Processing capacity; and • Block precedence.	• Shows the impact of grade uncertainty on long-term production planning (LTPP).	• It cannot quantify the risk of a project; • It does not generate an optimal solution in the presence of grade uncertainty; and • Use of repeated simulations as successive input to a mine scheduling process is cumbersome.
Denby and Schofield (cited in Osanloo *et al.*, 2008)	1994	Genetic algorithm	Heuristic algorithm	• Mining capacity; • Processing capacity; • Average grade control; and • Extraction sequence.	• Can optimise production scheduling and pit limits simultaneously; • The model converges quite quickly to the near-optimal solution; and • The model shows a remarkable flexibility to accommodate changes.	• The effect of pit volume on the unit is not considered; • Results produced largely differ from one run to another; and • Does not guarantee an optimal solution.
Kumral and Dowd	2005	Simulated annealing	Multi-objective SA and Stochastic relaxation	• Access constraint; • Sufficient number of satisfied periods; • Mill and plant capacities; and • Ore content.	• Can improve sub-optimal solution obtained by deterministic methods; • Provides a solution that minimises the extracting cost; and • Uses Lagrangian relaxation to reduce the size of the problem.	• Lagrangian parameters reduce the revenue of some ore blocks; • Susceptible to be trapped on a local optimum solution; and • Does not reach the maximum profit.
Ferland *et al.*	2007	Particle swarm optimisation	Resource-constrained project scheduling problem	• Extraction capacity; and • Time horizon.	• Generates better solution if the initial population solution is of good values; • Uses sub-solution generated from conventional methods as initial solution; • Able to accommodate large size population solution to deliver a near-optimal solution; and	• Does not seem to be sensitive to the input parameter values of the model; • Quality solution depends on the quality of initial values of particles in the swarm; and • Indicates no clear impact of modifying the values of some parameters.

Researcher	Year	Algorithm	Solution method	Constraints	Advantages	Main drawbacks
					• Generates a large number of different solutions.	
Consuegra and Dimitrakopoulos	2009	Simulated annealing	Metropolis algorithm	• Number of mining sequence; • Slope angle; • Ore production rate; and • Waste production rate.	• Able to optimise simultaneously the pit limits and production scheduling; • Blocks are assigned according to the probability rank; and • Integrates grade uncertainties in the realisation of the production schedule.	• Loses the sensitivity when the number of input mining sequence exceeds 10; and • Can be labour intensive compared to standard conventional techniques.
Khan and Niemann-Delius	2015	Particle swarm optimisation	PSO formulation	• Reserve constraints; • Slope constraints; • Mining capacity constraints; and • Processing capacity constraints.	• Can produce better quality solution in a relatively shorter time compared to deterministic techniques; • The model is well structured and quite flexible; • Can accommodate multiple variable constraints such as variable slope angle, grade and market uncertainty; and • Delivers a very small standard deviation value demonstrating its robustness.	• Susceptible to get trapped in a local optimum solution; • Local variants show relatively slower rate convergence; • The computation time depends on the input parameters and the size of the problem; and • The standard deviation tends to increase with problem size.
Shishvan and Sattarvand	2015	Ant colony optimisation	Max-Min ant system and ant colony system	• Mining capacity; • Processing capacity; • Average grade; and • Variable slope angle.	• Strong capability to improve the quality of the initial sub-optimal solution; • Can optimise simultaneously the pit limits and the OMPS problems; • Capability of considering any type of objective function, non-linear constraints and real technical restrictions; • Able to consider variable slope angle;	• Does not guarantee to reach the global optimum schedule; • The quality of solution largely depends on the quality of the input parameters; • The procedure is not always proven mathematically to reach the optimal solution; and • Small perturbation occurred in the control of the size of pushbacks.

Researcher	Year	Algorithm	Solution method	Constraints	Advantages	Main drawbacks
					• Can incorporate some of the constraints into the objective function; and • The computation time is less sensitive to the size of the model.	
Liu and Zhang	2016	Particle swarm optimisation	Improved PSO	• Mining horizons; • Mining capacity limits; • Mining sequence requirement; and • Sloping requirement.	• The solution time is faster than of the deterministic approach; and • Delivers good solution with an average error of 2%.	• Does not guarantee an optimal solution; and • Some particles do not update their historical original information over many iterations.
Alipour	2017	Genetic algorithm	Normalisation method	• Access constraints; • Mining capacity; • Processing capacity; and • Block conservation constraints.	• Demonstrates its efficiency to solve the OMPS problem; • Able to generate feasible solutions in 2D space; • Delivers an NPV solution with a gap of less than 5% to the optimum solution; and • Computing time Relatively shorter.	• Does not convince to be able to handle large number of blocks; • May face challenges on a real-world problem; and • Cannot be implemented on a 3D block model.
Khan	2018	Particle swarm optimisation	Global PSO and multi-start PSO	• Reserve constraints; • Slope constraints; • Mining capacity; • Stochastic processing capacity constraints; and • Stochastic metal production constraints.	• Able to handle the grade-related uncertainties in a more efficient way; • Can generate sufficiently good solution to the problem at hand in a shorter computing time; and • Quantifies penalties when violating capacity constraints.	• During the process, the particle positions and velocity are initialised to their initial values; • Likely to get trapped in a local optimum solution; and • Due to the one-dimensional nature of the model, the solution may not be feasible in terms of the required slope angles.

3.8 Description of some heuristic algorithms applied on OMPS problem

3.8.1 Ant colony optimisation algorithm

The idea behind the ant colony optimisation (ACO) originated from the observation of the behaviour of a group of ants in the search for food (Sattarvand, 2009). The foraging behaviour of ants characterised by finding the shortest path within a colony that leads to the food source has been an inspiration to solve the OMPS long-term problem. Considering the location of food as the optimal solution, ants within a colony move toward discovering the food on which all the ants desire to reach.

Initially, ants travel randomly in the search for food having no clue of the location of the food. While travelling around, ants lay down a chemical substance called pheromone to trace their paths. The pheromone allows ants to transmit a message to other members to reduce the random manner of travelling within the search space (Sattarvand, 2009). Once one ant finds food, the rest of them tend to follow its path to reach the food location. The transmission of the message of finding food enables others to converge to the food source. However, the pheromone trails vanish over time making the longer path to the food source disappear while the shorter path increasing the amount of the pheromone as per number of ants trafficking along (Shishvan and Sattarvand, 2015). The shorter path, therefore, represents the best route to reach the optimal solution and remains the only way in use to reach the food location in a relatively shorter time.

3.8.2 Simulated annealing algorithm

The fundamental idea of the simulated annealing (SA) algorithm is based on the analogy of thermodynamics. In this process, metallic, glass and plastic solids are heated up sufficiently followed by a gradual cooling down to allow the rearrangement of atoms and molecules in a new configuration. The SA algorithm consists of improving an initial solution to its optimal level by applying temperature and cooling down parameters in the system. The algorithm starts with a random solution, which is stirred up to create a new solution potentially better than the initial solution with respect to the constraints. The gradual lowering of temperature continues until the approximately optimum solution has been reached. However, a poor setting of initial parameters might not improve the sub-optimal solution making the system to run for a shorter time. In this case, a probabilistic decision on validating the new solution is made based upon the current parameters of the system.

The critical parameters for an efficient SA optimisation are initial temperature, cooling rate, number of iterations and final temperature (Nogholi, 2015). A low initial temperature makes the convergence of the process toward the optimal solution faster, but a high starting temperature and low cooling rate slow down the process. The algorithm search process may exit the potential local-optimum solution space quite early when proceeding at a faster cooling rate, thus skipping the global optimum solution. Therefore, the cooling rate and initial temperature should be selected at values that enable taking advantage of using a good sub-optimum solution (Kumral and Dowd, 2005). This heuristic idea has found interest in mine planning for solving the OMPS combinatorial optimisation problem.

3.8.3 Particle swarm optimisation algorithm

The particle swarm optimisation (PSO) algorithm finds its origin from two main aspects namely, natural life and evolutionary computation. Natural life refers to bird flocking and fish schooling behaviour, while evolutionary computation is related to both evolutionary programming and the genetic algorithm (Kennedy and Eberhart, 1995). The algorithm conducts the searching process using a population of particles. Consider a flock of birds searching for corn in a finite space where none of them has any indication of food location. The flock finds food in a cooperative way where each bird adjusts its position continually by flying in a multi-dimensional space according to its own experience and according to the flying experience of its relative members. Within the swarm, each particle or bird represents a potential solution to the optimisation problem. The algorithm starts with the allocation of random position and random velocity to each individual particle in the swarm. Each particle is composed of three vectors and two fitness values for storing the temporal individual optimum position of a particle and the global optimum position on every iteration of the algorithm.

The implementation of the PSO in the optimisation of the open-pit production scheduling starts with random initialisation of the feasible solution consisting of blocks that are potentially mineable in a given mining period. The process updates the solution by selecting a block randomly, which is later assigned to the current mining phase (Khan and Niemann-Delius, 2014). The selection of blocks proceeds with respect to block precedence constraints to avoid any unpractical mining schedule. Before moving to the next period, the process continues until the blocks selected for the current period satisfy the mining and processing capacities requirements. The algorithm terminates the process when there exist no more free blocks or when all the mining periods are covered (Khan and Niemann-Delius, 2014).

3.9 Commercial software for open-pit mine production scheduling

The utilisation of software along the mine value chain has played a vital role to improve the efficiency in the mine planning process. However, the mining software packages incorporate in their functionalities, algorithms that have been highlighted in the previous section. In the mine planning activity, software packages are mainly used for geological data management, geological modelling and resource estimation, design and layout, production scheduling, financial valuation and mine optimisation (Katakwa *et al*., 2013). The importance of solving mine production scheduling has attracted the attention of many mining software developers. Genc (2015) listed 67 mining software packages of which about 46% pertain to production scheduling. Table 3.4 illustrates the most used software for scheduling in open-pit mine production.

Table 3.4 Description of some production scheduling software packages

Software package	Description
CADSMine	CADSMine is a Microstation based mine design and scheduling solution. It provides mine planners with the ability to design and schedule very complex mine layouts and to generate multiple scheduling scenarios by varying numerous input parameters. CADSMine integrates 3D mine layout design and mine-wide layout scheduling in the same 3D space (Gürtunca, 2003).
Dragsim	Dragsim is an advanced dragline simulation system designed to optimise equipment productivity and waste movement. It assists users to obtain practical productivity and production cost data with speed and precision (RPMGlobal, 2018).
EPS (MineRP)	EPS is a product of MineRP that sets the industry standard for mine production scheduling and forecasting. It addresses the shortcomings of traditional project scheduling and spreadsheet-based solutions by providing a logical and unified approach to mine scheduling (MineRP, 2012).
MineSched	MineSched is one of the most innovative scheduling software that has been used for all types and sizes of open-pit and underground mines to deliver complete short-term and long-term production schedules. It can schedule from pre-created block model, grid and polygon models that originate from many different mine planning systems, including GEOVIA Surpac, GEOVIA Minex, GEOVIA GEMS and other industry geology and mine planning products (GEMCOM, 2013).

Software package	Description
Studio 5D Planner	Studio 5D planner is a product of Datamine Studio for mine design and project scheduling. It provides complete design and scheduling products for short and medium-term planning of open-pit mines with respect to their 3D design and sequences (Datamine, 2015).
MRM	MRM software is a product of the MineRP that provides an integrated mining business solution for the management and control of the mine's orebody and mining activities. MRM is a software that is able to perform quite a number of activities in mine planning including geological data management, geological modelling and resource estimation, design and layout, scheduling, financial valuation and mine optimisation (MineRP, 2012; Genc, 2015).
Xpac	Xpac is a product of RPMGlobal that is used to plan production. Xpac makes it easy to view mine data, from individual reserve blocks through to the whole mine. It provides the users the ability to identify where to dump waste material and what equipment is needed to meet production targets (RPMGlobal, 2018).

3.10 Chapter summary

The objective function of the OMPS long-term optimisation is to maximise the NPV of the undiscounted cash flows of the mine by providing the time horizon through which blocks in the pit are extracted. It expresses the sum of the block BEV scheduled in each mining period. The optimisation of the objective function required taking into consideration the following constraints:

- Grade control that enables a smooth process of the mineral material at the subsequent stages in the mine value chain;
- Mining capacity that constrains the OMPS problem or objective function in such a way that the tonnage of ore blocks scheduled for extraction per period meets the required production rate of the mine;
- Processing capacity that constrains the OMPS objective function by setting the upper and lower limits of supplied ore in order to produce a balanced production schedule;
- Reserve constraint that controls the production scheduling model to avoid any rescheduling for the same block; and
- Block precedence relationship that constrains the extraction of the concerned block and forces the production schedule to proceed with overlying blocks before its extraction.

OMPS optimisation techniques find their origin in the discipline of operations research. These techniques are classified as deterministic or heuristic methods. Deterministic methods face challenges of impracticability due to the complexity of the OMPS problem. Deterministic techniques discussed in this chapter include linear programming, mixed-integer programming and integer programming that fail to produce an optimal and feasible OMPS plan. Therefore, the introduction of heuristic methods was intended to overcome the challenges faced with deterministic methods. Heuristic methods are inspired by the behaviour of natural species such as birds, fish, ants and rabbits. The description of heuristic methods included the ant colony optimisation, simulated annealing, particle swarm optimisation, and the genetic algorithm. The following chapter explains in detail the genetic algorithm.

4 RESEARCH METHODOLOGY

4.1 Chapter overview

The methodology technique adopted in this research for solving the OMPS long-term problem is the genetic algorithm. This chapter describes in detail the genetic algorithm. The advantages of using the method approach are given as well as the fundamental reasons of why the genetic algorithm would be suitable in dealing with the problem. This is followed by adjusting the genetic algorithm into the context of the problem at hand and providing the algorithm procedure for solving the long-term open-pit mine production scheduling problem.

4.2 Brief history and origin of the genetic algorithm

Since the 1950s, several researchers have engaged in the study of the natural evolutionary process as an optimisation tool for solving engineering problems (Mitchell, 1999). Given a population of candidate solutions, the idea was to solve a problem by evolving the fitness of the population using techniques applied in the biological evolution. The natural phenomenon in the life of animal species has motivated computer scientists over the years in many domains which include neural networks, machine learning and evolutionary computation (Mitchell, 1999). In the 1980s, the development of evolutionary computation experienced a dramatic resurgence whereby the genetic algorithm was one of the most prominent techniques (Mitchell, 1999). Since the introduction of the genetic algorithm, many variants have been developed and applied in different areas driven by the advancement in computer technology. Nevertheless, the work of John Holland is widely acknowledged as the pioneering work on the genetic algorithms upon which other variants are built on.

John Holland initiated the development of the genetic algorithm in the 1960s with the help of his colleagues and students. Originally, the model was intended to study the natural occurrence of the phenomenon of adaptation and to develop a mechanism of how this should be imported into a computer system. Later, in 1975, Holland designed an algorithm based on the genetic process, known as the genetic algorithm (GA) for solving specific problems. In his work titled "*Adaptation in*

Natural and Artificial Systems", he presented a GA as an abstraction of biological evolution and proposed a framework for adaptation under the GA. This was the first attempt for computational evolution based on heuristic concepts (Mitchell, 1999).

The GA is a metaheuristic algorithm based on Darwin's theory of natural evolution to provide tools for global optimisation problems, with the idea of imitating natural evolution (Kumral, 2004). The theory of evolution supports the idea in which inherent traits are transferred from one generation to another through genetic reproduction. The procedure follows the evolutionary method that mimics the natural selection process of evolution endorsing the principle of "survival of the fittest" (Holland, 1975). Suppose a population of natural species is exposed to environmental threats, from which they must resist and spare their lives according to their fitness. Then relatively fitter species will survive the deadly threats while unfit species will lose their lives. Over many years, the number of unfit or weak species will decrease, as only a few of them survive just because they are lucky. Consequently, the average fitness of the population is bound to improve generations after generations. The resulting generation after many years will consist of fitter and stronger species compared to the initial generation. This is the evolutionary process of which the genetic algorithm has been inspired from.

This improvement of the GA uses a population consisting of individuals represented in the form of strings or chromosomes depicting the genotype characteristics of individuals. The latter inherit the genotype from their respective parents, and the fitness is directly linked to the parents' fitness. GA seeks to evolve the fitness of the offspring through the application of some genetic operators such as selection, crossover, mutation and inheritance (Alipour *et al*., 2017).

4.3 Concept of the genetic algorithm

The GA is part of the evolutionary algorithms that use biological evolution and hereditary (natural genetics and natural selection) development as a strategy to solve difficult problems. It belongs to the group of metaheuristic algorithms that mimic the natural behaviour of some species to find an optimum solution beyond human understanding (Mitchell, 1999). Metaheuristic algorithms have their place in global optimisation techniques that explore properties gathered by the algorithm to select which candidate solution should be considered in the next generation. Additionally, they use probabilistic means to reach quickly the solution in a relatively shorter time compared to the deterministic algorithms. Being a probabilistic algorithm, GA does not guarantee a global optimum solution, but assists in getting a near-optimal solution to problems impossible to be solved with deterministic algorithms (Verhoeff, 2017).

The GA technique is a random global search and optimisation model in a space of potential solutions, where it improves its solution through the evolution of the population fitness. The problem is expressed as an objective function to be maximised or minimised under certain conditions. The search starts with a population of random solutions where each chromosome (individual) represents a potential solution. Chromosomes denote a binary string of ones and zeros or also real numbers, of which the value in the objective function defines their fitness. The algorithm evaluates and ranks each chromosome according to the fitness value. In the maximisation of an objective function, the higher the value, the more likely it is to be selected to participate in the reproduction of the new population of solutions. In other words, the probability of selecting parent chromosomes is proportional to their fitness.

Two parent-solutions are paired using the crossover concept to reproduce new solutions that converge towards optimality. The crossover exchanges some features from the parent solutions to constitute new solutions of a mixture of parents' traits. The procedure also uses mutation and elitism to speed up the process towards reaching a better solution. The mutation must identify the weakest genes in the chromosome and improve by replacing them with stronger genes. As the aim is to improve the average solution quality, elitism seeks to select the best among the strongest, the better are the solutions the faster is the convergence to the optimal solution. The crossover, mutation and elitism operations are carried out on the offspring solutions repeatedly until reaching the optimum point. The algorithm terminates the search process when there is no improvement realised on the pit layout solution after a certain number of iterations or generations.

4.4 How the genetic algorithm works?

After expressing the problem to be solved as a fitness function (objective function), either to maximise or minimise, and then representing the model of the candidate solution as a bit string (Boolean or real numbers); a simple GA proceeds as follows (Mitchell, 1999):

- Step 1: Create randomly an initial population of candidate solutions $G(t) = \{x_1^t, x_2^t, \dots, x_N^t\}$ in the form of a chromosome. Each candidate represents a potential and feasible solution to the problem.
- Step 2: Evaluate the fitness value $f(t)$ of each candidate from the fitness function. Rank all the candidates according to their fitness, from the fittest to the least fit.
- Step 3: Create a new population of solutions by applying the following genetic operators:
    - Selection: select $m \leq N$ fittest solutions from the current population as parent chromosomes to participate in the reproduction of offspring. The probability of being selected is proportional to the fitness values.
    - Crossover: the selected parent chromosomes are paired to produce offspring. The crossover is done through the exchange of some genes between the parents.
    - Mutation: mutate one or more weak genes in the new chromosomes to improve their fitness. The mutation is inversely proportional to the genes' fitness values.
- Step 4: Replace the least fit solutions with the new better solutions to improve the average fitness of the new population, and then rank them in descending order of fitness values.
- Step 5: Go to Step 3 until the fitness values of chromosomes in the population are close enough to each other meaning that no more improvement can be made. The standard deviation of the fitness values is at its lowest level to end the process.

4.5 Genetic algorithm terminology and key concepts

It is important to understand how the genetic operators work during the implementation of the genetic algorithm in order to visualise the nature of the outcome. In addition, the different versions of the genetic algorithm are based not only on the adaptation of the procedure to the problem at hand, but also on how the genetic operators work. For instance, one version would consider real coding with binary or Boolean numbers instead of working with real numbers. Also, the resulting offspring from crossover and mutation in Step 3 may replace their respective parent chromosomes in one version rather than replacing the least fit chromosomes in another version. There exists different methods of crossover, mutation and inversion that also differ from one version to another. The following sub-sections describe the different methods of genetic operators and actions involved in the running of the genetic algorithm.

4.5.1 Population

In the genetic algorithm, a population or generation consists of a set of chromosomes (individuals) representing potential solutions to the problem (Verhoeff, 2017). The quality of the population should demonstrate strong variability in the individuals in order to extend the space solution. Generations evolve over time in terms of the fitness when parents pass their genetic traits to their children making a new generation fitter than the parents' one. The new population in the algorithm is formed through the selection of the stronger chromosomes from the parent population and the offspring population. After several iterations, the algorithm converges to the best individual that is considered as the solution to the problem.

4.5.2 Chromosome

A chromosome is a string of genes defining the features inherited from the parents. A gene represents a segment of the chromosome that contains a unit of genetic information (Mitchell, 1999). Each gene is located at a locus (position) on the chromosome. Most often, chromosomes are encoded as strings of ones and zeros, but also as real numbers depending on the nature of the problem. The value of the chromosome in the objective function defines its fitness.

4.5.3 Selection

The principle behind the genetic algorithm is essentially Darwinian natural selection. This operator selects chromosomes in the population according to their fitness for reproduction. The fitter the chromosome, the higher the chance that it is to be selected for breeding. The selection directs the genetic algorithm search towards promising regions in the search space. Several selection techniques have been proposed, among them include the roulette wheel, tournament selection, sharing, ranking and scaling, and steady-state reproduction (Mitchell, 1999).

The roulette wheel selection or fitness proportionate selection, as proposed by Holland in 1975, is the most common selection method. The goal is to determine the selection probability for each chromosome proportional to the fitness value. The selection probability is calculated using Equation 4.1. The roulette wheel is divided into a number equal to the number of chromosomes, where the size of the portions corresponds to the probability values. The wheel is spun a number of times equal to the population size, each time selecting a chromosome to take part in the reproduction phase (Mitchell, 1999).

$$p_i = \frac{f_i}{\sum_{j=1}^{N} f_i} \tag{4.1}$$

Where:

- p_i= probability of chromosome i for being selected;
- f_i= the fitness of chromosome i; and
- N= number of chromosomes in the population.

4.5.4 Crossover

Crossover is a reproduction operator that randomly picks genes from two parent-chromosomes and exchanges them to create new offspring. The goal is to combine two good solutions to generate two new better solutions. The average fitness of the new population should be better than the one for parent generation. The two best-known methods for crossover are single-point crossover and dual-point crossover. These methods are discussed in the following sub-sections.

Single-point crossover

Figure 4.1 illustrates an instance of a single-point crossover. For this method, two parent-chromosomes are selected and undergo a random or semi-random cut-point along the string (Verhoeff, 2017). This point represents the point from where the two strings must be split into two parts regardless of the nature of the genes. The algorithm applies the crossover to exchange the parts resulting in two new chromosomes (children), and both receiving genetic information from the two parents.

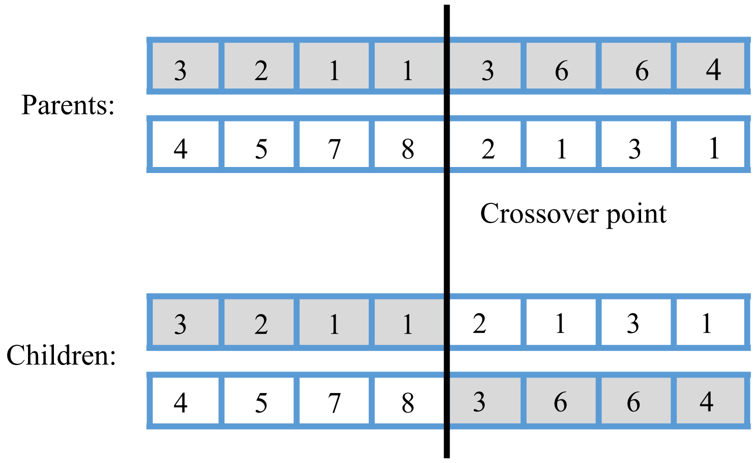


Figure 4.1 Illustration of a single-point crossover (Source: Verhoeff, 2017)

Dual-point crossover

This method imposes two cutting points on parent chromosomes to split them into three parts (Verhoeff, 2017). The new solutions are obtained by crossing over the parts in the middle from the parent strings. It has the advantage of being selective compared to the single-point method. Figure 4.2 shows an instance of the dual-point crossover.

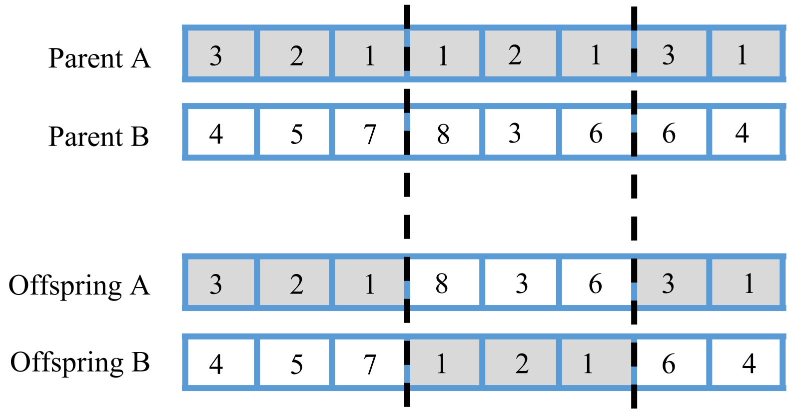


Figure 4.2 Illustration of a dual-point crossover *(Adapted from*: Verhoeff, 2017)

4.5.5 Mutation

This operator randomly flips some of the genes along the string to improve the fitness of a chromosome. Mutation can occur at any gene position in a chromosome with a probability of 1/L, where L represents the length of the chromosome (Verhoeff, 2017). The mutation operator prevents the algorithm from being stuck at a local optimum solution through the introduction of a random transformation. The following sub-sections present the different varieties of mutation.

Flip-bit mutation

Flip-bit mutation consists of selecting and changing one or more genes of a chromosome into the opposite number, usually applied in a binary string. For example, if the specific gene houses an allele of one, the flip-bit changes it into a zero and vice versa as illustrated in Figure 4.3 (Verhoeff, 2017). The idea is to eliminate weak genes from the chromosome to improve its fitness.

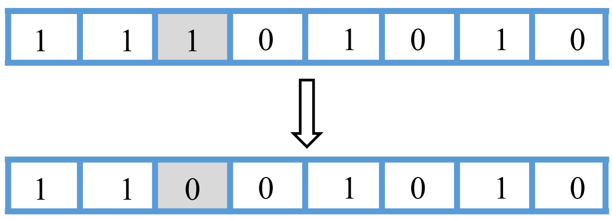


Figure 4.3 Illustration of a flip-bit mutation

Swap mutation

Figure 4.4 illustrates an example of a swap mutation. It consists of identifying two genes and swap their positions. Some genes can contribute massively to the fitness of a chromosome if they occupy certain positions other than their current position. This technique is suitable when using real number string compared to the flit-bit mutation. The swap mutation has the potential of improving significantly the fitness of real number string. However, it does not bring any effect if both genes have the same value (Verhoeff, 2017).

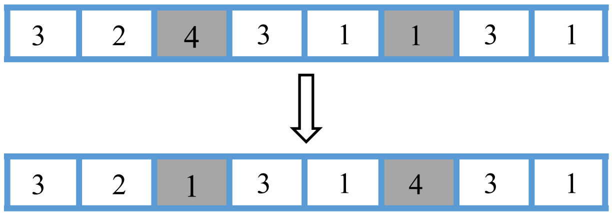


Figure 4.4 Illustration of a swap mutation

Scramble mutation

This type of mutation consists of selecting a number of adjacent genes and shuffle their positions randomly. The number of selected genes subject to undergo the scramble mutation depends on the coding algorithm and the length of the chromosome (Verhoeff, 2017). Figure 4.5 illustrates an example of a scramble mutation.

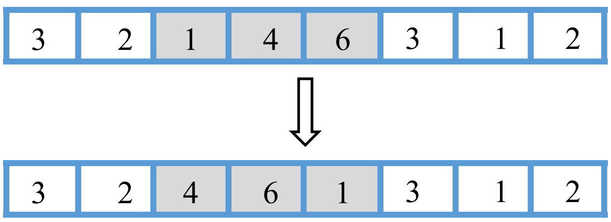


Figure 4.5 Illustration of a scramble mutation

Inversion mutation

Likewise, the scramble mutation, the modification in the inversion mutation is performed on a number of adjacent genes selected along a chromosome. However, instead of randomly shuffling them, the order in the string is simply reversed as shown in Figure 4.6 (Verhoeff, 2017).

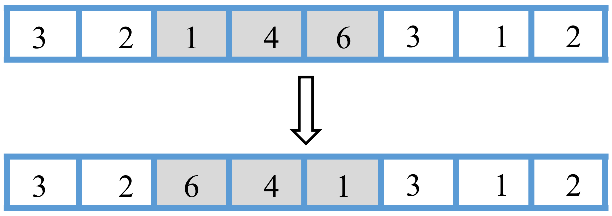


Figure 4.6 Illustration of an inversion mutation

4.6 Why the genetic algorithm is used for OMPS problem?

The views of researchers working on evolutionary algorithms, suggest that the process structure of the GA seems to be suitable for many of the most demanding computational problems of which the OMPS problem belongs (Alipour *et al*., 2017; Verhoeff, 2017). The OMPS problem would, therefore, necessitate a method of searching through a vast space of feasible solutions. The existence of such a space solution results from the various possibilities of block extraction sequence from the ultimate pit limit. Finding the optimal extraction sequence requires a mechanism that shows strong adaptability to incorporating several constraints, and a strong capability to handle a large number of blocks. However, the GA as an evolutionary technique provides a tool for addressing problems of such nature in a much-reduced amount of time when compared with other approaches (Alipour *et al*., 2017).

As the aim of this research study is to find the optimal OMPS long-term plan, the choice of implementing the GA was mostly motivated by the following reasons in addition to the above-mentioned advantages:

- The GA is considered to be less complex to be understood, easy to develop and to code in any programming language (Shishvan and Sattarvand, 2015);

- The process of evolution is known to be a successful robust method through the adaption of a biological procedure (Sattarvand, 2009);
- The GA has proved to be flexible, demonstrating a strong mechanism to incorporating constraints and uncertainties in the solution procedure;
- The use of crossover and mutation operators are suitable for exploiting and exploring a vast space of feasible solutions;
- The GA procedure structure helps to avoid an inherent tendency of getting stuck at a local optima solution (Verhoeff, 2017);
- The solution representation as chromosomes matches with the block model structure in the open-pit mine;
- The GA has the ability to deliver usable near-optimal solutions fast enough to problems that are impossible to be tackled with a human brain (Verhoeff, 2017); and
- The GA has a huge potentiality to optimise simultaneously the pit limit and the production scheduling in the mine planning process (Shishvan and Sattarvand, 2015).

4.7 Some applications of the genetic algorithm

With regard to the advantages provided by the GA, many researchers have applied the GA in various fields including DNA analysis, business properties, image processing, vehicle routing, neural networks, machine learning, game playing and robotics (Mitchell, 1999). In the mining industry, the approach has been also applied to solve a variety of problems, some of which are highlighted in the next paragraphs.

Ruiseco (2016) used the GA model to optimise dig-limits on a bench, given equipment constraints, grade control and mining and processing costs. After testing the algorithm on multiple case studies, it was found that the robustness and flexibility of the GA in solving such problems were acceptable.

Alipour *et al*. (2017) used the GA to optimise the production scheduling of a hypothetical copper orebody represented by blocks in a two-dimensional space. The objective function was subject to access constraints, mining capacity and processing capacity constraints. The experimental results showed the efficiency of the GA in handling the open-pit production scheduling problem, leaving a gap of less than 5% off the optimal solution.

Ahmadi and Shahabi (2018) implemented the GA for cut-off grade optimisation and measured its efficiency against Lane's method. The procedure was directed in terms of, firstly the optimal mineable grade at the beginning and at the end of the life of the mine. Lastly, the procedure was evaluated in terms of the NPV and the volume of the output per unit from mining, processing and refinery operations. The experiment indicated convergence in the results of the two models, but with a higher speed of computing and more accurate results generated by the GA.

4.8 Genetic algorithm in the context of the OMPS

The GA is subject to minor modifications on the structural procedure to adapt with the problem at hand, while the solution model of the problem must be represented in such a shape suitable for applying genetic operators. Therefore, in this research study, the following concepts were adopted for the optimisation of the long-term production scheduling of an open-pit mine:

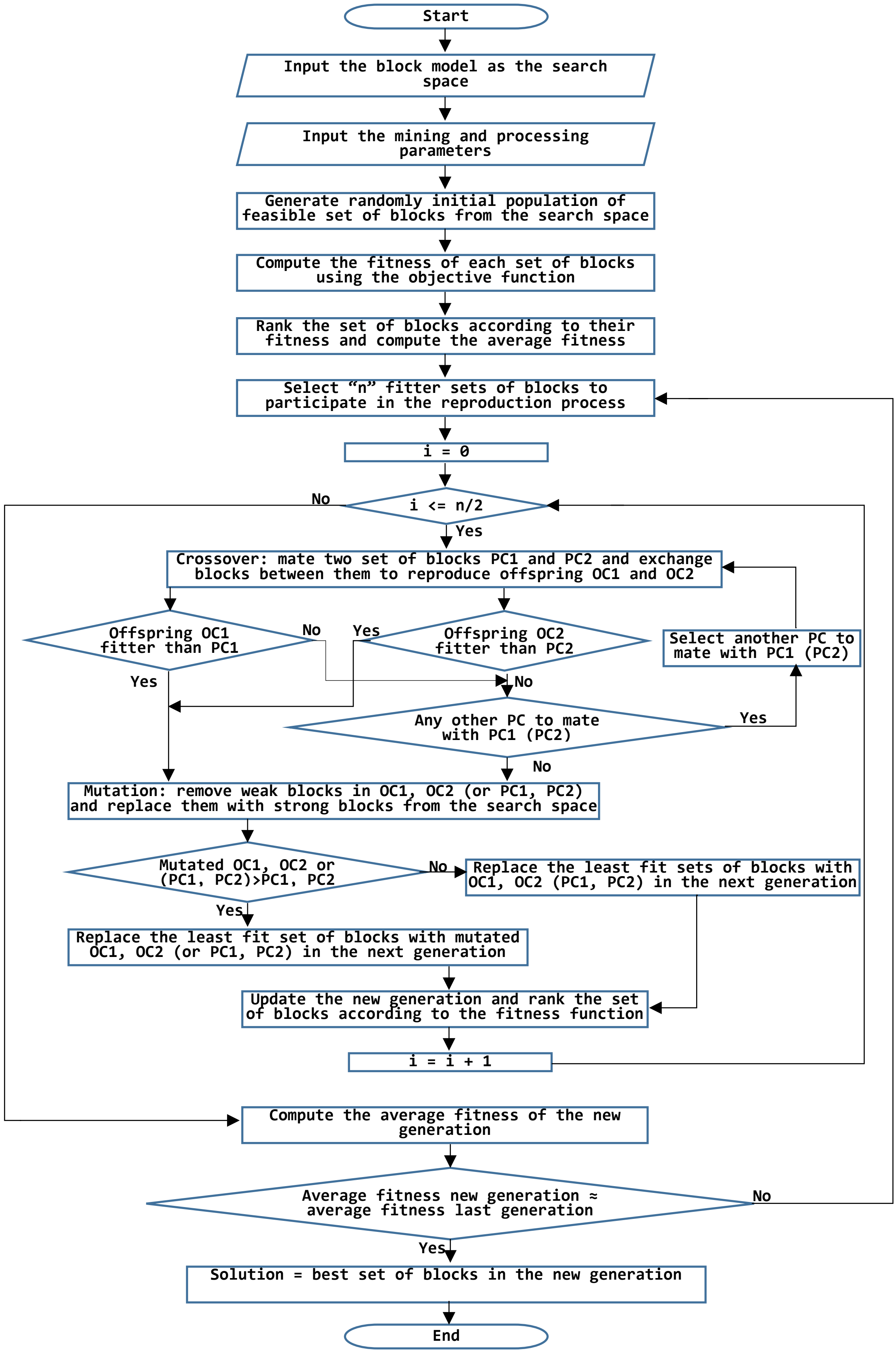
- Search space: the GA is set to seek for the most economical and feasible combination of mining blocks for each time period across a wide and finite search space defined within the ultimate pit limit. When the best combination of blocks is found for a period, these blocks should not be considered in the next iteration in order to comply with the reserve constraints to avoid the block duplication problem. As a result, the search space narrows with each iteration period after period;
- Population or generation: this is a pool of set of blocks representing potential solutions that might be scheduled for mining for a period of time. Each set of blocks is a combination of blocks practically feasible to be extracted. To comply with the genetic representation, each set of blocks is displayed in the form of a chromosome. In an open-pit mine, a feasible solution means a solution that complies with the block precedence constraints which appears to be the most demanding constraint during production scheduling;

- Chromosome: represents a string or a list of mining blocks of which value has to be improved through the application of genetic operators. Each individual block indicates a gene and is identified by the block number or block-ID along the chromosome. The biological feature of a block is carried by the BEV. Blocks of zero and positive BEVs are classified as ore blocks while those of negative BEVs are waste blocks. The order of the blocks in the chromosome does not matter. The number of blocks in the chromosome determines its length and must comply with the required number of blocks to mine per period, thus the mining capacity constraint;
- Initial population: consists of a pool of feasible solutions randomly generated. The size of the initial population must be enough to cover enough area in the search space. The GA runs in such a way the overall initial value is improved generation after generation. The quality of the initial solutions has an impact on the computing time to reach the best solution. The higher the values are the faster is to get closer to the optimal solution;
- Fitness function: represents the objective function that sums up the BEV of blocks in a chromosome. The objective function determines the economic value or the fitness of the chromosome. Regarding the value, the chromosomes are ranked from the most valuable to the least valuable chromosomes. The higher the value, the better is the chromosome and more likely it is to be selected for reproduction. The quality of the chromosome should also be assessed by meeting the processing capacity requirement;
- Selection: the likelihood of a chromosome to be selected for reproduction is done according to the fitness value and the quality of the ore blocks to meet the plant requirement;
- Crossover: consists of exchanging some blocks between two parent-chromosomes (PC1 and PC2) with different values, in order to reproduce two new offspring-chromosome (OC1 and OC2) with the fitness value higher than the two parent-chromosomes. In fact, the weak genes (blocks with negative BEV) in PC1 are replaced by blocks with positive BEV from PC2, and vice versa. Ideally, in PC1 (or PC2), it would be recommended to start removing weak blocks from bottom to top, as they are located in the

block model to avoid violating the block precedence constraints. However, strong genes (blocks with positive BEV) are selected from top to bottom in PC2 (or PC1);

- Mutation: seek to improve the quality of the offspring-chromosome OC1 and OC2 generated from the crossover, through the alteration of some genes in order to satisfy the processing capacity and to improve the mining capacity. The procedure consists of expanding or shortening the chromosome. In the first case, blocks are added to the chromosome while in the second case blocks are removed to adjust both the mining capacity and the processing capacity; and
- Termination condition: the condition for reaching the best solution is defined when the average fitness of the new population is approximately equal to the average fitness of the last population.

Figure 4.7 illustrates the flowchart of the proposed GA to solve the OMPS long-term problem. It should be noted that it ends with the best combination of blocks for one period. Therefore, to proceed for the next period the search space should be free of the blocks found for the previous time period.



PC = Parent-chromosome; OC = Offspring-chromosome

Figure 4.7 Flowchart of the proposed genetic algorithm for solving the OMPS long-term problem

4.9 Chapter summary

This chapter has demonstrated how the long-term production schedule should be solved using the genetic algorithm. The GA technique is a random global search and optimisation model in a space of potential solutions, where it improves its solution through the evolution of the population fitness. There are five stages that the GA follows: initial population, fitness function, selection, crossover and mutation. The primary advantage of the GA comes from the crossover and mutation operations. In the genetic algorithm, solutions are represented in the form of chromosomes. In the production schedule, a chromosome represents a string or a list of mining blocks, and each one of them indicates a gene and is identified by the block number or block-ID. The biological feature of a block is carried by the BEV.

The crossover operator on two chromosomes of blocks consists of exchanging some blocks between two parent-chromosomes PC1 and PC2 using the dual-point technique. In fact, the weak genes (blocks with negative BEV) in PC1 are replaced by blocks with positive BEV from PC2, and vice versa. Ideally, from PC1 (or PC2), it is recommended to start removing weak blocks from bottom to top, as they are in the block model in order to not violate the block precedence constraint. However, strong genes (blocks with positive BEV) are selected from top to bottom in PC2 (or PC1). A swap mutation is used to improve the quality of the offspring-chromosome OC1 and OC2 generated from the crossover, through the alteration of some genes in order to satisfy the processing capacity and to improve the mining capacity. The procedure consists of expanding or shortening the chromosome. In the first case, blocks are added to the chromosome while in the second case blocks are removed to adjust both the mining capacity and the processing capacity. After several iterations, the algorithm terminates when the average fitness of the new population is approximately equal to the average fitness of the last population. The best solution is the best option in the last generation. The following chapter implements the proposed genetic algorithm on two case studies.

5 CASE STUDIES AND RESULTS INTERPRETATION

5.1 Chapter overview

This chapter illustrates the implementation of the proposed genetic algorithm in two case studies. Firstly, it presents the source from which the data was collected, and the resources used to run the algorithm. Secondly, for each case study, the long-term production plan is produced, and a critical analysis is done on the results obtained as well as an examination on complying with the constraints. Finally, in order to validate the proposed genetic algorithm, results obtained are measured against benchmark solutions under results validation section.

5.2 Data source, resources used and algorithm approach

This research study used open-pit model instances freely downloaded from MineLib (http://mansci-web.uai.cl/minelib/), which is a public library website that provides available mining test problems for research experiments. These models are designed for three classical mining problems that include ultimate pit problem (UPL), constrained pit limit problem (CPIT) and precedence constraints production scheduling problem (PCPSP) (Espinoza *et al*., 2012). So far, the library avails 11 mining projects from real-world data and simulated data. Each instance is made up of four data files. The first file contains geological data that describes the mineral deposit, the second file defines the precedence relationship between blocks and the two last files contain benchmark solutions of constrained pit limit and production scheduling, respectively. In this research report, two instances were used for the implementation of the proposed approach.

For this research study, the genetic algorithm was adopted to solve the long-term open-pit mine production scheduling problem. The proposed genetic algorithm was written in PyCharm and interpreted using Python, which is an interpreted, interactive and object-oriented programming language. Python is one of the most used programming languages when it comes to data mining and machine learning (Porcu, 2018). It is also used in automation, artificial intelligence (AI), applications and website. Table 5.1 illustrates the pseudo-algorithm on which the code is based, while the full code is given in the Appendix. The code was run on computer facilities offered by the School of Mining Engineering of the University of the Witwatersrand. Computer specifications on which the running speed depends on, are given in Table 5.2.

Table 5.1 Proposed pseudo algorithm of the OMPS genetic algorithm

```
Input = {S, G, Mmax, Mmin, Pmax, Pmin}
Time period t = 0
while search space S size > Mmax:
        while size of population G is not met:
                block chromosome = []
                while block chromosome < Mmin:
                        select random block b from the search space S
                        block list = block b + block b predecessors
                        block chromosome =+ block list
                return block chromosome
                end while
                add block chromosome to population G
        return G of set of block chromosomes (feasible solutions)
        end while
        while termination condition is not met:
                For each block chromosome g in G
                        calculate the economic value (fitness) of g
                calculate the average fitness of G
                select n best g of G
                for each pair of n best:
                        crossover two block chromosome g1 and g2
                        mutation on offspring g11 and g22 from crossover
                        replace two weak g in new generation G with g11
                         and g22
                end for
        return best g in the last generation G (solution for time period
        t)
        end while
        time period t =+ 1
        search space S = search space - best g in the last generation
end while
return search space as solution for the last period of time period t
end
```

Table 5.2 Computer specifications

Feature	Specification
Operating system	Windows 10 Education
Processor	Intel(R) Xenon(R) CPU X5550 @ 2.67Ghz
RAM	6.00 GB
System type	64-bit OS
Graphics	NVIDIA Quadro FX 580

5.3 Case Study 1: Newman1 instance

5.3.1 The Newman1 model presentation

Newman1 is an academic open-pit mining data set constructed by Alexandra Newman (MineLib, 2012). It is a convenient model for testing open-pit mine planning related problems. Figure 5.1 shows the block model for the mineral deposit. The block model displays the location of each block from the centre-point of the block as the size of the blocks is unknown. Table 5.3 lists the mining parameters provided by MineLib (2012) for the extraction of the Newman1 deposit.

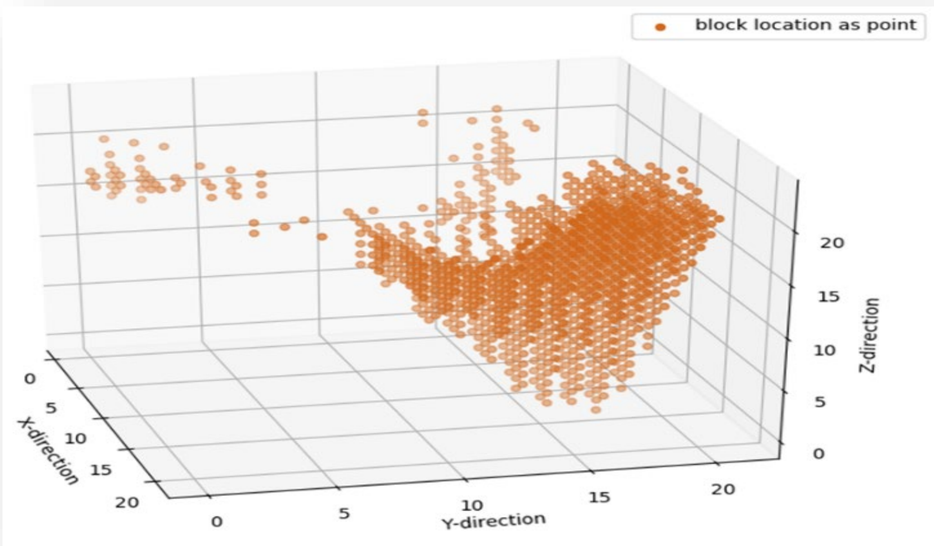


Figure 5.1 Spatial view of block locations in the Newman1 model

Table 5.3 Description of the mining parameters for the Newman1 instance

Parameter	Description	Value
$\boldsymbol{N}$	Number of blocks	1,060
$\boldsymbol{P_i}$	Maximum immediate block predecessors per block	5
$\boldsymbol{d}$	Discount rate	0.08
$\boldsymbol{M_{min}^{tm}}$	Minimum annual mining capacity	1,400,000t
$\boldsymbol{M_{max}^{tm}}$	Maximum annual mining capacity	2,000,000t
$\boldsymbol{P_{min}^{tm}}$	Minimum annual processing capacity	900,000t
$\boldsymbol{P_{max}^{tm}}$	Maximum annual processing capacity	1,100,000t

The geological file contains data for 17 of the 1,060 blocks in the block model of the mineral deposit as illustrated in Table 5.4. It stores information at a block-by-block level, where each row corresponds to a block in the model. The column header, Block-ID, stores a unique identifier of blocks starting from zero. The coordinates of blocks are stored under X, Y, and Z headers. The positive X-axis directs to the left of the viewer, while positive Y-axis points directly towards the viewer and the positive Z-axis points towards the upwards direction. The zero Z-coordinate corresponds to the bottom-most level in the model. The Newman1 model comprises three rock types namely: OXOR, FRWS and FROR that could be read under the Rock Type header. The grade or the amount of mineral of interest in a block is given under the header called Grade, while the tonnage and the economic block value are stored in the Block Tonnage and Block Value columns, respectively. In the absence of a threshold grade, the economic value is used to distinguish ore blocks from waste blocks. Blocks with zero and positive economic values are treated as ore blocks and blocks with negative values are considered to be waste blocks.

Table 5.4 Block description of the Newman1 instance

Block ID	X	Y	Z	Rock Type	Grade	Block Tonnage	Block Value
0	0	1	20	FRWS	0.07	2,192.93	-36,475.56
1	1	1	15	FROR	1.37	5,664.00	24,829.11
2	1	1	16	OXOR	0.91	5,184.00	34,347.21
3	1	1	17	OXOR	0.60	5,184.00	6,743.22
4	1	1	18	OXOR	0.20	5,184.00	-29,020.43
5	1	1	19	OXOR	0.18	1,015.64	-6,077.89
6	1	2	20	OXOR	0.07	2,282.72	-18,188.06
7	2	0	17	OXOR	1.02	5,184.00	44,052.01
8	2	0	18	OXOR	0.80	5,184.00	24,538.24
9	2	1	16	OXOR	0.99	5,184.00	41,760.36
10	2	1	17	OXOR	1.15	5,184.00	55,701.24
11	2	1	18	OXOR	0.76	5,184.00	21,413.26
12	2	2	15	FROR	1.47	5,664.00	34,000.35
13	2	2	16	OXOR	1.64	5,184.00	100,319.01
14	2	2	17	OXOR	0.32	5,184.00	-18,204.53
15	2	2	18	OXOR	0.14	5,184.00	-34,836.37
16	2	2	19	OXOR	0.17	1,049.86	-6,514.75

The block precedence file contains the relationships between blocks in the model. Table 5.5 illustrates a model template of the blocks listed in Table 4.5 as part of the block model file. This file is connected with the geological file through the Block-ID header. The number of block predecessors for a block is given in the second column. The rest of the columns indicate the block predecessors. For example, Block 9 has five block predecessors which are blocks {3, 7, 10, 14, 20}. In the context of the OMPS long-term plan, it means that these blocks must be scheduled in a previous time period than Block 9 or in the same period as Block 9, provided the predecessor blocks are all depleted in that time period.

Table 5.5 Block predecessors of each block in the Newman1 instance

Block ID	No. of Predecessors	Pred1	Pred2	Pred3	Pred4	Pred5
0	0					
1	2	2	9			
2	2	3	10			
3	2	4	11			
4	1	5				
5	2	0	6			
6	0					
7	3	8	11	19		
8	0					
9	5	3	7	10	14	20
10	5	4	8	11	15	21
11	2	5	16			
12	2	9	13			
13	3	10	14	22		
14	3	11	15	23		
15	1	16				
16	2	6	17			

5.3.2 OMPS long-term plan for the Newman1 instance

As discussed in Chapter 4, the genetic algorithm has shown its suitability to tackle the long-term open-pit mine production scheduling. To solve the OMPS problem requires defining the best time at which a block should be extracted in order to simultaneously shorten the payback period on the initial capital invested and maximise the NPV. A production schedule with maximum possible ore blocks in the first years favours these objectives. Applying the genetic algorithm, the best time at which each block should be extracted in an OMPS long-term plan was determined. Figure 5.2 displays in 3D space, the best block combination that should be mined in each time period or year. The long-term plan schedules the extraction of blocks over the life of the mine. Therefore, the proposed schedule delivers a four-year plan of mining activities for the depletion of the Mineral Reserves of the Newman1 project. The sequence of extraction as displayed in Figure 5.2 follows the sequential method of pit expansion, which is more flexible and offers multiple active benches on different levels.

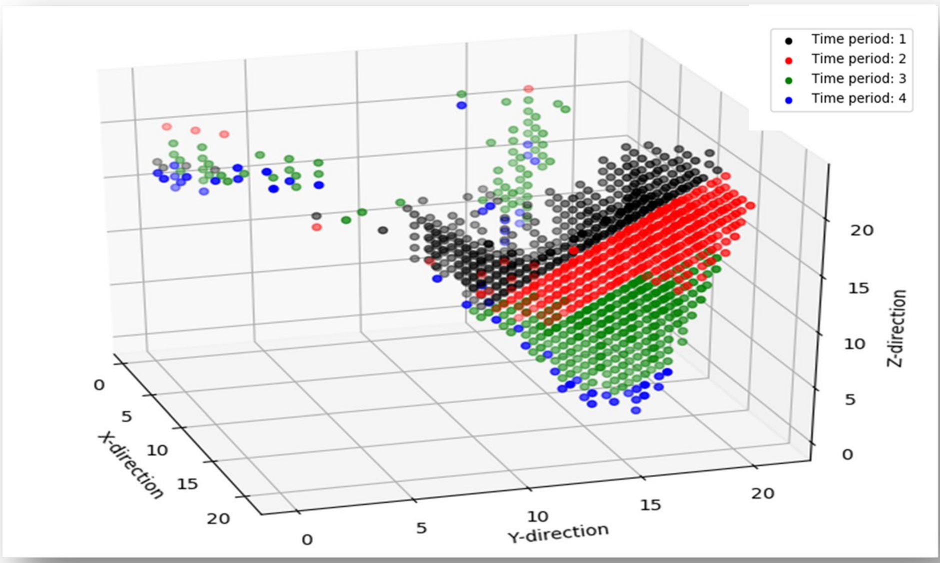


Figure 5.2 Graphical representation of the long-term extraction sequence for the Newman1 project

Table 5.6 summarises the results obtained in each time period. During the first two years, the proposed algorithm schedules an equal number of blocks (350 blocks), while 308 and 52 blocks will be extracted during the two last years, respectively. However, in terms of tonnage, the highest amount of mass material stripped from the mine will be achieved during Year 2 totalling nearly 2 million tonnes. This is because of the presence of blocks with larger tonnages being extracted in Year 2. In terms of ore quality, the weighted average ore grade per time period ranges from 1.27g/t (Year 4) to 2.01g/t (Year 2). Regarding the undiscounted cash flow (UCF) generated each year, the highest value is obtained during the third year of activities, which is justified by the presence of higher number of ore blocks as the pit deepens.

Table 5.6 Newman1 long-term production scheduling results

Period	No. of Blocks	Total Tonnage [t]	Ore Tonnage [t]	Weighted Av. Grade [g/t]	UCF [USD]	Discount Rate	NPV [USD]
0							21,993,571.55
1	350	1,748,384.46	840,065.73	1.63	6,886,035.83	0.08	
2	350	1,931,373.55	571,474.37	2.01	7,344,060.50	0.08	
3	308	1,655,226.77	1,192,708.09	1.61	10,350,919.68	0.08	
4	52	284,392.97	284,392.97	1.27	1,502,483.90	0.08	
Total	1,060						

Figure 5.3 displays the amount of ore and waste scheduled for each time period. It shows that during the first year, the total amount of waste is slightly more than the amount of ore resulting in a stripping ratio of 1.07. During the second year, the stripping ratio is even higher reaching 2.38. After mining at a stripping ratio above one in the first and second years, the two last years will deliver lower stripping ratios of 0.39 and 0, respectively. Blocks scheduled for the fourth year are all ore blocks since the mineral deposit enriches with depth. In summary, the operations in the two first years will mostly consist of stripping the barren material before exposing ore blocks ready to be extracted during the two last periods.

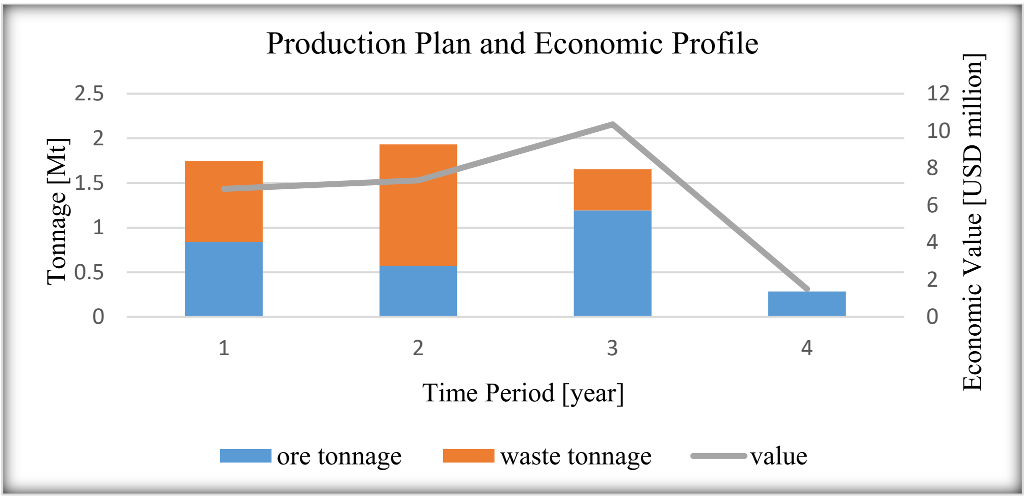


Figure 5.3 Newman1 long-term production plan and economic profile obtained using the genetic algorithm

5.3.3 Compliance with the constraints for the Newman1 instance

An optimum mine production schedule should comply with related constraints in order to guarantee a feasible solution. A critical analysis of the Newman1 OMPS long-term plan generated using the genetic algorithm is examined below against the long-term mine production scheduling constraints:

- Mining capacity: Target-mining capacities are achieved during the first three years range between the minimum and maximum requirements, while the production during the last year will only consist of mining the remaining blocks to the completion of the Mineral Reserves;
- Processing capacity: during the two first years, the material supplied to the plant was slightly below the minimum target, because of the removal of overburden material to open access to ore blocks. However, in the third year, the total tonnage of ore material will exceed the maximum requirement. The extra product would be stockpiled for processing by the plant during the last period;
- Grade control: this parameter was not taken into consideration since the data used does not provide the range of grade product to be mined per period. Ore blocks were separated from waste blocks through the use of the economic block values. However, from Table 5.6, weighted average grade delivered to the plant ranged between 1.27g/t and 2.01g/t;
- Precedence constraint: this is a very important constraint to consider during the production scheduling process. The proposed plan has fully observed the relationships between blocks. Looking at the production plan as displayed in Figure 5.2, blocks that are located near the surface are scheduled prior to those underneath. In other words, blocks scheduled in the first year are block predecessors to blocks in the second year and subsequent years; and
- Reserve constraint: results show that the proposed genetic algorithm does not violate the reserve constraint as the sum of blocks scheduled throughout the life of the mine equal the total block number in the model.

5.3.4 Results validation for the Newman1 instance

In order to validate the results obtained through the application of the genetic algorithm on the Newman1 instance, a numerical result comparison with the current best-known solution was considered. Despite providing available mining data set, MineLib also provides the best solutions for each model instance. Table 5.7 illustrates a concise comparison between the current best solutions and the solution obtained in this research study. The best-known PCPSP solution on the Newman1 delivers an NPV of US$24.4 million. This is an unattainable and infeasible solution obtained with the use of LP relaxation approach. So far, the best feasible solution was obtained using a modified TopoSort heuristic algorithm yielding an NPV of US$23.6 million. However, the GA generated a long-term production schedule that delivers an NPV of US$21.9 million leaving a gap of 7.0% to the modified TopoSort heuristic algorithm. It should be mentioned that the generated NPV is achieved only over four years against a plan of six years for the modified TopoSort algorithm. The running time of the proposed approach was approximately 14.28 seconds, using a computer with specifications provided in Table 5.2.

Table 5.7 Results comparison on Newman1 instance

Method	**NPV [USD]**	**Gap to TopoSort [%]**	**Running Time [seconds]**
TopoSort algorithm	23,658,230.00	0	Not stated
Genetic algorithm	21,993,571.55	7.0	14.28

5.4 Case Study 2: Zuck Small instance

5.4.1 The Zuck Small model presentation

Zuck Small is an academic open-pit mining data set constructed by Mark Zuckerberg and downloaded in MineLib. It is a convenient model for testing open-pit mine planning related problems. Figure 5.4 displays the block model under the pit limit boundaries. This block model shows the location of each block from the centre-point of the block. The model contains 9,400 blocks of unknown size. Table 5.8 describes the mining parameters for the extraction of the Zuck Small deposit.

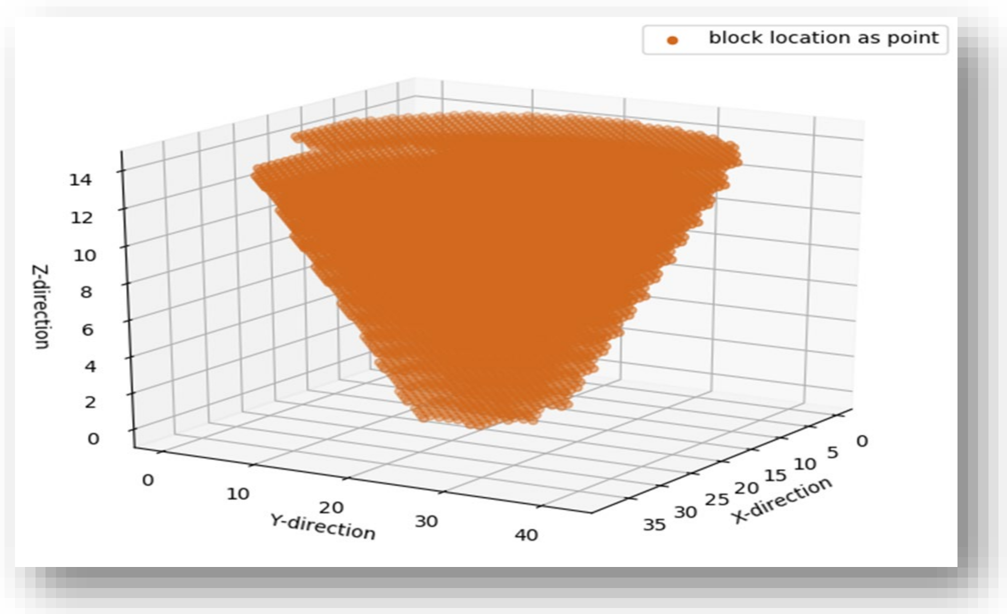


Figure 5.4 Spatial view of block locations in the Zuck Small model

Table 5.8 Description of the mining parameters for the Zuck Small instance

Parameter	Description	Value
$\boldsymbol{N}$	Number of blocks	9,400
$\boldsymbol{P_i}$	Maximum immediate block predecessors per block	32
$\boldsymbol{d}$	Discount rate	0.1
$\boldsymbol{M_{max}^{tm}}$	Maximum annual mining capacity	< 60,000,000t
$\boldsymbol{P_{max}^{tm}}$	Maximum annual processing capacity	< 20,000,000t

The geological file contains data for 17 of the 9,400 blocks in the block model of the mineral deposit as illustrated in Table 5.9. It stores information at a block-by-block level, where each row corresponds to a block in the model. The column header, Block-ID, stores a unique identifier for the block starting from zero. The coordinates of a block are stored under X, Y, and Z headers. The positive X-axis directs to the left of the viewer, while the positive Y-axis points directly towards the viewer and the positive Z-axis points towards the upwards direction. The zero Z-coordinate corresponds to the bottom-most level in the model. The cost of mining a block is recorded under the Cost header, while the economic block values are stored in the Value column. The economic value also distinguishes ore blocks from waste blocks. Blocks with zero and positive economic values are considered as ore blocks and blocks with negative economic values are treated as waste blocks. The tonnage of the block can be read under the Rock Tonnes column followed by the Ore Tonnes column, which stores the amount of ore in a block.

Table 5.9 Block description of the Zuck Small instance

Block ID	X	Y	Z	Cost	Value	Rock Tonnes	Ore Tonnes
0	13	17	10	64,395.01	30,681.26	71,550.01	71,550.01
1	14	17	10	64,503.01	83,755.13	71,670.01	71,670.01
2	15	17	10	65,016.00	158,543.02	72,240.00	72,240.00
3	16	17	10	65,124.00	307,540.09	72,360.00	72,360.00
4	17	17	10	65,124.00	466,480.44	72,360.00	72,360.00
5	18	17	10	65,124.00	538,594.67	72,360.00	72,360.00
6	12	18	10	64,395.01	-6,664.44	71,550.01	0.00
7	13	18	10	64,395.01	64,461.10	71,550.01	71,550.01
8	14	18	10	64,395.01	123,946.92	71,550.01	71,550.01
9	15	18	10	64,503.01	234,989.52	71,670.01	71,670.01
10	16	18	10	65,070.00	334,953.74	72,300.00	72,300.00
11	17	18	10	65,124.00	435,240.68	72,360.00	72,360.00
12	18	18	10	65,124.00	491,320.48	72,360.00	72,360.00
13	19	18	10	65,124.00	505,816.04	72,360.00	72,360.00
14	11	19	10	60,273.00	-60,273.00	66,970.00	0.00
15	12	19	10	64,170.00	-23,061.92	71,300.00	0.00
16	13	19	10	64,395.01	100,194.74	71,550.01	71,550.01

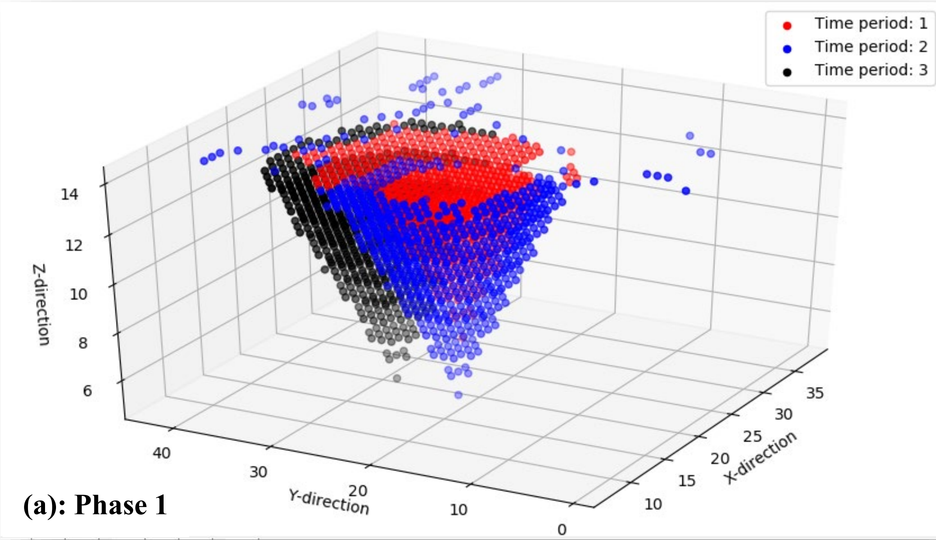
The block precedence file for the Zuck Small instance contains the relationships between blocks in the model. Table 5.10 illustrates a model template of 17 of the 9,400 blocks in the file. This file relates to the geological file through the Block-ID header. The number of direct block predecessors for a block is given in the second column. The rest of the columns identify the block predecessors. For the Zuck Small, not all the block predecessors are located on the level immediately above a block. For example, Block 0 has 13 block predecessors, which are blocks {86, 79, 87, 97, 88, 204, 228, 196, 242, 360, 462, 374, 443}. Block 0 is located on Level 10 according to the Z-coordinate in Table 5.9. However, only blocks {86, 79, 87, 97, 88} are located directly above Block 0 on Level 11. Blocks {204, 228, 196, 242} are located on Level 12 and blocks {360, 462, 374} are located on Level 13.

Table 5.10 Block predecessors of each block in the Zuck Small instance

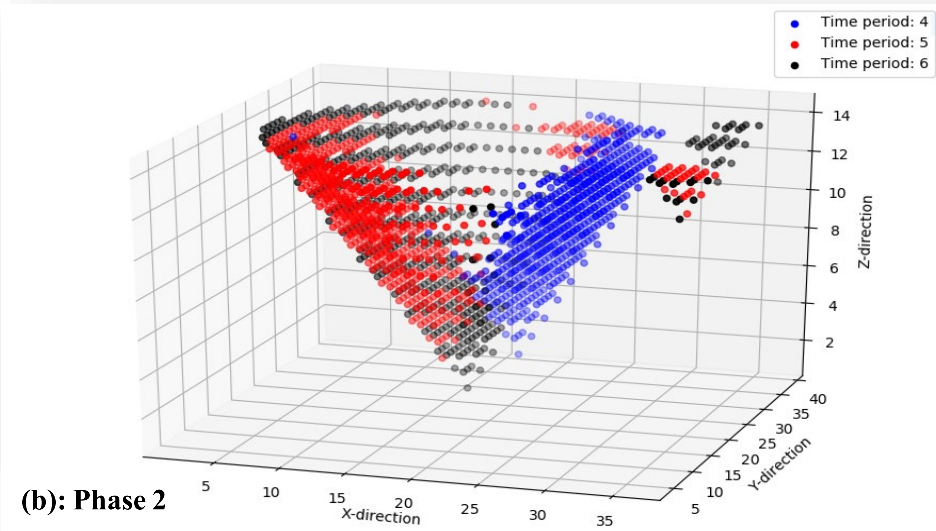
Block ID	No. Pre	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13
0	13	86	79	87	97	88	204	228	196	242	360	462	374	443
1	13	87	80	88	98	89	205	229	197	243	361	463	375	444
2	13	88	81	89	99	90	206	230	198	244	362	464	376	445
3	13	89	82	90	100	91	207	231	199	245	363	465	377	446
4	13	90	83	91	101	92	208	232	200	246	364	466	378	447
5	13	91	84	92	102	93	209	233	201	247	365	467	379	448
6	13	95	86	96	107	97	214	240	204	255	372	481	388	462
7	13	96	87	97	108	98	215	241	205	256	373	482	389	463
8	13	97	88	98	109	99	216	242	206	257	374	483	390	464
9	13	98	89	99	110	100	217	243	207	258	375	484	391	465
10	13	99	90	100	111	101	218	244	208	259	376	485	392	466
11	13	100	91	101	112	102	219	245	209	260	377	486	393	467
12	13	101	92	102	113	103	220	246	210	261	378	487	394	468
13	13	102	93	103	114	104	221	247	211	262	379	488	395	469
14	13	105	95	106	118	107	226	253	214	268	386	501	404	481
15	13	106	96	107	119	108	227	254	215	269	387	502	405	482
16	13	107	97	108	120	109	228	255	216	270	388	503	406	483

5.4.2 *OMPS long-term plan for the Zuck Small instance*

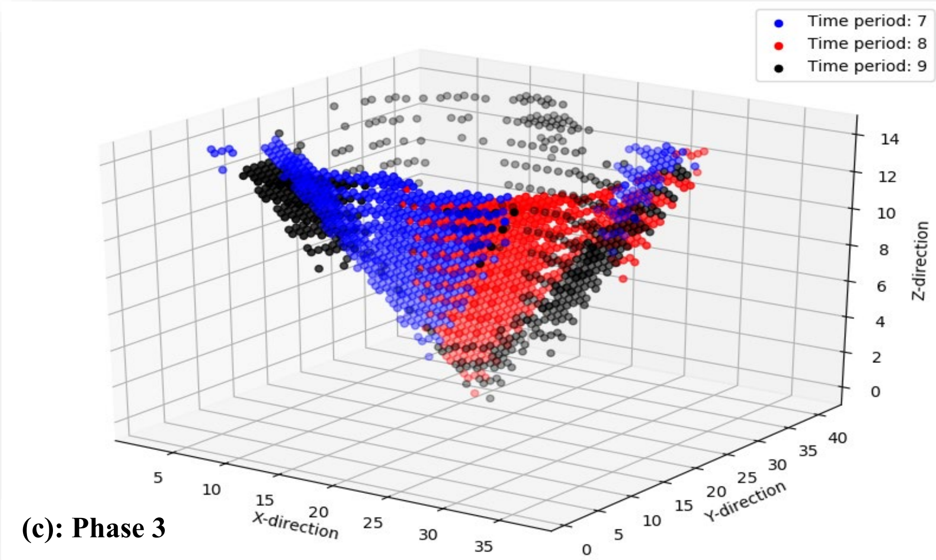
The application of the proposed genetic algorithm on the Zuck Small instance produced a long-term production plan for 17 years. During each year, the best combination of blocks was found to yield the maximum NPV with the use of the genetic algorithm. Table 5.11 shows numerical results obtained using the proposed approach applied on Zuck Small model. The NPV obtained from the generated production schedule was estimated at USD800 million. Due to the size of the block model and for clear visibility, the production plan was separated into six phases each comprising three time periods as shown in Figure 5.5(a) to (f).



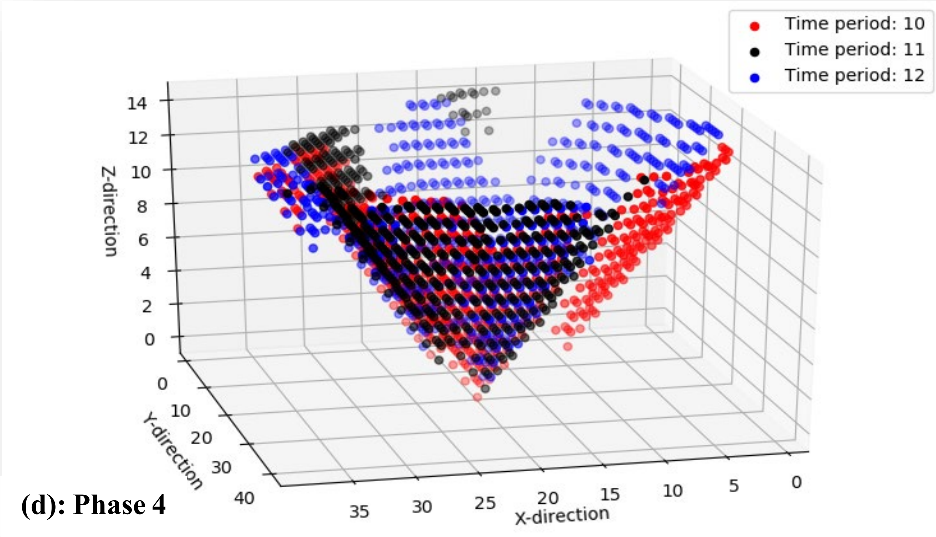
(a): Phase 1



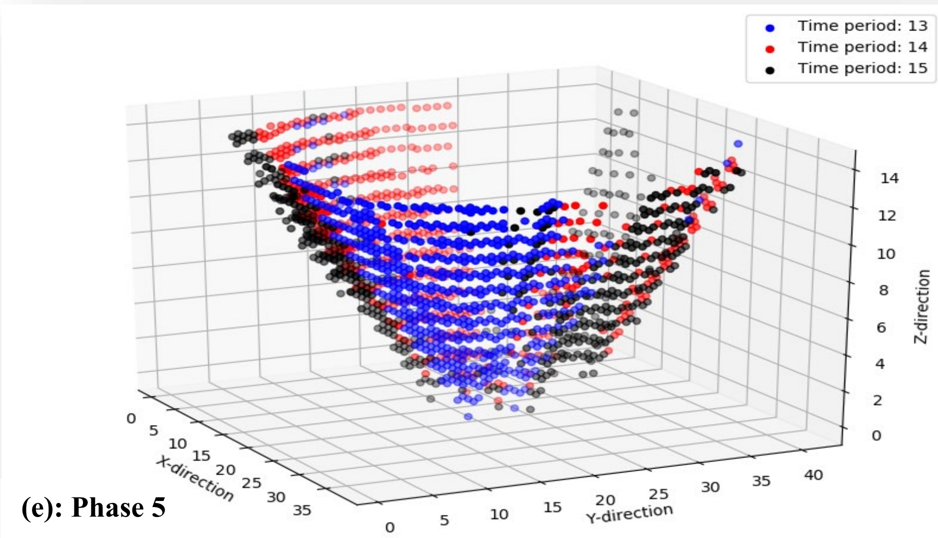
(b): Phase 2



(c): Phase 3



(d): Phase 4



(e): Phase 5

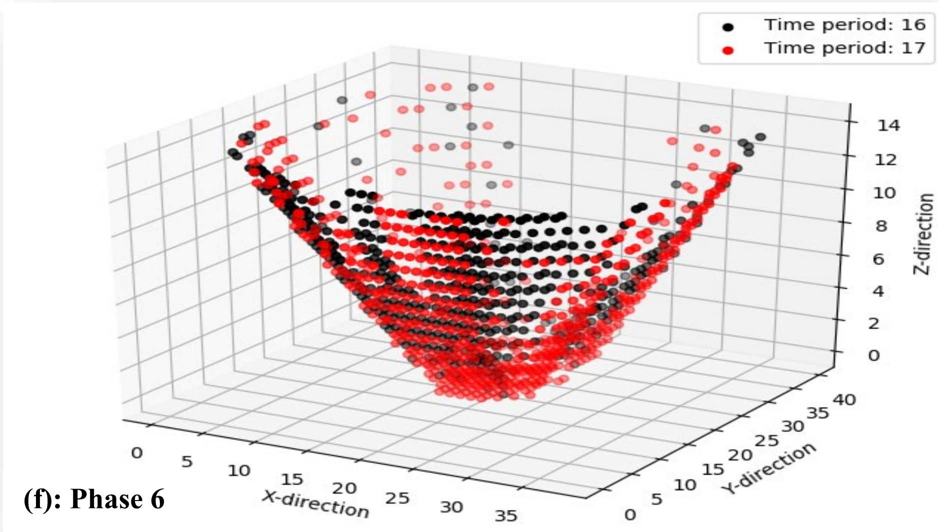


Figure 5.5 Graphical representation of the long-term extraction sequence for the Zuck Small project, displayed in six phases (a), (b), (c), (d), (e) and (f)

Table 5.11 Zuck Small long-term production scheduling results

Time Period	No. of Blocks	Total Tonnage[T]	Ore Tonnage[T]	Cash Flow [USD]	Discount Rate	NPV [USD]
0						800,387,066.84
1	600	29,322,860.97	27,236,208.27	116,881,329.30	0.1	
2	600	33,957,194.91	24,960,970.78	113,708,217.40	0.1	
3	529	33,832,301.32	25,766,382.12	154,564,609.90	0.1	
4	556	36,525,220.68	28,252,063.04	193,025,223.60	0.1	
5	567	33,859,080.01	13,419,760.74	83,059,613.12	0.1	
6	572	35,014,744.09	17,844,010.09	114,877,550.70	0.1	
7	547	33,016,293.80	16,105,900.96	73,669,049.47	0.1	
8	532	34,434,580.42	17,693,550.79	103,145,808.30	0.1	
9	564	33,254,540.27	15,491,691.03	89,963,236.76	0.1	
10	556	36,292,576.12	20,977,530.70	95,308,452.70	0.1	
11	518	31,789,797.61	10,356,330.44	30,155,100.94	0.1	
12	570	35,209,221.82	14,789,240.73	55,849,655.43	0.1	
13	538	33,633,705.57	12,659,090.53	38,956,309.49	0.1	
14	525	32,345,426.42	10,351,160.35	26,966,211.07	0.1	
15	521	32,927,088.46	13,064,151.32	35,231,722.61	0.1	
16	512	31,674,313.37	9,731,780.35	25,589,438.62	0.1	
17	593	39,040,763.38	21,531,011.35	71,769,557.29	0.1	
Total	9,400					

Figure 5.6 presents the annual amount of ore and waste scheduled for the extraction of the Zuck Small mineral deposit alongside with the total revenue generated each year. It shows that during the first year of block extraction, the amount of ore material is about 92% of the total tonnage extracted in that year, resulting in a stripping ratio of 0.07. Overall, the stripping ratio follows a descending trend with small variations over the life of the mine. Over the first four years, the ore tonnage has exceeded the maximum required capacity to the plant. The extra ore material will be sent to a stockpile for processing by the plant in later periods (years). From Year 5, ore tonnage has been kept within the required range except for Years 10 and 17 where the amount of ore material sent to the plant is slightly above the maximum required. Therefore, the plan has demonstrated its efficiency to be able to supply enough ore to the plant even during periods of poor production. In terms of the mining capacity, the total tonnage across the operations will average between 25Mt and 40Mt. The UCF generated also follows a decreasing overall trend over the years reaching a peak in Year 5 of operation.

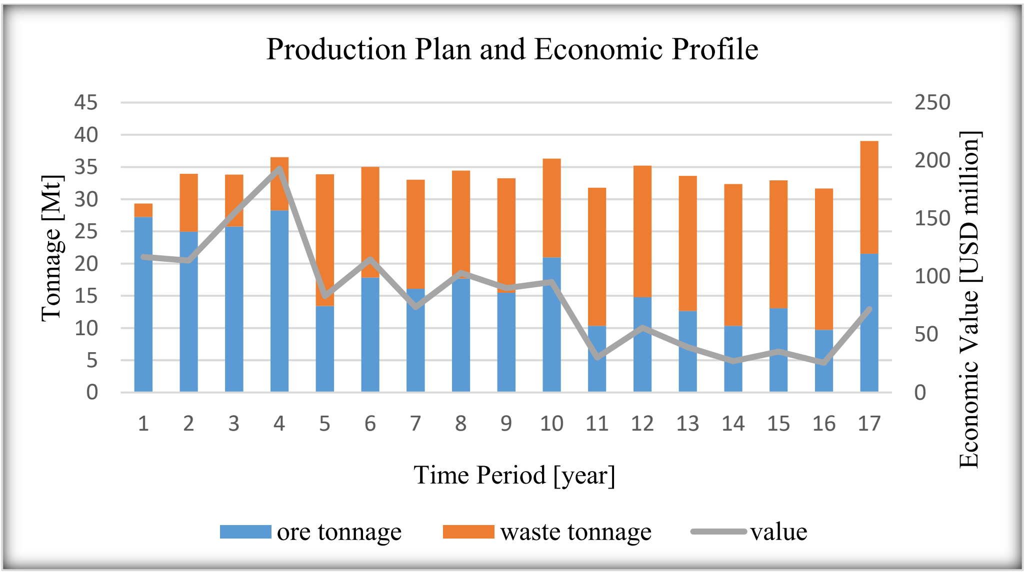


Figure 5.6 Zuck Small long-term production plan and economic profile obtained using the genetic algorithm

5.4.3 *Compliance with the constraints for the Zuck Small instance*

An optimum mine production schedule should comply with related constraints in order to guarantee a feasible solution. A critical analysis of the Zuck Small OMPS long-term plan generated using the genetic algorithm is examined below against the long-term mine production scheduling constraints:

- Mining capacity: the Zuck Small instance defined the maximum mining capacity, which should be less than 60Mt per year. However, no minimum restriction was given. The production plan generated has met the mining capacity requirement producing between 25Mt and 40Mt each year throughout the life of mine. The lowest tonnage (ore and waste material) occurs during the first year and the highest tonnage during the last year;
- Processing capacity: likewise, the mining capacity, the minimum processing capacity is not given, while the maximum was fixed at 20Mt of annual ore material to be sent to the plant. Practically, if the maximum processing capacity is violated, it causes no prejudice because all the extra ore material will be stockpiled and supplied to the plant during periods of insufficient ore production;
- Grade control: this parameter was not taken into consideration since the Zuck Small data provided by MineLib does not reflect any information in regards. Ore blocks were separated from waste blocks using the block economic values;
- Precedence constraint: the proposed genetic algorithm applied on the Zuck Small model has complied with this constraint for the production scheduling activity. The extraction plan has fully observed the relationships between blocks. Blocks that are located closer to the surface are scheduled prior to those underneath. In other words, blocks scheduled in the first year are block predecessors to blocks in the second year and subsequent years. There is no block, which is scheduled before its block predecessors. The GA approach was able to deal with the precedence constraint even for big block models of the size of the Zuck Small instance; and

- Reserve constraint: results show that the genetic algorithm applied on the Zuck Small instance does not violate the reserve constraint as the sum of blocks scheduled throughout the life of the mine equal the total number of blocks (9,400 blocks) in the model.

5.4.4 Results validation for the Zuck Small instance

The LP relaxation method to solve the PCPSP on the Zuck Small model generated an infeasible solution with an NPV of US$905.8 million. However, the best feasible solution up to now has been obtained using a modified TopoSort heuristic algorithm achieving an NPV of USD872.3 million. The proposed GA model in this research has delivered an NPV of USD800.4 million when applied on the Zuck Small model. This result has left a discrepancy of about 8.3% to the modified TopoSort approach. Table 5.12 presents the results obtained with these best-known solutions provided by MineLib. Nevertheless, it should be noted that for the GA solution the NPV would be delivered after 17 years of operation against 20 years for the modified TopoSort heuristic algorithm. Additionally, the results obtained could be improved with the same proposed GA model because of the randomness process. The quality of the results depends on the quality of the initial generation in the GA process. The running time of the proposed approach was approximately 17 minutes, using a computer with specifications provided in Table 5.2.

Table 5.12 Results comparison on Zuck Small instance

Method	NPV [USD]	Gap to TopoSort [%]	Running Time [minutes]
TopoSort algorithm	872,372,967.00	0	Not stated
Genetic algorithm	800,387,066.84	8.3	17

5.5 Chapter summary

In this chapter, two case studies of block model instances freely downloaded from the MineLib library were used to test the proposed GA model. The first case study called Newman1 with 1,060 blocks was scheduled for extraction over four years. The generated production schedule reveals that an equal number of 350 blocks should be extracted in the first and second years, while 308 blocks and 52 blocks should be scheduled for the two last years, respectively. However, the stripping ratio at which the mine should operate was evaluated at 1.07, 2.38, 0.39 and 0 during the corresponding years. Hence, the mining activities in the two first years will mostly consist of stripping the barren material before exposing ore blocks ready to be extracted during the two last periods.

It was demonstrated that the production plan has met the required mining production capacity every year except the last year where the remaining blocks were not enough to produce the capacity required. In regard to the processing capacity, the production schedule delivered ore material to the plant slightly below the minimum requirement in the first two years. During the third year, the ore amount to be sent to the plant exceeds the maximum required capacity. The extra ore material would be stockpiled for a latter supply during the last year of operations. This is because the mineral grade increases, as the orebody deepens. The precedence relationship between blocks defined by the slope angle was observed as block predecessors to any block in the model are scheduled in a prior year or in the same year. The last constraint is the ore reserve; it was shown that the generated plan does not violate the reserve constraint since the sum of blocks scheduled throughout the life of the mine equalled the total block number in the model.

The numerical results obtained were measured against the benchmark results provided in MineLib. The generated OMPS long-term plan delivered an NPV of USD21.9 million leaving a gap of 7.0% to the current best-known result obtained using the modified TopoSort heuristic approach. However, it should be mentioned that the NPV realised was achieved only over four years against a plan of six years for the modified TopoSort approach. The running time of the proposed approach was approximately 14.28 seconds.

The second case study called Zuck Small comprises 9,400 blocks in the ultimate pit limit, which is approximately 10 times larger than the Newman1 model. The generated OMPS long-term plan covered a period of 17 years over which the number of mining blocks were determined on a yearly basis. It was shown that during the first year of mining activities, the amount of ore material produced is about 92% of the total tonnage produced in that year, resulting in a stripping ratio of 0.07. Overall, the stripping ratio decreases from the starting year of activities to the closing year with small variations.

Regard compliance to the constraints, the proposed OMPS long-term plan met the mining capacity requirement producing between 25Mt and 40Mt each year throughout the life of the mine. Considering the processing capacity requirements, over the first four years, the ore tonnage exceeded the maximum required capacity to the plant that aligns with an efficient plan goal to mine as many as possible ore blocks in the first years. The extra ore material would be sent to the stockpile for later processing in the plant. From Year 5, the ore tonnage has been kept within the required range except for Years 10 and 17 where the amount of ore material to be sent to the plant were slightly above the maximum required. Therefore, the plan demonstrated its efficiency to be able to supply enough ore product to the plant even during periods of poor production.

Regarding the block precedence relationship, blocks scheduled in the first year are block predecessors to blocks in the second and subsequent years. There was no block, which was scheduled before its block predecessors. The proposed GA model demonstrated ability to handle block precedence constraints even for large block models of the size of the Zuck Small instance. Finally, the reserve constraint was not violated as the sum of blocks scheduled throughout the life of the mine equal the total number of blocks (9,400 blocks) in the model.

Looking at the numerical results, the proposed GA model applied to the Zuck Small instance delivered an NPV of US$800.4 million after 17 minutes of running the algorithm. This result left a gap of 8.3% to the current best-kwon solution obtained using the modified TopoSort heuristic approach. This NPV was delivered after 17 years of mining operations against 20 years for the modified TopoSort approach. Additionally, the results obtained could be improved with the same proposed GA model because of the inherent randomness of the process. For example, if the best ever blocks are randomly selected for the initial generation, the proposed GA would probably deliver better results. The next chapter concludes the findings of the research and provides recommendations for future work.

6 CONCLUSIONS AND RECOMMENDATIONS

6.1 Chapter overview

This chapter firstly presents the problem being investigated and the method approach used. Secondly, it provides findings of the research by answering the research questions surrounding the problem and presenting the main results from the case studies. Lastly, recommendations for future research work raised from the findings are suggested.

6.2 Introduction

The economic sustainability of a mining project is highly dependent upon a careful mine planning and management of the Mineral Resource. The primary goal of mine planning is to optimise all the integrated activities in order to maximise the NPV of the project. The optimisation of mine planning also includes the OMPS long-term problem. The OMPS long-term problem consists of defining the best time at which each block should be extracted from the pit over the life of the mine. Finding the optimum production schedule is not a straightforward answer, because it involves large datasets and multiple hard and soft constraints which make it a large combinatorial optimisation problem. That is why mine production scheduling requires the use of operations research techniques that are suitable for handling complex and difficult problems.

The aim of this research report was to find the best OMPS long-term plan that maximises the NPV of the mine under geological and physical constraints. Considering the nature of the problem, the research report made use of the GA to answer the following question: "*When and how should each block be extracted from the pit in such a way that the NPV of the mine is maximised over the life of the mine*?" Therefore, a GA model was developed and coded in Python programming language in order to be run on a block model. The application of the proposed GA in this research report determined the best extraction sequence out of thousands of feasible solutions. However, GA being a heuristic approach did not provide the optimum solution to the problem but helped to reach a near-optimal solution to the

OMPS problem investigated, which is difficult to be solved holistically using deterministic techniques.

6.3 Research findings

The implementation of the GA to solve the long-term mine production scheduling in an open-pit mine has shown encouraging results when applied to two mining case studies. It was demonstrated that the GA fits well in solving the OMPS problem as it integrates the crossover and the mutation operators to improve the solution over many iterations. The solution procedure involves a search space through which the best solution is found out of multiple feasible solutions. The advantages discovered for using the GA could be summarised as following:

- The GA is simple to understand, develop and code in Python programming language;
- The GA is quite flexible to incur some modifications in order to handle the OMPS long-term problem by simultaneously complying with constraints such as mining capacity, processing capacity, block precedence relationships and reserve constraint;
- The GA generates near-optimal solutions for small and large data sets in a relatively short time; and
- With the use of genetic operators such as selection, crossover and mutation, the GA can improve solutions through iterations. These genetic operators are suitable for exploiting and exploring the vast search space of feasible solutions.

In order to test and validate the proposed GA model, this study used two instances of block models freely downloaded from MineLib. The first model called Newman1 has 1,060 blocks that were scheduled through the life of the mine to produce the maximum NPV. When applied to this model, the GA provided a feasible OMPS long-term plan of four years for the depletion of the Mineral Reserve. The running time to deliver this production plan was approximately 14.28 seconds. The OMPS long-term plan revealed that an equal number of 350 blocks should be mined in the first and second years, 325 blocks in the third year and 52 blocks in the last year of operations. The production schedule generated an NPV of

USD21.9 million leaving a discrepancy of 7.0% to the optimal solution obtained using the TopoSort heuristic approach. The NPV generated was obtained after four years of operations against a plan of six years generated by the TopoSort algorithm.

The second model instance called Zuck Small had 9,400 blocks in the ultimate pit limit. However, the implementation of the proposed GA model on the Zuck Small generated an OMPS long-term plan of 17 years. This plan was obtained after 17 minutes of running the GA model in Python. In fact, the generated production schedule yielded an NPV of USD800.4 million leaving a discrepancy of 8.3% to the current feasible and optimal solution delivered over 20 years of life of mine using the TopoSort heuristic algorithm.

6.4 Recommendations for future work

Although a successful application of the GA for the production scheduling on an open-pit mine, further research needs to be done to improve the proposed GA in order to close the gap with the optimal feasible solution. The presence of the gaps might be due to the randomness procedure in the GA having a vital effect on the quality of the initial feasible solutions. However, the room for improvement should be in redefining the method for the creation of the initial solutions. Adding to this, improving crossover and mutation operators would be a major step to be able to find the best blocks to crossover and mutate in terms of block position and block quality (ore block or waste block). Future work should be aligned with the following:

- Applying the proposed GA on a real-world active open-pit mine in order to be able to integrate the grade control constraint;
- Developing and integrating stockpile and waste dump modules into the proposed GA model;
- Developing a new GA model for the medium and short-term plan arising from the long-term plan to find the block-by-block extraction sequencing that will incorporate the mine operating cost and technical constraints;
- Extending the application of the GA to underground mines; and
- Incorporating the GA as a built-in component into the available mining software.

7 REFERENCES

Ahmadi, M. R. and Shahabi, R. S., 2018. Cutoff grade optimization in open pit mines using gentic algorithm. *Resources Policy,* 55, pp. 184-191.

Akaike, A. and Dagdelen, K., 1999. A strategic production scheduling method for an open pit mine. Colorado, *Proceedings of the 28th Application of Computers and Operations Research in the Minerals Industries,* pp. 729-738.

Akbari, A., Osanloo, M. and Shirazi, M., 2008. Ultimate pit limit (UPL) determination through minimizing risk costs associated with price uncertainty, [Online]. Available at: https://min-pan.krakow.pl/wp-content/uploads/sites/4/2017/12/akbari-osanloo-shirazi.pdf [Accessed 02 May 2019].

Alipour, A., Khodaiari, A. A., Jafari, A. and Tavakkoli-Moghaddam, R. T., 2017. A genetic algorithm approach for open-pit mine production scheduling. *International Journal of Mining and Geo-Engineering,* 51(1), pp. 47-52.

Behrang, K., Hooman, A. N. and Clayton, D., 2014. A linear programming model for long-term mine planning in the presence of grade uncertainty and a stockpile. *International Journal of Mining Science and Technology,* 24(4), pp. 451-459.

Birch, C., 2019. Optimisation of mining block size for narrow tabular gold deposits. *In: Widzyk-Capehart E., Hekmat A., Singhal R. (eds) Proceedings of the 27th International Symposium on Mine Planning and Equipment Selection - MPES2018.* Switzerland: Springer, Cham, pp. 121-141.

Blom, M., Pearce, A. R. and Stuckey, P. J., 2018. Short-term planning for open pit mines: A review. *International Journal of Mining, Reclamation and Environment,* 33(5), pp. 318-339.

Caccetta, L. and Hill, S. P., 2003. An application of branch and cut to open pit mine scheduling. *Journal of Global Optimization ,* 27, pp. 349-365.

Chicoisne, R., Espinoza, D., Goycoolea, M., Moreno, E. and Rubio, E., 2012. A new algorithm for the open-pit mine production scheduling problem. *Operations Research,* 60(3), pp. 517-528.

Consuegra, F. A. and Dimitrakopoulos, R., 2009. Stochastic mine design optimisation based on simulated annealing: pit limits, production schedules, multiple orebody scenarios and sentivity analysis. *Mining Technology,* 118(2), pp. 79-90.

Cuba, M. A. and Deutsch, C. V., 2011. An approximate pit optimization methodology suited to the SLM framework. *CCG Annual Report 13*, pp. 305-1 - 305-12. Available at: http://www.ccgalberta.com/ccgresources/report13/2011-305_approximate_mine_sequencing.pdf [Accessed 14 March 2019].

Dagdelen, K. and Johnson, T., 1986. Optimum open pit mine production scheduling by Lagrangian parameterization. New York. In: *19th APCOM Symposium of the Society of Mining Engineers (AIME)*, pp. 127-141.

Darling, P., 2011. *SME Mining Engineering Handbook.* 3rd ed. USA: Society for Mining, Metallurgy and Exploration, Inc.

Datamine, 2015. Studio 5D planner update*,* [Online]. Available at: https://www.dataminesoftware.com/studio-5d-planner-update/ [Accessed 08 July 2019].

Denby, B. and Schofield, D., 1994. Open pit design and scheduling by use of genetic algorithms. *Transactions of the Institution of Mining and Metallurgy,* 103, pp. A21-A26.

Dimitrakopoulos, R. and Jewbali, A., 2013. Joint stochastic optimisation of short and long term mine production planning: method and application in a large operating gold mine. *Mining Technology,* 122(2), pp. 110-123.

Djilani, M.C., 1997. Interactive open-pit design using parameterization techniques*, PhD Thesis,* Leeds: University of Leeds.

Elahi zeyni, E., Kakaie, R. and Yousefi, A., 2011. A new algorithm for optimum open pit design: Floating cone method III. *Journal of Mining & Environment,* 2(2), pp. 118-125.

Erarslan, K. and Celebi, N., 2001. A simulative model for optimum open pit design. *Canadian Institute of Miming (CIM) Bulletin,* 94(1055), pp. 59-68.

Espinoza, D., Goycoolea, M., Moreno, E. and Newman, A., 2012. MineLib: A library of open pit mining problems. *Annals of Operations Research,* 206, pp. 93-114.

Ferland, J. A., Amaya, J. and Djuimo, M. S., 2007. Application of a particle swarm algorithm to the capacitated open pit mining problem. In: *Mukhopadhyay S.C., Gupta G.S., eds. Autonomous Robots and Agents, Studies in Computational Intelligence,* 76, pp. 127-133.

Gaupp, M. P., 2008. Methods for improving the tractability of the block sequencing problem for open pit mining, *PhD Thesis,* Colorado School of Mines.

GEMCOM, 2013. GEMCOM MineSched, Surface and Underground Scheduling. Manual MineShed 8.0, *ed. Santiago*.

Genc, B., 2015. A methodology for evaluating utilisation of mine planning software and consequent decision-making strategies in South Africa, *PhD Thesis,* Johannesburg: University of the Witwatersrand.

Gershon, M., 1983. Optimal mine production scheduling: evaluation of large scale mathematical programming approaches. *International Journal of Mining Engineering,* 1, pp. 315-329.

Githiria, J. and Musingwini, C., 2018. Comaparison of cut-off grade models in mine planning for improved value creation based on NPV. In: *Proceeding of the 6th Regional Conference of the Society of Mining Professors, The SAIMM, Johannesburg,* pp. 347-362.

Gleixner, A. M., 2008. Solving large-scale open-pit mining production scheduling problems by integer programming*, MSc Thesis,* Germany: Technical University of Berlin.

Godoy, M. and Dimitrakopoulos, R., 2004. Managing risk and waste mining in long-term production scheduling of open-pit mines. *Society for Mining, Metallurgy, and Exploration,* 316, pp. 43-50.

Gürtunca, R., 2003. South African mining industry in the new millenium and new technologies for the future. *10th International Mining Congress and Exhibition of Turkey-IMCET*, pp. 57-64.

Holland, J. H., 1975. *Adaptation In Natural And Artificial Systems.* London: The MIT Press.

Hustrulid, W., Kuchta, M. and Martin, R., 2013. *Open pit mine planning and design.* 3rd ed. London: CRC Press.

Johnson, T. B., 1968. *Optimum open pit mine production scheduling.* 1st ed. University of California, Berkeley: Defense Technical Information Center.

Johnson, T. and Sharp, W., 1971. A three-dimensional dynamic programming method for optimal ultimate open pit mine design. U.S. Bureau of Mines Report of Investigations 7553.

Katakwa, T., Musingwini, C. and Genc, B., 2013. Online database of mine planning and peripheral sofware used in the South African mining industry. *Journal of the Southern African Institute of Mining and Metallurgy,* 113(6), pp. 497-504.

Kennedy, J. and Eberhart, R., 1995. Particle swarm optimization. *Proceedings of ICNN'95 - International Conference on Neural Networks,* Perth, Australia, 4, pp. 1942-1948.

Khalokakaie, R., Dowd, P. and Fowell, R., 2000. Lerchs-Grossmann algorithm with variable slope angles. *Mining Technology,* 109(2), pp. 77-85.

Khan, A., 2018. Long-term production scheduling of open pit mines using particle swarm and bat algorithms under grade uncertainty. *Journal of the Southern African Institute of Mining and Metallurgy,* 118(4), pp. 361-368.

Khan, A. and Niemann-Delius, C., 2014. Production scheduling of open pit mines using particle swarm optimization algorithm. *Advances in Operations Research*, 2014, pp. 1-9.

Khan, A. and Niemann-Delius, C., 2015. Application of particle swarm optimisation to the open pit mine scheduling problem. *Proceedings of the 12th International Symposium Continuous Surface Mining* - Aachen 2014*,* pp. 195-212.

Kloppers, B., Horn, C. & Visser, J., 2015. Strategic and tactical requirements of a mining long-term plan. *Journal of the Southern African Institute of Mining and Metallurgy,* 115(6), pp. 515-521.

Kumral, M., 2004. Genetic algorithms for optimization of a mine system under uncertainty. *Production Planning & Control,* 15(1), pp. 34-41.

Kumral, M. and Dowd, P., 2005. A simulated annealling approach to mine production scheduling. *Journal of the Operational Research Society,* 56(8), pp. 922-930.

Kwiri, J. and Genc, B., 2017. Mine planning and optimization techniques used in surface mining - Position paper. In: *Proceedings of the International Mining Congress of Turkey (IMCET),* pp. 210-221.

Lane, K. F., 1964. Choosing the optimal cut-off grade. *Colorado School of Mines Quarterly,* 59(4), pp. 811-829.

Lerchs, H. and Grossmann, I. F., 1965. Optimum design of open-pit mines. *Transactions, C.I.M Bulletin,* 58, pp. 47-54.

Liu, W. and Zhang, H., 2016. PSO algorithm for block sequencing problem in open pit mining. *Proceedings of the 2015 5th International Conference on Computer Sciences and Automation Engineering (ICCSAE 2015),* pp. 434-441.

Lyeme, H. and Selemani, M. A., 2012. *Introduction to Operations Research: Theory and Applications.* Deutschland, Germany: LAP LAMBERT Academic Publishing.

Marcotte, D. and Caron, J., 2012. Ultimate open pit stochastic optimization. *Computers & Geosciences,* 51(2013), pp. 238-246.

Matamoros, M. E. V., 2014. Stochastic short-term production scheduling accounting for fleet allocation, operational considerations, blending restrictions and future multi-element ore control data. *MEng. Thesis*, Montreal, Quebec: McGill University.

Meagher, C., Dimitrakopoulos, R. and Avis, D., 2014. Optimized open pit mine design, pushbacks and the gap problem - A review. *Journal of Mining Science,* 50(3), pp. 508-526.

MineLib, 2012. Datasets: Newman1, [Online]. Available at: http://mansci-web.uai.cl/minelib/newman1.xhtml [Accessed 07 August 2019].

MineRP, 2012. Mine planning and scheduling, [Online]. Available at: https://www.minerp.com/mine-planning---scheduling [Accessed 08 July 2019].

Mitchell, M., 1999. *An introduction to genetic algorithms.* 1st ed. London: First MIT Press.

Moosavi, E. and Gholamnejad, J., 2015. Long-term production scheduling modeling for the open pit mines considering tonnage uncertainty via indicator Kriging. *Journal of Mining Science,* 51(6), pp. 1226-1234.

Mugwagwa, D., 2017. Cut-off grade optimisation for a bimetallic deposit: A case study of the Ruashi Mine copper-cobalt deposit, *MSc Thesis,* Johannesburg: University of the Witwatersrand.

Musingwini, C., 2016. Presidential Address: Optimization in underground mine planning - developments and opportunities. *Journal of the Southern African Institute of Mining and Metallurgy,* 116(9), pp. 809-820.

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

Nogholi, A. A. M., 2015. Optimisation of open pit mine block sequencing. *PhD Thesis*, Queensland: Queensland University of Technology.

Ortiz, J. and Emery, X., 2006. Geostatistical estimation of mineral resources with soft geological boundaries: a comparative study. *Journal of the Southern African Institute of Mining and Metallurgy*, 106(8), pp. 577-584.

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

Pana, M.T. 1965. The simulation approach to open-pit design. In: *5th APCOM,* Tucson Arizona, March 16–19, pp. ZZ-1 to ZZ-24.

Porcu, V., 2018. *Python for data mining quick syntax reference.* Nuoro: Apress.

Ramazan, S., Dagdelen, K. and Johnson, T., 2005. Fundamental tree algorithm in optimising production scheduling for open pit mine design. *Mining Technoogy,* 114(1), pp. 45-54.

Rendu, J.-M., 2014. *An introduction to cut-off grade estimation.* 2nd ed. Colorado: Society for Mining, Metallurgy and Exploration.

Rendu, J.-M. and Miskelly, N., 2001. Mineral resources and mineral reserves: progress on the international definitions and reporting standards. *Mining Techonology,* 110(3), pp. 133-138.

Roman, R.J., 1974. The role of time value of money in determining an open pit mining sequence and pit limits. *12th Symposium on the Application of Computer and Operations Research in the Mineral Industry (APCOM)*, pp. 72-85.

Rosas, J. S., Jélvez, E., Amaya, J. and Morales, N., 2016. Optimizing open-pit block scheduling with exposed ore reserve. *Journal of the Southern African Institute of Mining and Metallurgy,* 116(7), pp. 655-662.

Rovenscroft, P.J., 1993. Risk analysis for mine scheduling by conditional simulation. *International Journal of Rock Mechanics and Mining Sciences & Geomechanics Abstracts,* 30(2), pp. A104-A108.

RPMGlobal, 2018. Softwares/xpac, [Online]. Available at: https://www.rpmglobal.com/softwares/dragsim/ [Accessed 08 July 2019].

Ruiseco, J. R., 2016. Dig-limit optimisation in open pit mines through genetic algorithms. *MSc. Thesis*, Montreal: McGill University.

Samavati, M., 2017. New long-term production scheduling methodologies for open-pit mines. *PhD Thesis,* Australia: The University of New South Wales, Canberra.

Samavati, M., Essam, D., Nehring, M. and Sarker, R., 2018. A new methodology for the open-pit mine production scheduling. *Omega,* 81, pp. 169-182.

SAMREC, 2016. *The South African Code for the Reporting of Exploration Results, Mineral Resources and Mineral Reserves,* South Africa.

Sattarvand, J., 2009. Long-term open-pit planning by ant colony optimization. *PhD Thesis,* Germany: RWTH Aachen University.

Shishvan, M. S. and Sattarvand, J., 2015. Long term production planning of open pit mines by ant colony optimization. *European Journal of Operational Research,* 240(3), pp. 825-836.

Shonts, E. T. and Nettleton, J. M., 2011. Chapter 10.9: Strip Mine Planning and Design. In: *SME Mining Engineering Handbook. Cornwall: Society for Mining, Metalurgy, and Exploration,* pp. 1013-1026.

Tachefine, B. and Soumis, F., 1997. Maximal closure on a graph with resource constraints. *Computers and Operations Research*, 24(10), pp. 981-990.

Thompson, R. J. and Visser, A. T., 2006. Employing best practice in designing, constructing, monitoring and maintaining haul roads. *Perth, Australia, Mine Machine Productivity*, pp. 1-19.

Thorley, U., 2012. Open pit mine planning: Analysis and system modeling of conventional and oil sands applications, *PhD Thesis,* Kingston, Ontario: Queen's University.

Tolwinski, B. and Underwood, R. 1992. An algorithm to estimate the optimal evolution of an open pit mine. In: *Proceedings of the 23rd International Symposium on the Application of Computers and Operations Research in the Minerals Industries*, pp. 399-409.

Verhoeff, R.L.A., 2017. Using genetic algorithms for underground stope design optimisation in mining: A stochastic analysis. *MSc Thesis,* Netherland: Delft University of Technology.

Wetherelt, A. and Van der Wielen, K. P., 2011. Chapter 10.1: Introduction to open-pit mining. *In: SME Mining Engineering Handbook. Cornwall: Society for Mining, Metallurgy, and Exploration,* pp. 857-876.

Whittle, D., 2011. Chapter 10.2: Open-pit planning and design. In: *SME Mining Engineering Handbook, Society for Mining, Metallurgy, and Exploration,* pp. 877-901.

Wright, E.A., 1999. Moving cone II – A simple algorithm for optimum pit limits design. *28th International Symposium on Application of Computers and Operations Research in the Mineral Industry*, pp. 367–374. Colorado School of Mines: CSM.

8 APPENDIX

Genetic algorithm Python code using Zuck Small input files

```
import csv
import random
import time

preceds_file = "zuck_small_preceds.csv"
values_file = "zuck_small_pit.csv"
coordinates_file = "zuck_small_main.csv"
minimum_size = 500
maximum_size = 600
max_times = 40
size_of_top_best = 20
index_of_preceds = 1
index_of_tonnage = 6
index_of_z_coordinate = 3

start = time.time()

with open(preceds_file, newline='') as cvs_preceds_file:
    preceds_lines = list(csv.reader(cvs_preceds_file, delimiter=';'))


    def generate_initial_search_space():

        list_0 = []

        for line_0 in preceds_lines:
            list_0.append(int(line_0[0]))

        return list_0


    def get_preced_line(generated_block):

        return preceds_lines[generated_block]


    def generate_block_not_in_block_list(search_space_1, blocked_list):

        choices = [x for x in search_space_1 if (x not in blocked_list)]

        if len(choices) == 0:

            return -1

        else:

            return random.choice(choices)


    def add_preceds_to_list(new_list, block_id, excluded):

        line = get_preced_line(block_id)
```

```
        num_preceds_in = int(line[index_of_preceds])
        if num_preceds_in > 0:

            for index in range(num_preceds_in):

                preced = int(line[index + 2])

                if preced not in new_list and preced not in excluded:
                    new_list.append(preced)

        return new_list


    def get_block_id_direct_preceds(block_id):

        line = get_preced_line(block_id)
        num_preceds_in = int(line[index_of_preceds])

        new_list = []

        if num_preceds_in > 0:

            for index in range(num_preceds_in):
                preced = int(line[index + 2])

                new_list.append(preced)

        return new_list


    def get_block_id_preceds(block_id, found, excluded_list):


        line = get_preced_line(block_id)
        num_preceds_in = int(line[index_of_preceds])

        new_list = []

        if num_preceds_in > 0:

            for index in range(num_preceds_in):

                preced = int(line[index + 2])

                if preced not in excluded_list:
                    new_list.append(preced)

        index = 0
        len_of_new = len(new_list)

        while index < len_of_new:

            preced_id = new_list[index]

            if preced_id not in found:
                preceds_of_preced = get_block_id_preceds(preced_id,
found, excluded_list)
```

```
                found = merge_to_list_distinct(found, preceds_of_preced)
                new_list = merge_to_list_distinct(new_list,
preceds_of_preced)

            index += 1

        return new_list


    def get_block_id_with_preceds_of_preceds(block_id, excluded_list):

        line = get_preced_line(block_id)
        num_preceds_in = int(line[index_of_preceds])

        new_list = [block_id]

        if num_preceds_in > 0:

            for index in range(num_preceds_in):

                preced = int(line[index + 2])

                if preced not in excluded_list:
                    new_list.append(preced)

        index = 1
        len_of_new = len(new_list)


        found = [block_id]

        while index < len_of_new:
            preced_id = new_list[index]
            preceds_of_preced = get_block_id_preceds(preced_id, found,
excluded_list)
            found = merge_to_list_distinct(found, preceds_of_preced)
            new_list = merge_to_list_distinct(new_list,
preceds_of_preced)

            index += 1

        return new_list


    def merge_to_list_distinct(old_list, new_list):


        if len(new_list) == 0:
            return old_list

        list__ = old_list.copy()

        for block_in_new_list in new_list:

            if block_in_new_list not in old_list:
                list__.append(block_in_new_list)
```

```
        return list__


    def remove_list_items(old_list, items):

        list__ = old_list.copy()

        for kk in items:

            if kk in old_list:
                list__.remove(kk)

        return list__


    def generate_list_of_list(search_space_1, excluded_list_1):

        list_of_list = []
        excluded_list = excluded_list_1.copy()
        none_duplicated = [excluded_list.copy()]
        local_search_space = search_space_1.copy()

        count = 0
        while count < max_times:

            new_block =
generate_block_not_in_block_list(local_search_space, none_duplicated)

            if new_block == -1:

                print("No more blocks can be generated, Error with
list")

            else:

                block_list =
get_block_id_with_preceds_of_preceds(new_block, excluded_list)

                num_gens = 0

                while len(block_list) < minimum_size and len(block_list)
!= len(search_space_1):

                    new_gen_in =
generate_block_not_in_block_list(local_search_space, block_list)

                    if new_gen_in == -1:

                        print("No more blocks can be generated, Error
with list")

                    else:
                        num_gens += 1

                        generated_new =
get_block_id_with_preceds_of_preceds(new_gen_in, excluded_list)
```

```
                        clean = []

                        for block in generated_new:

                            if block not in block_list:
                                clean.append(block)

                        if len(block_list) + len(clean) <= maximum_size:
                            block_list =
merge_to_list_distinct(block_list, clean)

                if minimum_size <= len(block_list) <= maximum_size:
                    list_of_list.append(block_list)

                    count += 1

                    none_duplicated =
merge_to_list_distinct(none_duplicated, [new_block])

        return list_of_list


    with open(values_file, newline='') as cvs_values_file:
        value_lines = list(csv.reader(cvs_values_file, delimiter=';'))


    def get_value_line(generated_block):
        return value_lines[generated_block]


    def get_sorted_by_value_dic(dictionary):

        return sorted(dictionary.items(), key=lambda x: x[1],
reverse=True)


    index_of_value = 1


    def get_position_values_dic(list_of_list):

        position_values = {}

        position_counter = 0

        for block_ids in list_of_list:

            final_value = float('0')

            for block_id in block_ids:
                value_line = get_value_line(block_id)

                block_value = float(value_line[index_of_value])

                final_value += block_value

            position_values[position_counter] = final_value
```

```
            position_counter += 1

        return position_values


    def get_block_with_values_dictionary(block_list_1):

        dic = {}

        for block_id in block_list_1:

            value = float(get_value_line(block_id)[index_of_value])
            dic[block_id] = value

        return dic


    def get_weakest_block_id(list_best):

        return
(sorted(get_block_with_values_dictionary(list_best).items(), key=lambda
x: x[1])[0])[0]


    def get_strongest_block_id(list_weak):

        return
(sorted(get_block_with_values_dictionary(list_weak).items(), key=lambda
x: x[1], reverse=True)[0])[0]


    with open(coordinates_file, newline='') as cvs_coordinates_file:
        coordinate_lines = list(csv.reader(cvs_coordinates_file,
delimiter=';'))


    def get_coordinate_line(generated_block):
        return coordinate_lines[generated_block]


    def get_lowest_value_block_id_at_bottom(waste):

        lowest_z_coordinate = -1

        bottom_blocks = []

        for row in waste:

            coordinate_line = get_coordinate_line(row)

            coordinate_z = int(coordinate_line[index_of_z_coordinate])

            if lowest_z_coordinate == -1:

                lowest_z_coordinate = coordinate_z
                bottom_blocks.append(row)
```

```
        else:

            if coordinate_z < lowest_z_coordinate:
                lowest_z_coordinate = coordinate_z
                bottom_blocks = [row]

            elif coordinate_z == lowest_z_coordinate:

                bottom_blocks.append(row)


    return get_weakest_block_id(bottom_blocks)


def get_highest_value_block_id_at_top(ore):

    top_z_coordinate = -1

    top_blocks = []

    for row in ore:

        coordinate_line = get_coordinate_line(row)

        coordinate_z = int(coordinate_line[index_of_z_coordinate])

        if top_z_coordinate == -1:

            top_z_coordinate = coordinate_z
            top_blocks.append(row)

        else:

            if coordinate_z > top_z_coordinate:
                top_z_coordinate = coordinate_z
                top_blocks = [row]

            elif coordinate_z == top_z_coordinate:

                top_blocks.append(row)


    return get_strongest_block_id(top_blocks)


def get_bottom_blocks(list_blocks):

    lowest_z_coordinate = -1

    bottom_blocks = []

    for row in list_blocks:

        coordinate_line = get_coordinate_line(row)

        coordinate_z = int(coordinate_line[index_of_z_coordinate])
```

```
        if lowest_z_coordinate == -1:

            lowest_z_coordinate = coordinate_z
            bottom_blocks.append(row)

        else:

            if coordinate_z < lowest_z_coordinate:
                lowest_z_coordinate = coordinate_z
                bottom_blocks = [row]

            elif coordinate_z == lowest_z_coordinate:

                bottom_blocks.append(row)

    return bottom_blocks


def get_top_blocks_without_preceds(list_blocks, excluded):

    blocks = []

    for block in list_blocks:

        line = get_preced_line(block)
        num_preceds = int(line[index_of_preceds])

        if num_preceds == 0:

            blocks.append(block)

        else:

            preceds = get_block_id_direct_preceds(block)

            if list_contains_all_values(excluded, preceds):
                blocks.append(block)

    return blocks


def get_blocks_value(list_blocks):

    value = float('0')
    for block_id in list_blocks:
        value_line = get_value_line(block_id)
        block_value_1 = float(value_line[index_of_value])
        value += block_value_1

    return value


def get_ore_blocks(list_blocks):

    blocks = []

    for block in list_blocks:
```

```
            block_value_1 = float(get_value_line(block)[index_of_value])
            if block_value_1 >= 0:
                blocks.append(block)

        return blocks


    def get_top_blocks(list_blocks):

        top_z_coordinate = -1

        top_blocks = []

        for row in list_blocks:

            coordinate_line = get_coordinate_line(row)

            coordinate_z = int(coordinate_line[index_of_z_coordinate])

            if top_z_coordinate == -1:

                top_z_coordinate = coordinate_z
                top_blocks.append(row)

            else:

                if coordinate_z > top_z_coordinate:
                    top_z_coordinate = coordinate_z
                    top_blocks = [row]

                elif coordinate_z == top_z_coordinate:

                    top_blocks.append(row)

        return top_blocks


    def get_highest_value_blocks_of_size(list_blocks, size):

        sorted_dict =
sorted(get_block_with_values_dictionary(list_blocks).items(), key=lambda
x: x[1], reverse=True)

        blocks = []

        counter_1 = 0

        while counter_1 < size:
            block_id = sorted_dict[counter_1][0]

            counter_1 += 1
            blocks.append(block_id)

        return blocks
```

```
    def list_contains_all_values(list_blocks, check_list):

        for block in check_list:

            if block not in list_blocks:
                return False

        return True


    def get_mutated_list(search_space_11, list__, excluded_list):

        list_temp = list__.copy()
        search_space_1 = search_space_11.copy()

        if len(list_temp) < maximum_size:
            open_spaces_of_list = maximum_size - len(list_temp)
            original_minus_list = remove_list_items(search_space_1,
list_temp)
            available_ore_top_blocks =
get_top_blocks_without_preceds(original_minus_list, excluded_list)
            available_ore_top_blocks_size =
len(available_ore_top_blocks)

            to_add = available_ore_top_blocks

            if available_ore_top_blocks_size > open_spaces_of_list:
                to_add =
get_highest_value_blocks_of_size(available_ore_top_blocks,
open_spaces_of_list)

            list_temp = merge_to_list_distinct(list_temp, to_add)

        return list_temp


    def get_crossed_mutated_lists(search_space_11, list_11, list_22,
excluded_list):

        list_1 = list_11.copy()
        list_2 = list_22.copy()
        search_space_1 = search_space_11.copy()

        bottom_blocks_1 = get_bottom_blocks(list_1)
        bottom_blocks_2 = get_bottom_blocks(list_2)

        top_blocks_1 = get_top_blocks_without_preceds(list_1,
excluded_list)
        top_blocks_2 = get_top_blocks_without_preceds(list_2,
excluded_list)

        bottom_blocks_value_1 = get_blocks_value(bottom_blocks_1)
        bottom_blocks_value_2 = get_blocks_value(bottom_blocks_2)

        top_blocks_with_ore_1 =
get_top_blocks_without_preceds(top_blocks_1, excluded_list)
        top_blocks_with_ore_2 =
```

```
get_top_blocks_without_preceds(top_blocks_2, excluded_list)

        top_blocks_with_ore_value_1 =
get_blocks_value(top_blocks_with_ore_1)
        top_blocks_with_ore_value_2 =
get_blocks_value(top_blocks_with_ore_2)

        less_than_1 = bottom_blocks_value_1 <
top_blocks_with_ore_value_2
        list_does_not_contains_all_values_1 =
list_contains_all_values(list_1, top_blocks_with_ore_2) is False
        enter_if_statement_1 = less_than_1 and
list_does_not_contains_all_values_1

        if enter_if_statement_1:

            list_1_temp = list_1.copy()

            list_1_temp = remove_list_items(list_1_temp,
bottom_blocks_1)

            list_1_temp = merge_to_list_distinct(list_1_temp,
top_blocks_with_ore_2)

            if minimum_size <= len(list_1_temp) <= maximum_size:
                list_1 = list_1_temp

        less_than_2 = bottom_blocks_value_2 <
top_blocks_with_ore_value_1
        list_does_not_contains_all_values_2 =
list_contains_all_values(list_2, top_blocks_with_ore_1) is False
        enter_if_statement_2 = less_than_2 and
list_does_not_contains_all_values_2

        if enter_if_statement_2:

            list_2_temp = remove_list_items(list_2, bottom_blocks_2)
            list_2_temp = merge_to_list_distinct(list_2_temp,
top_blocks_with_ore_1)

            if minimum_size <= len(list_2_temp) <= maximum_size:
                list_2 = list_2_temp

        list_1 = get_mutated_list(search_space_1, list_1, excluded_list)
        list_2 = get_mutated_list(search_space_1, list_2, excluded_list)


        crossed_0 = [list_1, list_2]

        return crossed_0


    def get_top_best_in_value_of_size(list_of_list, size):

        sorted_dic =
get_sorted_by_value_dic(get_position_values_dic(list_of_list))
```

```
        counter_0 = 0
        best = []

        while counter_0 < size:
            dic_item = sorted_dic[counter_0]
            position_in_set = dic_item[0]

            counter_0 += 1
            best.append(list_of_list[position_in_set])

        return best


    def merge_to_list_of_lists_distinct(list_of_list_11,
list_of_list_22):

        list_of_list_1 = list_of_list_11.copy()
        list_of_list_2 = list_of_list_22.copy()

        counter_q = 0

        while counter_q < len(list_of_list_1):

            list_item = list_of_list_1[counter_q]
            sorted_list_item = sorted(list_item)
            list_of_list_1[counter_q] = sorted_list_item

            counter_q += 1

        counter_q = 0

        while counter_q < len(list_of_list_2):

            list_item = list_of_list_2[counter_q]
            sorted_list_item = sorted(list_item)
            list_of_list_2[counter_q] = sorted_list_item

            counter_q += 1

        combined = list_of_list_1.copy()

        for list_2 in list_of_list_2:

            if list_2 not in list_of_list_1:

                combined.append(list_2.copy())

        return combined


    def get_left_overs(list_of_list):

        sorted_dic =
get_sorted_by_value_dic(get_position_values_dic(list_of_list))

        left_overs_list = []
```

```
    counter_0 = size_of_top_best

    while counter_0 < len(list_of_list):
        dic_item = sorted_dic[counter_0]
        position_in_set = dic_item[0]

        counter_0 += 1
        left_overs_list.append(list_of_list[position_in_set])

    return left_overs_list


def get_total_tonnage_value(list_):

    tonnage = float('0')

    for block_id in list_:
        line = get_coordinate_line(block_id)

        value_1 = float(line[index_of_tonnage])

        tonnage += value_1

    return tonnage


def get_ore_tonnage_value(list_):

    tonnage = float('0')

    for block_id in list_:

        value_line = get_value_line(block_id)

        block_value = float(value_line[index_of_value])

        if block_value >= float(0):
            line = get_coordinate_line(block_id)

            value_1 = float(line[index_of_tonnage])

            tonnage += value_1

    return tonnage

def remove_list_items22(old_list, items):

    list__ = old_list.copy()

    for kk in items:
        if kk not in list__:
            print('remove_list_items22: ', kk)
        list__.remove(kk)

    return list__

def get_value_of_generation(generation):
```

```
        total_value = float('0')

        for list_0 in generation:

            value_of_list = get_blocks_value(list_0)

            total_value += value_of_list

        return total_value

    def generate_time_periods_data():

        print('Generating best block combination for each period:')
        search_space = generate_initial_search_space()
        best_list_of_last_gen = []
        used_best_list_of_last_gen = []
        time_period_counter = 0
        time_period_data = []

        while len(search_space) > maximum_size:
            print('Processing result for time period: ',
time_period_counter + 1)
            generated = generate_list_of_list(search_space,
used_best_list_of_last_gen)
            generation = get_top_best_in_value_of_size(generated,
len(generated))
            generations = [generation]

            counter = 0
            value_of_prev_gen = float(0)
            value_of_last_gen = get_value_of_generation(generation)
            entered_while = False

            while value_of_last_gen != value_of_prev_gen or
entered_while is False:

                entered_while = True

                top_best = get_top_best_in_value_of_size(generation,
size_of_top_best)
                left_overs = get_left_overs(generation)

                counter_in = 0
                cross_mutated_list = []

                while counter_in < size_of_top_best:
                    top_list = top_best[counter_in]

                    bottom_list = top_best[counter_in + 1]


                    crossed_mutated =
get_crossed_mutated_lists(search_space,

top_list,
```

```
bottom_list,

best_list_of_last_gen)

                    cross_mutated_list.append(crossed_mutated[0])
                    cross_mutated_list.append(crossed_mutated[1])

                    counter_in += 2
                temp_generation =
merge_to_list_of_lists_distinct(top_best, cross_mutated_list)

                temp_generation =
merge_to_list_of_lists_distinct(temp_generation, left_overs)

                generation =
get_top_best_in_value_of_size(temp_generation, max_times)

                generations.append(generation)

                value_of_prev_gen = value_of_last_gen
                value_of_last_gen = get_value_of_generation(generation)

                counter += 1

            average_values = []


            for generation in generations:

                total_value = float('0')

                for list_0 in generation:
                    value_of_list = get_blocks_value(list_0)


                    total_value += value_of_list

                average_value = total_value / float(len(generation))

                average_values.append(average_value)

            counter = 0
            while counter < len(average_values):
                print("Average Value for generation: %s is: %s USD" %
(counter, average_values[counter]))

                counter += 1

            counter = 0
            while counter < len(generations):
                generation = generations[counter]
                best_list = generation[0]
                best_list_value = get_blocks_value(best_list)

                print("Best list for generation %s is: \n%s \n has
length of %s blocks and value of : %s USD " %
                      (counter, best_list, len(best_list),
```

```
best_list_value))
                print("has total tonnage of: %s tonnes and ore tonnage
of : %s tonnes "
                      % (get_total_tonnage_value(best_list),
get_ore_tonnage_value(best_list)))
                print('------------- Generation separator --------------
---')
                counter += 1

            last_gen = generations[len(generations) - 1]

            best_list_of_last_gen = last_gen[0]

            time_period_data.append(best_list_of_last_gen)

            for b in best_list_of_last_gen:
                used_best_list_of_last_gen.append(b)

            search_space = remove_list_items(search_space,
best_list_of_last_gen)

            time_period_counter += 1
            print("Best list for time period %s is: \n%s \n has length
of %s blocks and value of : %s USD " %
                  (time_period_counter, best_list_of_last_gen,
                   len(best_list_of_last_gen),
                   get_blocks_value(best_list_of_last_gen)))

            print("has total tonnage of: %s tonnes and ore tonnage of :
%s tonnes " %
                  (get_total_tonnage_value(best_list_of_last_gen),
                   get_ore_tonnage_value(best_list_of_last_gen)))
            print('---------------------------------- Period separator
----------------------------------------')

        time_period_data.append(search_space)
        print("Best list for the last time period: %s is: \n%s \n has
length of: %s blocks, total tonnage of: %s "
              "tonnes, ore tonnage of : %s "
              "tonnes and value of: %s USD" %
              (time_period_counter + 1, search_space, len(search_space),
get_total_tonnage_value(search_space),
               get_ore_tonnage_value(search_space),
               get_blocks_value(search_space)))

        return time_period_data

    def write_results_to_file(results):

        with open("block_period_zuck_small.csv", "w", newline="") as f:
            writer = csv.writer(f)
            writer.writerows(results)


    data = generate_time_periods_data()

    write_results_to_file(data)
```

```
end = time.time()

print("Total time it took to finish list of size: %s is: %s seconds " %
(max_times, end - start))
```