Genetic Algorithm Framework for Stochastic Open Pit Mining Optimization Problems

by

Shadrach Yaw Amponsah

Thesis submitted in partial fulfillment

of the requirements for the degree of

Master of Applied Science (MASc.) in Natural Resources Engineering

The Faculty of Graduate Studies

Laurentian University

Sudbury, Ontario, Canada

Please leave blank

This thesis is dedicated to Lael

ABSTRACT

Genetic algorithm (GA) is a metaheuristic evolution algorithm that has been applied in the area of combinatorial optimization problems. One of these areas of optimization is mine production scheduling. Mine production scheduling is the process of determining the sequence of block extractions over a period of time that will yield the maximum net present value (NPV) for the mining operation subject to a set of constraints. The block extraction sequence needs to occur under certain resource constraints, which presents an NP-hard problem. NP-hard problems are computationally intractable and complex to solve for large-scale problems using exact mathematical models such as Mixed Integer Linear Programming (MILP) and Stochastic Mixed Integer Linear Programming (SMILP).

In the mining production scheduling problem, the conventional approach is to use a single interpolated orebody model as the basis for production scheduling. This approach, however, does not consider grade uncertainties. These uncertainties have a significant impact on the NPV and can only be accounted for when modelled as part of the optimization problem. However, modeling these as part of the optimization problem increases the complexities associated with the production scheduling. In this research, a metaheuristic optimization framework based on GA was designed and implemented to solve the NP-hard large-scale open pit production scheduling (OPPS) problem. The problem definition and the resource constraints were formulated and optimized using a specially designed mining-specific GA. A multiple chromosome encoding technique was used in the GA to handle partial block processing to obtain a near-optimal solution.

Two case studies from an oil sands dataset were presented in this research. A stochastic formulation (SGA) of the OPPS problem to incorporate grade uncertainty based on equally probable orebody realizations was considered and optimized with the GA framework. A deterministic approach (DGA) to the OPPS problem that did not consider grade variability was also presented. The NPV and computational time from the DGA scenario were compared to a MILP model solved with CPLEX and the SGA scenario was compared to a SMILP model solved with CPLEX as a means to validate and analyze the GA results. There was a 53.6% improvement in the computational time for the DGA compared to the MILP

model with CPLEX in Case Study 1. However, the NPV was within 5.1% of the MILP model with CPLEX. The SGA model for Case Study 1 generated an NPV within 5.3% of the NPV of the SMILP model with CPLEX. Conversely, the result of the SGA model was generated in a computational time that was 75.2% better than the SMILP model with CPLEX. For Case Study 2, whereas the MILP model generated NPV of $10,175 M at a gap of 10% after 226 hours, the DGA model generated NPV of $9,142 M at 12.9% gap after 1.3 hours. Additionally, while the SMILP model was at a gap of 101% after 28 days, the SGA model generated NPV of $10,045 M at 10.6% gap after 1.5 hours.

Keywords

genetic algorithm, metaheuristic framework, production scheduling optimization, open pit mining, stochastic programming, Sequential Gaussian Simulation, grade uncertainty, Ordinary Kriging

ACKNOWLEDGEMENTS

I give thanks to God Almighty for the grace and mercy He has shown me throughout my MASc journey. I am eternally grateful to Him.

I would also like to express my heartfelt gratitude to Dr. Eugene Ben-Awuah, my supervisor, for the funding, guidance, support, time, ideas, and motivation he gave me throughout my MASc experience. I know for certain that there was no way I would have made it this far without him. My sincere thanks and appreciation go to my advisory committee members, Dr. Pawoumodom Matthias Takouda and Dr. Dean Millar, for their wisdom, insight, and comments they provided to me.

I am also extremely grateful for the insight from my colleagues Obinna Mbadozie, Dr. Ahlam Maremi, Pedro Pablo Vasquez-Coronado, Dr. Bight Oppong Affum, Precious Eme and Emmanuel Appianing.

Special thanks to my parents, Mr. Amponsah Benjamin and Mrs Owiredua Deborah for their incredible love and support. My gratitude also goes to Petra Demeyere for her unending support and prayers throughout my stay in Sudbury, Ontario.

I deeply express my gratitude and will forever remain thankful to my wife, Jedidah, for her patience, love, and the faith she has shown towards me all these years in attaining this feat. *I owe you more than love.*

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

ACO	Ant Colony Optimization
DP	Dynamic Programming
EBV	Economic Block Value
GA	Genetic Algorithm
GDR	Geologic Discount Rate
GP	Goal Programming
IP	Integer Programming
LG	Lerchs-Grossman algorithm
LP	Linear Programming
LPE	Literal Permutation Encoding
MIP	Mixed Integer Programming
MILGP	Mixed Integer Linear Goal Programming
MILP	Mixed Integer Linear Programming
MPM	Mathematical Programming Model
NPV	Net Present Value
NSGA-II	Non-dominated Sorting Genetic Algorithm
OK	Ordinary Kriging
OPPS	Open Pit Production Scheduling
PSO	Particle Swarm Optimization
SA	Simulated Annealing
SIP	Stochastic Integer Programming
SGS	Sequential Gaussian Simulation
SMILP	Stochastic Mixed Integer Linear Programming
TS	Tabu Search

TSP	Traveling Salesman Problem
UPL	Ultimate Pit Limit
VND	Variable Neighbourhood Descent

CHAPTER 1

INTRODUCTION

1.1 Background

Genetic algorithm is a metaheuristic evolutionary algorithm that has widely been studied and applied in combinatorial optimization problems in the areas of Finance, Supply Chain Management, and Information Technology. Researchers have investigated its application to mining-related optimization problems as well. These applications span across different aspects of mine planning optimization. This includes Denby and Schofield (1994), Myburgh and Deb (2010), Alipour et al. (2017), Paithankar and Chatterjee (2019) and Alipour et al. (2020) who used genetic algorithm for open pit production scheduling optimization. Ahmadi and Shahabi (2018) also used genetic algorithm for cut-off grade optimization while Ruiseco et al. (2016) used genetic algorithm in ore-waste pit limit optimization. Similarly, Franco-Sepúlveda et al. (2019) used genetic algorithm for the optimization of open-pit mining operations with geological and market uncertainty.

Mine planning optimization is a complex yet necessary combinatorial optimization problem. Combinatorial optimization problems are problems whose optimal solution must be obtained from a finite number of possibilities (Anani, 2016). Mine planning optimization specifies the source, destination, time, and extraction sequence of mineral resources that maximizes the net present value (NPV) of a mining operation. A resultant activity of mine planning is the production scheduling of the mineral resource from the mine. Production scheduling can be carried out on either an open pit or an underground mine with various physical and technical constraints. Production scheduling can also be classified into short term, medium term or long term. Caccetta and Hill (2003) described the time and sequence of extracting blocks from an open pit mine in order to maximize the NPV of the mining project as open pit production scheduling (OPPS) optimization. Alipour et al. (2017) referred to the OPPS problem as a combinatorial optimization problem in the class of Non-deterministic polynomial-time (NP) hardness. An optimization problem is said to be NP-hard if the

algorithm for solving it can be converted to one for solving any NP problem. NP-hard therefore means "at least as hard as any NP-problem," although it might, in fact, be harder (Weisstein, 2022). As the OPPS optimization problem gets larger and the number of integer variables increase, finding an optimal solution to the problem in some instances becomes intractable or uses too much computing resources when an exact solution methodology is implemented. An optimization problem is tractable if a solution is obtained in polynomial time. This solution may or may not be optimal. Setting an optimality gap for the optimization problem can ensure tractability for exact algorithms. An optimality gap, therefore, refers to the difference between the best known solution (best integer) and the value that bounds the best possible solution (IBM ILOG CPLEX Inc., 2017). An optimal solution in the case of exact algorithms is a solution with a 0% optimality gap. This demonstrates that the solution is the best that exists because the difference between the best integer and the best bound is 0%.

This research focuses on using a metaheuristic optimization approach in the field of evolutionary algorithms to tackle the NP-hard large scale OPPS problem. Bianchi et al. (2009) define metaheuristics as algorithms that incorporate or develop heuristics (heuristics are simple approximate algorithms that look for good solutions in a solution space) to solve an optimization problem in a generic framework. Metaheuristics are thus higher level than heuristics, as the term "meta" in metaheuristics implies. The concept of metaheuristics is mostly inspired by natural biology.

The conventional approach to orebody modelling based on Ordinary Kriging (Krige, 1951) generates a single interpolated block model for pit limit and production scheduling optimization. In using this single interpolated block model for production scheduling, geological uncertainties, which are inevitable in a typical mining project, are not taken into consideration. This may result in schedules that either 1) overestimate or 2) underestimate the true representation of the optimal solution. Researchers such as Dimitrakopoulos and Ramazan (2008), Sabour and Dimitrakopoulos (2011), and more recently Mbadozie (2022) have investigated the incorporation of grade uncertainty in the OPPS problem formulation using simulation and mathematical programming. Multiple realizations of the block model

are generated with Sequential Gaussian Simulation (SGS) and used as input to the mathematical programming model. The primary limitation of their implementation relate to generating tractable solutions for large scale optimization problems in a reasonable run time. The optimization solution run time is directly related to the problem size which is also a function of the size of the deposit, the number of simulation realizations, and the life-of-mine.

In this research, a metaheuristic optimization framework based on genetic algorithm has been designed and implemented for a large scale OPPS problem. A real number chromosome encoding technique is used in the genetic algorithm initial population to make possible partial block processing. Two variations of the GA framework referred to as Deterministic Genetic Algorithm (DGA) and Stochastic Genetic Algorithm (SGA) were implemented. DGA basically refers to the application of the GA framework in a conventional approach to production scheduling using an OK block model, while SGA refers to the application of the GA framework in a stochastic approach to production scheduling using SGS block model realizations. The conventional approach to the OPPS problem which does not consider grade uncertainty is investigated with a Mixed Integer Linear Programming (MILP) model with CPLEX and the DGA framework. The stochastic formulation of the OPPS problem that incorporates grade uncertainty is also considered and the resulting problem is optimized with a Stochastic Mixed Integer Linear Programming (SMILP) model with CPLEX and the SGA framework. The NPV and solution time for the conventional and stochastic frameworks are compared.

1.2 Statement of the problem

Mining is an industry that forms an integral part of the growth of many countries with access to mineable mineral resources. As a key industry, its processes require optimization to ensure that maximum profit and a great return on investment is achieved at all times. A major aspect of the processes involved in mining is the production scheduling activity. Production scheduling specifies the time and sequence in which mineable mineral reserves should be extracted from a mine in order to maximize the NPV of the mining project. In production scheduling, these mineable reserves are discretized into three-dimensional block model

representation of the mineral deposit. The challenge and the goal, therefore, is to extract this mineral deposit and process it in a way and manner that will ultimately maximize the profit of the mining operation.

The OPPS problem is a combinatorial optimization problem, and researchers in their bid to solve it have formulated mathematical models with different optimization techniques. Caccetta and Hill (2003) used a Mixed Integer Linear Programming (MILP) model and optimized it with a branch and cut approach with best-first and depth-first search strategies. Askari-Nasab *et al*. (2010) employed a MILP model with CPLEX solver (IBM ILOG CPLEX Inc., 2017) which is based on branch and cut optimization algorithm (Horst and Hoang, 1996). Their framework also featured a fuzzy logic clustering algorithm to aggregate blocks into "mining-cuts" reducing the number of variables in the optimization problem. Koushavand *et al.* (2014) also used a MILP model with CPLEX solver and a hierarchical clustering algorithm for blocks aggregation into mining-cuts. In Ben-Awuah *et al.*(2018) implementation, the authors employed a Mixed Integer Linear Goal Programming (MILGP) model with CPLEX solver and hierarchical clustering algorithm for blocks aggregation into mining-cuts and mining-panels. Their implementation also used a reward and penalty approach to prioritize important management goals in a multiple objective OPPS problem. These models generally take the form of an objective function for maximizing the NPV, subject to a set of constraints.

In the literature on solving the OPPS problem, deterministic and stochastic optimization models, which include heuristics and metaheuristics, are the two main approaches used by researchers for solving the problem. Deterministic approaches include methods such as Linear Programming (LP), Mixed Integer Linear Programming (MILP), and Mixed Integer Linear Goal Programming (MILGP). For many decades, MILP and MILGP models have been the most widely used optimization methodologies for OPPS since they guarantee optimal solution to the problem. However, the limitation of these approaches are evident in optimization problems for large scale deposits, which makes the problem difficult to solve or sometimes intractable. Even with today's computing power, large-scale MILP and MILGP

optimization problems emanating from the mining sector require excessive computing resources for those that converge or are computationally intractable.

The OPPS problem is complex as is, because of the different technical and physical constraints associated with the problem. With its complexities, the conventional approach to the problem does not include stochastic optimization to cater for grade uncertainty. Stochastic optimization is an optimization approach that includes or introduces randomness in the solution space. This includes random variables in the problem formulation and random iterates to obtain the solution (Fouskakis, D. and Draper, D., 2002). This therefore suits optimization problems that involves uncertain parameters such as grade uncertainty in the OPPS problem. Grade uncertainty affects the overall practicality of the generated production schedule. According to Dimitrakopoulos and Ramazan (2008), uncertainty largely contributes to the failure of mining operations to meet their production and financial targets, particularly during the early stages of mine life. Due to drill hole spacing and *in situ* grade variability, geological uncertainties are inevitable in a mining project (Mbadozie, 2022) and adds an extra layer of complexity to the OPPS optimization problem. It is therefore necessary for researchers to formulate and develop integrated optimization frameworks that are capable of optimizing the conventional and stochastic representations of the OPPS problem in a relatively shorter computational time. This motivated and led to this research on genetic algorithm framework for stochastic open pit optimization problems.

1.3 Summary of literature review

The planning of an open pit mine starts when exploration drilling for a potential mining location is undertaken and the resulting data is subsequently used to determine the size and extent of the ore body. This ore body must be relatively closer to the earth surface to warrant the use of surface mining methods to extract the ore. An ultimate pit limit definition which precedes the production scheduling process is a step carried out to determine the blocks (three-dimensional representation of the mineral deposit) that when included in the production schedule will maximize the cash flow of the mining operation. The main algorithm used in the literature to find the ultimate pit limit is the LG algorithm (Lerchs and Grossman, 1965). The LG algorithm is implemented in Whittle software (Geovia Dassault

Systems, 2017) which was used in this research. Once the final pit is determined and designed, the production scheduling process can commence. During production scheduling, blocks are classified as either waste or ore based on the current economics of the mine. Thus, blocks having a revenue higher than the cost of mining and processing can be categorized as ore or otherwise as waste. The production scheduling process can be grouped into two approaches: 1) conventional and 2) stochastic. The conventional approach does not consider grade uncertainty whereas grade uncertainty is modelled as part of the stochastic approach.

Mathematical programming models have been used throughout the literature to solve the OPPS problem. Researchers such as Dagdelen and Johnson (1986), Johnson (1968), and Gershon (1983) have proposed theories and formulations based on LP and MILP to solve the OPPS problem. Other methodologies such as goal programming (GP) (Zeleny, 1980; Zhang *et al.*, 1993) and MILGP (Ben-Awuah *et al.*, 2012) have also been formulated by these researchers in solving the OPPS problem. These researchers focused on the conventional approach of the OPPS problem.

A different class of optimization algorithms in the field of nature-based and evolution computing, known as metaheuristics, have also been investigated for production scheduling. In the application of heuristics and metaheuristics, the solution to an optimization problem often does not have one outcome. The same set of parameter values and initial conditions will lead to an ensemble of different outputs. Models based on heuristics and metaheuristics are capable of finding good, and sometimes optimal solutions to problem instances of realistic size within a generally shorter computation time than models based on exact methodologies (Bianchi *et al*., 2009). Popular metaheuristic optimization algorithms implemented in the literature for OPPS problems are: Ant Colony Optimization (ACO), Genetic Algorithms (GA), Particle Swarm Optimization (PSO), Tabu Search (TS), and Simulated Annealing (SA). Kumral and Dowd (2005) proposed a SA algorithm approach to solve the OPPS problem. An application of the ant colony algorithm was proposed by Shishvan and Sattarvand (2015) for OPPS. Lamghari and Dimitrakopoulos (2012) also used TS and Variable Neighbour Descent (VND) algorithms for attempting similar problems. Recently, Alipour *et al*., (2017) and Alipour *et al.*, (2020) presented a GA approach to the

OPPS problem. The problem size and complexity associated with OPPS optimization problems make the application of metaheuristics ideal since it is capable of finding a good solution in a much shorter computation time than models based on exact methodologies. Incorporating grade uncertainty in the stochastic approach to OPPS optimization increases the problem size and complexity hence the effort in this research to implement a metaheuristic optimization model based on GA for stochastic production scheduling optimization.

1.4 Objectives of the study

The principal objective of this research is to design, develop and test a GA framework for optimizing the mine production scheduling problem. The model was developed and verified with MATLAB (MathWorks Inc., 2020) programming language. A metaheuristic optimization model based on GA was used as the primary algorithm. To achieve the primary objective, the research includes the development of an incremental theoretical and conceptual framework that focuses on the following objectives:

- Develop a metaheuristic optimization model based on GA for optimizing the deterministic open pit mine production scheduling problem. This GA implementation is referred to in this research as the deterministic GA (DGA).
- Extend the GA optimization framework to incorporate grade uncertainties for a stochastic open pit mine production scheduling problem. This GA implementation is referred to in this research as the stochastic GA (SGA)
- Develop computer codes/tools that implement the formulated DGA and SGA frameworks for mining projects.

1.5 Context and scope of work

The context and focus of this research is to develop and implement a GA framework for long-term open pit production scheduling optimization. The GA framework takes into consideration the impact of grade uncertainty through multiple simulated orebody realizations. These orebody realizations are equally probable and are used as input to the GA

framework for optimization. The optimization is done on a block-by-block basis. The focus of this study consists of the following:

- Developing a DGA framework to optimize a conventional formulation of the OPPS problem based on a single orebody block model interpolated using Ordinary Kriging.
- Extending the DGA framework to a SGA to consider grade uncertainty using equally probable orebody realizations from Sequential Gaussian Simulation (SGS).

The following limitations apply and assumptions were made in developing the GA framework for this research:

- Future costs and selling price were kept constant, and rock type uncertainties were not considered.
- Each block in the GA model was assumed to be extracted over at most two periods.
- The data used for model implementation was an oil sands dataset obtained from Mbadozie (2020) which consists of ore, dyke and waste materials. In this research, dyke materials management were not considered.

1.6 Research methodology

The principal motivation for this research is to develop a GA model to optimize the open pit production scheduling problem and to find a solution in a faster computational time compared to an exact optimization approach. This therefore led to the development of the GA framework in this research, capable of solving the conventional or stochastic OPPS problem based on the user input data provided. To achieve this objective, relevant literature on metaheuristic optimization algorithms and stochastic OPPS problems were reviewed to form the core of how to approach the research problem. Figure 1-1 shows a summary of the research methodology used. Highlights of the research tasks completed to achieve the objectives of this study are as follows:

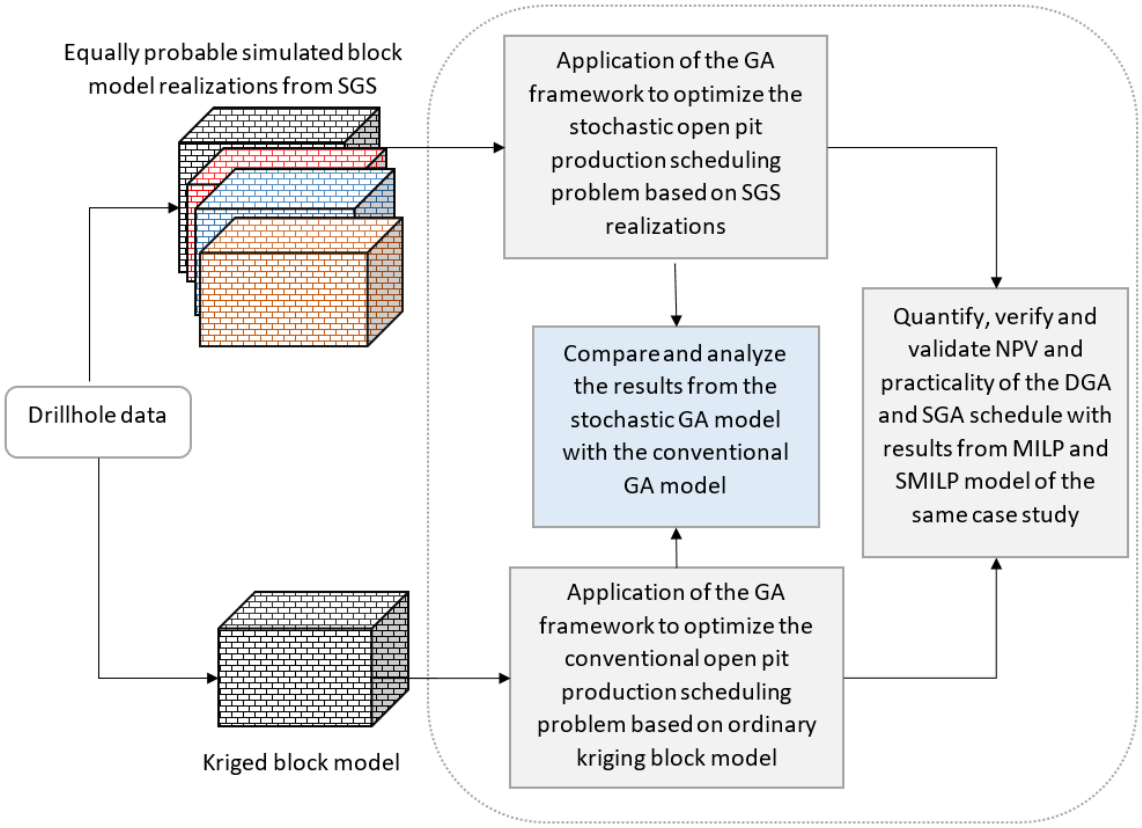

Figure 1-1: Summary of research methodology

- Create an initial population to adequately represent the problem definition accounting for multiple chromosome representations. This was implemented in the GA to allow for fractional block processing.
- Determine predecessor blocks for each block in the ultimate pit to guide the extraction sequence.
- Implement the GA framework to formulate and solve the conventional or stochastic OPPS optimization problem in a MATLAB environment.
- Verify and interpret the results by comparing results from the GA framework with similar mathematical programming model implementations based on MILP and SMILP by Mbadozie (2022). The case studies were based on oil sands mining.

- Provide documentation on the work flow and optimality assessment of the GA results in comparison with a relaxed LP solution which serves as the upper bound. This relaxed LP model is a version of the MILP or SMILP model where all integer variables are set as continuous variables.

1.7 Contributions of the research

The main contributions of this research is the development of an optimization framework based on GA that optimizes a conventional and a stochastic OPPS problem in the presence of grade uncertainty. The GA framework is computationally efficient in solving the OPPS problem. The highlights of the contributions made in this research are as follows:

- Developed an integrated GA framework that optimizes both the conventional and stochastic OPPS problems based on input data.
- Deployed a multiple chromosome encoding scheme in the GA framework to accommodate block processing over multiple periods.
- Documented a workflow to assess the optimality of the GA solution in comparison with the relaxed LP results.

1.8 Organization of thesis

This thesis is comprised of five chapters. The first chapter provides an introduction and a broad overview of the research. This chapter also includes the background of the research, followed by the problem definition and a summary of the literature review, the objectives of the study, the context and scope of the study, the applied research methodology, and the contributions of this research.

Chapter 2, which documents the literature review, provides an overview of open pit production scheduling. Literature related to genetic algorithms and genetic operators are discussed. The merits and demerits of constraints-handling methods in genetic algorithm applications are investigated. The implementation of genetic algorithms for mine production scheduling problems as opposed to linear programming and integer programming are reviewed. Stochastic optimization methodologies, heuristics and metaheuristics

optimization algorithms, and their applications to production scheduling problems are also discussed.

Chapter 3 details the theoretical formulation for both the conventional and stochastic open pit production scheduling problem. The chapter focuses on formulating, modeling, and developing the objective function, constraints, and their relationships. The GA problem representation and constraints handling are also discussed in this chapter.

Chapter 4 highlights the application of the methodologies and mathematical formulations of the genetic algorithm model discussed in the previous chapter. These are implemented for oil sands case studies. Relevant discussions on the performance of the GA framework and how it compares to MILP and SMILP models are provided for the case studies.

Finally, Chapter 5 contains the thesis summary and concluding statements. The benefits and contributions of this research and suggestions for future work on genetic algorithms for stochastic open pit optimization problems are summarized.

CHAPTER 2

LITERATURE REVIEW

2.1 Background

In this chapter, literature related to genetic algorithms and genetic operators are discussed. The merits and demerits of constraint-handling methods in genetic algorithm are investigated. The application of genetic algorithms to the mine production scheduling problem as opposed to linear programming and integer programming is reviewed. Stochastic optimization methodologies are also discussed. This chapter also discusses heuristic and other metaheuristic optimization algorithms and their applications to the mine production scheduling problem.

2.2 Genetic algorithm

Genetic algorithm (GA) is an evolutionary algorithm that follows biological processes as proposed by Darwin (Goldberg and Holland, 1988; Holland, 1992). GA, therefore, generates its solution to the optimization problem by strictly following the biological evolution process; such as inheritance, crossover, mutation, and selection. The inherent theory in this process is the survival of the fittest where organisms with good or fitter genes survive and transfer their genes to the next generation. Consequently, only organisms with the best gene will exist over time. GA follows the same approach when formulating problems. In summary, the GA workflow includes the following: 1) An initial population of individuals is created; the fitness functions of the created individuals are evaluated. 2) A set of genes and chromosomes are selected based on the fittest individuals; the selected genes will then crossover and mutate. 3) Elitism is then applied on the best individuals in the population to keep them for the next generation. 4) This process is repeated until a population of the best genes are obtained or a set of maximum generations are reached. Figure 2-1 shows a flow chart of the genetic algorithm process.

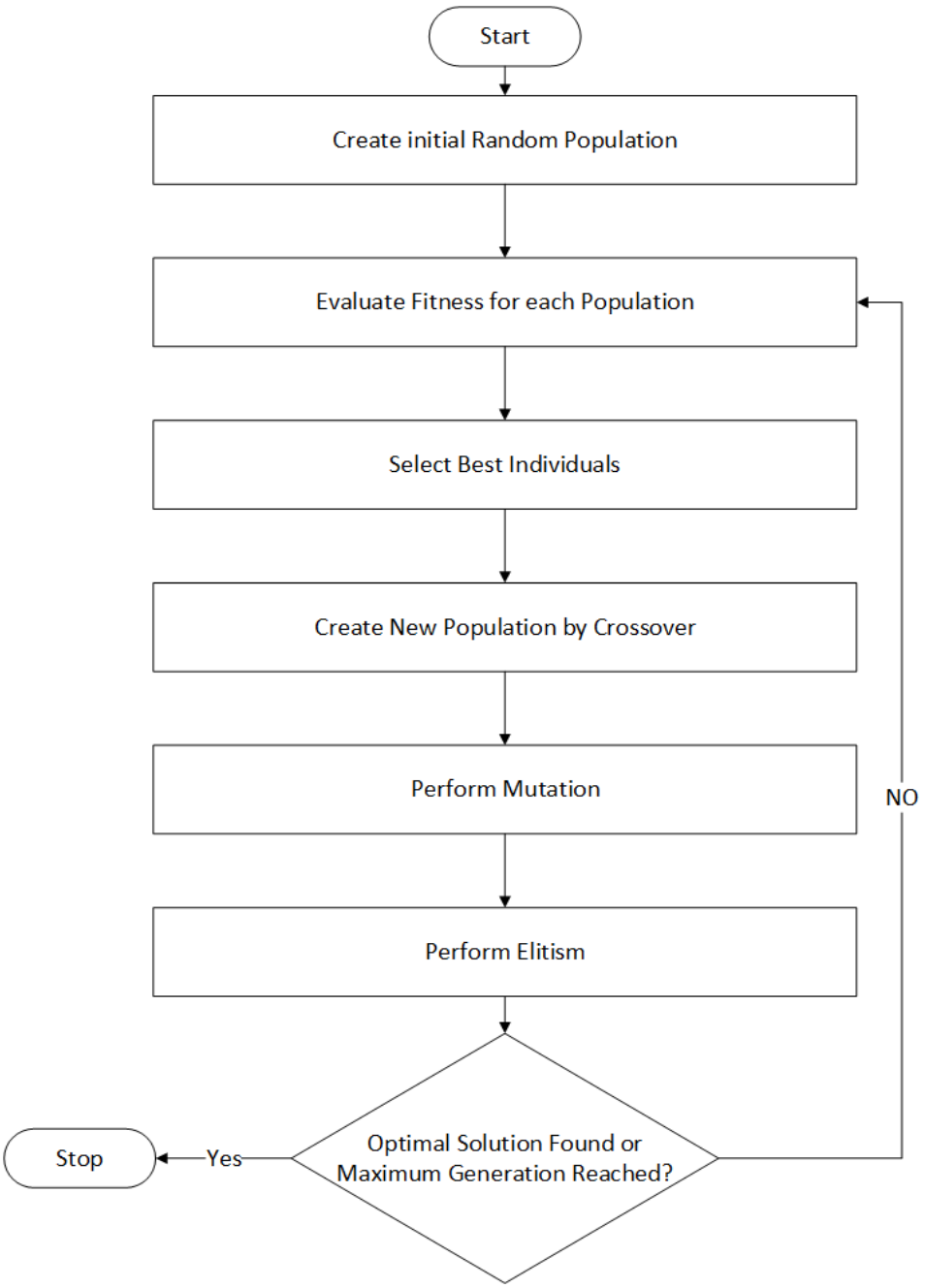

Figure 2-1: Genetic algorithm process

2.3 Initial population

This is the starting point of the GA where the population is initialized. According to Gen *et al.* (2008), there are two methods for generating initial population; the heuristic method and the random number generation method. In the random number generation method, GA randomly generates a solution space based on the problem size and this is referred to as the

initial population. This initial population can be generated from a Gaussian distribution to increase diversity. The population includes multiple solutions, which represents chromosomes and genes of individuals. Each chromosome has a set of variables which simulates the genes (Mirjalili, 2019). In the heuristic approach, a problem specific encoding algorithm is used to generate the initial population. Gen *et al.* (2008) however illustrated that this approach sometimes explore only a smaller portion of the solution space in a large scale combinatorial optimization problem. This then leads to a local optimum in the GA. In this research, the random number generation method was used. Figure 2-2 shows a sample initial population (Chromosomes 1, 2 and 3) obtained by the random number generation method. This figure shows the flexibility in setting up GA optimization problems as the initial population can be represented in different ways and with different characters.

Chromosome ***1***	0	0	1	0	1	0	1	0	0	1	1
Chromosome ***2***	3	5	7	4	8	1	8	12	56	4	7
Chromosome ***3***	0.3	0.2	0.6	0.05	0.6	0.52	0.41	0.3	0.3	0.54	0.8
~											
Chromosome ***n***	A	B	H	J	I	L	K	M	G	H	V

Chromosome / Solution

Genes

Figure 2-2: Sample initial population showing genes and chromosomes

2.4 Chromosome encoding

Chromosome encoding is an essential process in GA. It specifies the nature of the genotype in a population. The encoding scheme is mostly problem dependent and thus relies on the structure of the problem being optimized. Binary encoding, real number encoding, literal permutation encoding (LPE), and general data structure encoding are the various classifications of chromosome encoding according to Gen *et al.* (2008). In binary encoding, the genes in the population are represented by either 0 or 1. The encoded genes are then decoded to decimals when evaluating for their fitness. This process is done for every gene in the chromosome and may pose performance issues for large number of genes. This encoding forms the genotype of a feasible solution to the problem. An example of a problem

that benefits from the binary encoding scheme is the general knapsack problem. In the knapsack problem, the objective is to find the sum of weights producing the maximum profit or minimum cost to a problem while respecting the stipulated capacity of the knapsack. Given a set of n $\forall_n \in \{1,...n\}$ items each having a weight of w_i and a value of v_i with a maximum capacity of C, the knapsack problem can be modelled as in Equations (2.1) and (2.2) (Feng *et al.*, 2017).

$$Max \sum_{i=1}^{n} v_i x_i \tag{2.1}$$

Subject to:

$$\sum_{i=1}^{n} w_i x_i \leq C \qquad x_i \in \{0,1\} \tag{2.2}$$

Where x_i is the decision variable. The decision variable to this problem can be encoded as 0 or 1: $\begin{cases} 1 & if\ selected \\ 0 & otherwise \end{cases}$

When binary encoding is employed, it indicates that the problem assumes only discrete values, which is not always the case for many optimization problems.

Real number encoding or continuous variable encoding is the representation of the decision variables in the genotype with real numbers as opposed to binary encoding. In this process, there is no binary to decimal decoding and this improves the efficiency of the approach. According to Gen *et al.* (2008), real number encoding is suitable for functional and constrained optimization problems. The genotypic representation in real number encoding is close to the phenotypic space of the problem since there is no conversion between both spaces. To represent genes with this encoding scheme, genes have to be between a lower and upper bound of the decision variable. This is represented in Equation (2.3).

$$x_n = (x_{ub} - x_{lb}) \times r + x_{lb} \tag{2.3}$$

Where x_n is the *n-th* gene, x_{ub} is the upper bound, x_{lb} is the lower bound and r is a random number between [0, 1].

LPE or order encoding is the representation of the gene by the permutation of the decision variables. Since LPE is represented as the permutation of the decision variables, it is mostly suitable for optimization problems that involve permutation. An example is the travelling salesman problem. In the travelling salesman problem, a salesman has to visit *n* number of cities and every city can be visited only once. In such a problem, the genotype can be encoded as the order or sequence in which the cities are visited which is the permutation of the *n* cities. Figure 2-3 shows a sample LPE technique. Chromosomes A and B in Figure 2-3 show the sequence in which the cities can be visited by the travelling salesman as represented in GA.

Chromosome A	1	4	2	3	6	7	8	10	12	14	15	17	9	11	13	16	5

Chromosome B	2	4	3	5	7	6	8	10	15	14	16	12	11	13	17	1	9

Figure 2-3: Literal permutation encoding representation

Every combinatorial optimization problem that is optimized with GA requires its own chromosome encoding technique that represents the problem in great detail. There are no one size fits all chromosome encoding although certain type of combinatorial optimization problems do benefit from specific encoding schemes. Once a suitable encoding scheme is modelled for the problem, the various genetic operators can then be formulated around the specific chromosome encoded. The phenotype of the population is then derived from the genotype representation in the encoded chromosomes. In this research, multiple chromosome encoding was used to represent the genes in the population. A real number encoding or continuous encoding technique was used to represent the genotype of each block in the population. This allowed for fractions of blocks in the population to be processed. In addition, the LPE technique was also used as part of the encoding scheme to provide the order or sequence in which blocks could be mined. More on the problem specific chromosome encoding techniques used in this research are discussed in Chapter 3.5.

2.5 Fitness function

In GA, the fitness function is the function used to determine the viability of a gene in a population. The objective function in an optimization problem in GA is referred to as the fitness function. This function tests the population at every generation and becomes the means of determining fitter genes that survives to the next generation. The fitness function also aides in the selection of parents from the population as selection algorithms in GA ranks population based on their fitness value. When evaluating the fitness function, the genotype from the population is converted or decoded into phenotype. This phenotype is evaluated and assigned a value which becomes the fitness of that solution. The decoding of the gene is dependent on the chromosome encoding scheme used during the initialization of the problem. At the end of every generation, the fitness value of each population is assessed and ranked based on the objective of the optimization problem. If the objective of the optimization is to minimize cost, then the population with the least minimum fitness value is chosen as the best solution from that generation. If the objective of the optimization is to maximize profit, then the population with the maximum fitness value is selected. The fitness function and its evaluation ensures that the GA does not violate the general objective of the problem.

2.6 Selection

GA uses different selection algorithms with the inspiration of attaining the fittest individual from the initial population. Sharma and Gargi (2014), and Sivanandam and Deepa (2007) defined selection in GA as a method that randomly picks chromosomes out of the population according to their fitness function value. The higher the fitness function value, the better chance that an individual will be selected. There are several popular selection algorithms but there is no one preferred selection algorithm for GA. Each selection algorithm may have advantage over the other based on the specific problem being optimized. Various selection algorithms found in the literature are: Boltzmann selection (Goldberg, 1990), Roulette wheel selection also known as the Fitness proportionate selection algorithm (Cantú-Paz, 1998; Goldberg and Deb, 1991), Tournament selection (Blickle and Thiele, 1995; Goldberg and Deb, 1991), Random selection, Rank selection, and Stochastic universal sampling (Sivanandam and Deepa, 2007). Roulette wheel selection and Tournament selection are

explored for this research as they are the two well studied selection algorithms in GA (Kumar et al., 2010).

2.6.1 Roulette wheel selection

Roulette wheel or fitness proportionate selection is the original selection algorithm proposed by Goldberg and Holland (1988) in the GA. The roulette wheel selection is comparable to the traditional roulette wheel. In the roulette wheel, each individual is mapped to a portion on the wheel, and individuals with higher fitness are given larger portions to correspond with their fitness value. At every spin, a random number is generated and the individual whose portion the random number falls in is selected. The process is repeated until the required number of individuals are chosen. For a given set of N individuals having a fitness value of f_i, the probability of individual P_i being selected is given in Equation (2.4). Table 2-1 shows a pseudo code for the roulette wheel selection algorithm.

$$P_i = \frac{f_i}{\sum_{i=1}^{N} f_i} \qquad \forall i \in \{1, 2, ...N\} \tag{2.4}$$

Where $f_i > 0$;

Table 2-1: Pseudo code for roulette wheel selection

Pseudo Code for Roulette Wheel Selection
1**: Let P = Population; fi = fitness of members in the Population P; sum of probabilities =0;* ***2**: **For** i = 1; to size of P* *cumsum = Cumulatively sum the fitness fi* ***end for ***3**: **For** i = 1; to size of P* *probability = sum of probabilities + (fi / cumsum)* *sum of probabilities += probability* ***end for*** ***4**: **loop** until new population is full* *do this twice* *number = Random between 0 and 1* ***for** all members of population*

```
        if number > probability but less than next probability
            then you have been selected
      end for
   end
   create offspring
end loop
```

2.6.2 Tournament selection

Tournament selection is a more simpler selection methods to implement and one that has the least impact on computational time (Mirjalili, 2019). Tournaments are held randomly between two or more selected individuals from the population. The individuals in the selected population are ranked based on their fitness values. The best individuals are placed in a temporary population. The process is repeated based on the size of the tournament until a new population is generated (Blickle and Thiele, 1995). Table 2-2 shows a pseudo code for a tournament selection.

Table 2-2: Pseudo code for tournament selection

Pseudo Code for Tournament Selection
***1**: Choose the population i and size of tournament t*
***2**: Set best fitness b = null;*
***3**: **For** y = 1; to the size of t*
Pick individual z at random from the current population
***If** z fitness f is greater than best fitness b*
f = b;
end if
end for
***4**: **return** f*
***5**: Add best individual with fitness f to the mating pool*

2.7 Crossover

Crossover is the method of selecting two parents at random and recombining their chromosomes at a point with the intent of making offspring with better genes (Sivanandam and Deepa, 2008). Single point crossover and double point crossover are the two main approaches to crossover. Figure 2-4 shows a single and Figure 2-5 shows a double point crossover. However there are several other crossover approaches used in the literature: uniform crossover (Semenkin and Semenkina, 2012), three parents' crossover (Tsutsui *et*

al., 1999), half uniform crossover (Hu and Di Paolo, 2009), partially matched crossover (Goldberg and Lingle, 1985), position-based crossover (Syswerda, 1991), order crossover (Davis, 1985), cycle crossover (Oliver *et al.*, 1987), multi-point crossover (Eshelman *et al.*, 1989), masked crossover (Louis and Rawlins, 1991), and heuristic crossover (Grefenstette *et al.*, 1985). In this research, the double point crossover was implemented. Double point crossover was used since it has a high capacity to transmit useful genetic information from parent to offspring based on the study by Falkenauer (1999).

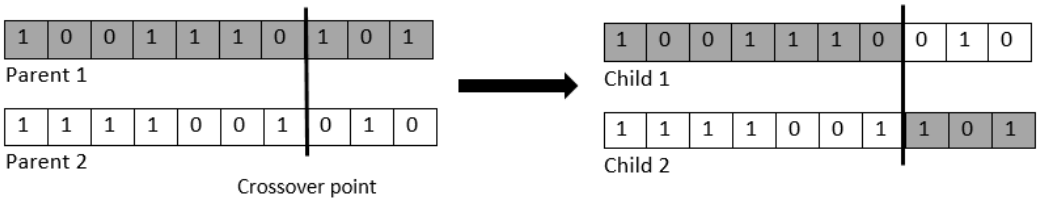

Figure 2-4: Single point crossover

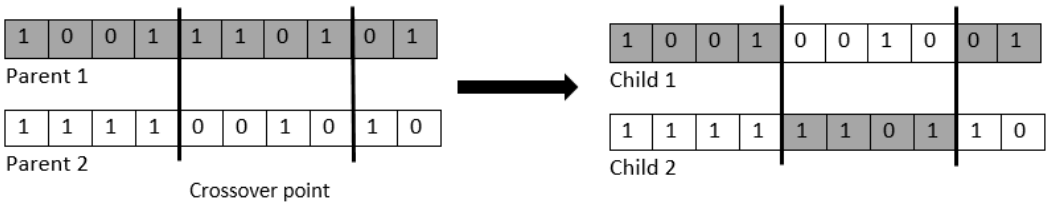

Figure 2-5: Double point crossover

2.8 Mutation

Mutation is the next stage after crossover in the GA algorithm process. When offspring from two parents are generated through crossover, it may occur that these offspring do not possess good enough genes to generate a good solution. Therefore, the GA process introduces mutation to alter or change the genes. Mutation is the other way to get new genomes. Mutation results in changing the value of genes (Sivanandam and Deepa, 2008). These changes occur randomly with a probability of mutation parameter set between [0, 1]. A random number in the same interval is generated for each gene in the new child. If this random number is less than the probability of mutation, the gene is assigned with a random number within the lower and upper bounds of the decision variable (Mirjalili *et al.*, 2020).

In Figure 2-6 is an illustration of before and after a mutation process where a new gene is introduced.

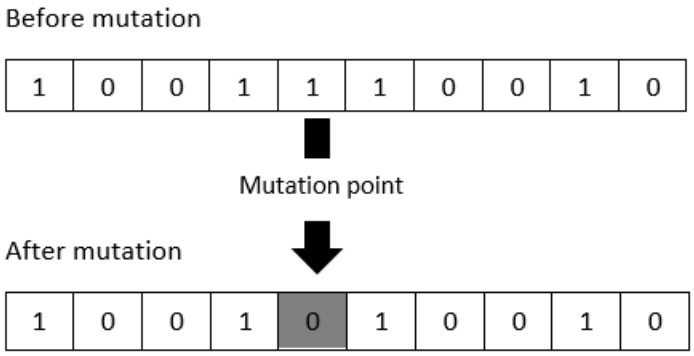

Figure 2-6: Sample mutation

2.9 Elitism

Elitism is a genetic operator applied to the chromosomes obtained after selection, crossover and mutation in some cases to preserve or copy the traits of the best chromosomes to the next generation (Ahn and Ramakrishna, 2003; Dumitrescu et al., 2000). This helps to keep a fit chromosome in each generation at all times by ensuring that these fit chromosomes are not lost during the iteration process. Elitism as a genetic operator was not part of the initial operators during the theoretical formulation of GA but has through the years proven to be efficient when applied as a genetic operator (Mirjalili *et al.*, 2020; Ahn and Ramakrishna, 2003).

2.10 Exploration and exploitation

Evolution algorithms go through exploration and exploitation to achieve their optimum solutions. Exploration is the process used by evolution algorithms to investigate the search space for solutions. Exploitation makes use of the explored solution by focusing on the fitter individuals in the solution space thereby reducing diversity. Search procedures like heuristic search solely depend on exploitation in attaining solutions in the search space while other search procedure like the random search uses exploration (Lin and Gen, 2009). GA is a general-purpose search that combines both exploration and exploitation. GA starts with an initial population or solution of individuals where the algorithm evolves. Exploration,

therefore, provides the mechanism of improving the diversity of the solution by continually probing different aspects of the search space through a mechanism like the mutation and crossover operator (Hansheng and Lishan, 1999). This procedure helps in the algorithm's effort to achieve a solution, but in isolation cannot adequately bring about an optimal solution. Exploitation is then introduced to focus on fitter individuals in the solution space. This approach however tends to reduce the diversity of the search space. In the quest to obtain an optimal solution for these algorithms, one cannot overemphasize the importance of attaining a good balance between exploitation and exploration. Lin and Gen (2009) in an attempt to develop a mechanism for maintaining the balance between exploration and exploitation, proposed an auto-tuning strategy based on fuzzy logic control (FLC). The general idea behind this approach is to tune the parameters in the GA based on the fitness of the parents in every generation. The auto-tuning is performed based on the change of the average fitness of the current and last generations. This maintains an appropriate balance between exploration and exploitation. An algorithm that emphasizes exploitation over exploration ultimately results in premature convergence, as the solution space is absent of diversity (Hansheng and Lishan, 1999). The genetic parameters available in GA helps to provide an important role in maintaining the balance between exploration and exploitation. Parameters such as crossover and mutation explore the search space and improve the diversity of the solution in the iteration process of the GA. This provides a structure for the GA to overcome premature convergence emanating from having a reduced diversity in the search space (Lin and Gen, 2009). In the aspect of selection and elitism, these genetic parameters are more focused on exploiting the solution to obtain fitter individuals in the population. The combination of these parameters maintains the delicate balance between exploration and exploitation. This, therefore provide a basis for GA in its efforts to solve combinatorial optimization problems.

2.11 Constraints handling

Evolution algorithms have different methods for handling constraints. These methods are generally problem dependent although some may be applied across different optimization problems. In the literature, the methods for handling constraints can be grouped into four different categories; the repair method, rejection method, penalty method and modification

of genetic parameters (Gen et al., 2008; Michalewicz, 1995b). Each method has merits that may suit a particular combinatorial optimization problem over the other.

In the repair method, population with infeasible chromosome is neither discarded nor penalized but rather a deterministic method for normalizing the infeasible chromosome is applied. This converts the infeasible chromosome to a feasible one (Michalewicz, 1995b). The method for normalizing or repairing the chromosome must consider the bounds of the constraints and create a chromosome that lies in the feasible region. This method is problem dependent and cannot be applied to any problem without first re-writing the repairing algorithm to suit the said problem. Two approaches of this method exists: (1) Always replacing the infeasible chromosome in the population with the repaired chromosome, and (2) using the repaired chromosome only for evaluation purposes without feeding it into the evolution. Both approaches are used in the literature. Nakano and Yamada (1991) used the always replacing approach and termed it as "forcing" where a feasible chromosome g' repaired from an infeasible chromosome g is forced to replace the chromosome g in the population.

The rejection method also termed as "death penalty" by Michalewicz (1995b) works by completely removing any infeasible chromosome from the population. In this approach, any infeasible chromosome in the population is discarded as opposed to being repaired. This approach has limitations. The initial population generated for a problem may have several infeasible solutions and per this method all these infeasible solutions need to be discarded which may lead the GA into premature convergence. Michalewicz (1995a) tested this method on five different cases and stated that it performed worse than the other constraints handling approaches. Outright rejection of infeasible solutions go against the nature of evolution algorithms (Richardson et al., 1989).

The penalty method is widely used and by far the easiest to implement. Constrained optimization problems are converted to unconstrained problems by applying a penalty function to the objective function of the problem. According to Dasgupta and Michalewicz (1997), the basic approach is to extend the objective function which in GA is represented as the fitness function of a Chromosome *i* in Equation (2.5);

$$fitness\ function = f(i) \pm Q(i) \tag{2.5}$$

Where $Q(i)$ represents the penalty for an infeasible Chromosome *i*, and $f(i)$ represents the objective function of the problem. For a maximization problem, the penalty function is expressed as $Q(i) < 0$ where *i* is infeasible, and $Q(i) = 0$ where *i* is feasible; whereas $Q(i) > 0$ where *i* is infeasible and $Q(i) = 0$ where *i* is feasible for a minimization problem. The general challenge with penalty functions as stated by Michalewicz (1995a) and Richardson *et al.* (1989) is knowing exactly what degree of penalty to apply to an infeasible chromosome or solution since all infeasible solutions are not alike. Figure 2-7 illustrates feasible and infeasible solution regions in a solution space. Assuming Solution *x* is the optimal solution without any prior knowledge, Solution *c* is closer to the optimal solution than Solution *b* and Solution *a*, although these solutions are in the infeasible region. Solution *c* may contain certain genes that may be relevant to attaining the optimal solution as opposed to Solution *b* and Solution *a*. Therefore, applying the same penalty value in this instance may not be ideal. Secondly, Solution *y* is farther than Solution *c* relative to the optimal Solution *x* although Solution *y* is in the feasible region. These complexities make finding the appropriate penalty value challenging.

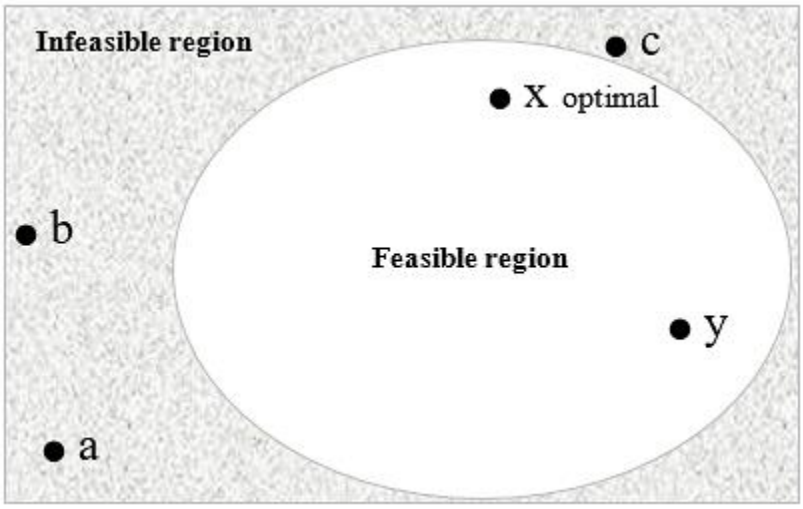

Figure 2-7: Feasible and infeasible solution space modified after Michalewicz (1995b)

The modification of genetic parameters approach works by creating problem specific methods that (1) represent the problem, and (2) modifies the conventional GA parameters

such as crossover and mutation to keep the optimization problem in the feasible domain (Gen et al., 2008). The approach always ensure the GA is kept only in the feasible search space at all times. Although this is desirable, it also limits the search space for the GA. In this research, both the repair and penalty method were implemented in handling constraints.

2.12 Termination

Generally, GA terminates when the maximum number of generations are reached. GA can also be terminated when a desired fitness value is met or when subsequent iteration does not improve the solution quality.

2.13 Multi-objective genetic algorithm

Optimization problems in the real world do not often have a single objective. There may be more than one aspect of the problem that needs to be optimized, resulting in a multi-objective optimization problem. For example, in the mining production scheduling problem, different objectives can be derived from the problem. The maximization of the NPV is one objective, waste management is another, and in oil sands, the minimization of dyke construction costs are all objectives that need to be optimized. The optimal solutions for multi-objective optimization problems are not always easy to find since there rarely exists a solution that simply maximizes or minimizes all the objectives of the problem compared to an optimization problem with a single objective. Equations (2.6) to (2.8) shows a general form of a single objective problem; where $f(x)$ is the objective function to be optimized, $g_i(x)$ is the constraint associated with the problem, and x is the decision variable. Equations (2.9) to (2.11) show a multi-objective problem where $f_{1...m}(x)$ represents the objectives of the problem, $g_i(x)$ are the constraints, and x is the decision variable. If there is no one solution that optimize all the objectives of a multi-objective problem without sacrificing portions of the other objectives, a set of possible solution is created and this set of solutions form what is known as the Pareto optimal solution or a non-dominated solution. Figure 2-8 demonstrates Pareto optimality. GA as an optimization algorithm is well suited for a multi-objective optimization problem. Deb *et al.* (2002) introduced the non-dominated sorting genetic algorithm II (NSGA-II) which is a robust approach for a multi-objective genetic algorithm. When a better solution with respect to all objective functions is found in the same

solution space, the solution is said to be dominated. Dominated solutions are depicted in Figure 2-8. The set of solutions to which there are no better solutions that satisfy both objectives are termed as non-dominated solutions or Pareto optimal solutions as shown in Figure 2-8.

$$Max\, z = f(x) \tag{2.6}$$

Subject to:

$$g_i(x) \le 0, \quad \forall i \in \{1, 2, ...N\} \tag{2.7}$$

$$x \ge 0 \tag{2.8}$$

$$Max[z_1 = f_1(x), z_2 = f_2(x), ...z_m = f_m(x)] \tag{2.9}$$

Subject to:

$$g_i(x) \le 0, \quad \forall i \in \{1, 2, ...N\} \tag{2.10}$$

$$x \ge 0 \tag{2.11}$$

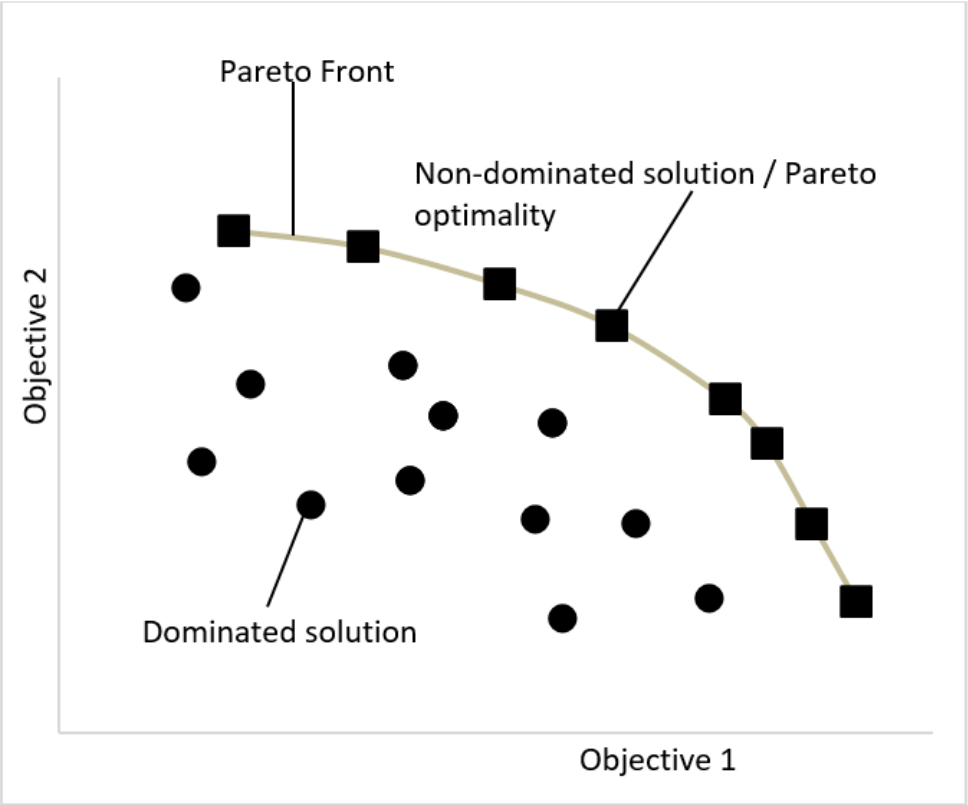

Figure 2-8: Concept of Pareto Optimality modified after Gen *et al.* (2008)

2.14 Open pit production scheduling

Open pit production scheduling (OPPS) problems can be defined as specifying the time and sequence in which blocks should be extracted from the mine in order to maximize the NPV subject to a variety of physical, environmental, operational and economic constraints (Caccetta and Hill, 2003). Production scheduling of an open pit mine is a major concern in mine planning and a complex optimization problem. Usually, the planning of an open pit mine starts with finding the ultimate digging or pit limit. This pit limit provides the list of blocks to be considered for production scheduling. The main algorithm used in the literature to find the ultimate pit limit is the LG algorithm (Lerchs and Grossman, 1965). Once the final pit is determined, the production scheduling process can commence. Researchers in their bid to solve the OPPS problem have formulated mathematical models with different optimization techniques (Askari-Nasab *et al.*, 2010; Caccetta and Hill, 2003; Koushavand *et al.*, 2014). These models take the form of an objective function for maximizing the NPV subject to the set constraints. There are two major research areas in the development of

production scheduling algorithms: (1) Deterministic algorithms and (2) Heuristics and Metaheuristic optimization algorithms.

2.14.1 Deterministic algorithms

Deterministic algorithms generate solutions based on the notion that the parameters of the optimization problem are known and have fixed real values from the start of the optimization. The solution to the optimization problem is fully determined by the parameter values and the initial conditions. These deterministic models have proven to be robust and capable of attaining near-optimal and optimal solution for relatively smaller optimization problems.

Johnson (1969) introduced Linear Programming (LP) for OPPS problems. The author did not obtain an optimal schedule for the problem due to the computational intractability. LP however proved to be a capable option for modelling the OPPS problem. Gershon (1983) and Dagdelen (1985) presented a MILP model which was an improvement of the LP model by Johnson. Formulating the model with MILP allowed for some of the decision variables to be presented as continuous variables to prevent infeasibility and allow for partial block processing. The model however could not obtain an optimum schedule for a real-size large scale OPPS problem. Caccetta and Hill (2003) proposed a MILP model solved with branch and cut algorithm for the OPPS problem. The authors used a cutting plane algorithm and a search strategy involving best first and depth first search to achieve a ‘‘good spread’’ of possible pit schedules. Their approach was capable of solving the OPPS problem on a small and medium scale optimization problem with 6,720 to 209,664 blocks. However, the optimization was computationally expensive. Due to commercialization and confidentiality agreements, the authors did not provide detailed information about their work. Dimitrakopoulos and Ramazan (2004) also proposed a MILP model for solving the OPPS problem. In their approach, they presented waste blocks as continuous variables in order to reduce the number of integer variables and improve the solution time.

Integer programming has also been studied and applied to the OPPS problem. Dagdelen and Johnson (1986) formulated the OPPS problem with integer programming and solved it using the Lagrangian relaxation method. Lagrangian relaxation is a method used to reduce the

complexity of the optimization problem by relaxing one or more constraints. A penalty term and a multiplier known as a Lagrangian multiplier used in the relaxed constraint is then added to the objective function. This is done to avoid violations of the relaxed constraint (Badiozamani and Askari-Nasab, 2010). In Dagdelen and Johnson (1986) formulation, the mining and processing constraints were relaxed and adjusted with Lagrangian multipliers to find the optimal solution. The authors decomposed the problem into smaller problems to allow for tractability of the solution. Ben-Awuah *et al.* (2018) implemented a MILP model that incorporates goal programming; a reward and penalty based approach to maximize the NPV. The authors used the clustering algorithm developed by Tabesh and Askari-Nasab (2011) to reduce the size of the optimization problem to ensure computational tractability. Their case study involved 16,985 blocks, and their model was able to find the optimal solution at 0% optimality gap.

2.14.2 Heuristics and metaheuristics

Heuristics are basic approximation algorithms that search the solution space to find a good solution and metaheuristics are algorithms that combine heuristics (that are usually very problem-specific) in a more general framework (Bianchi *et al.,* 2009). Metaheuristics are able to solve large optimization problems at a reasonable computational time. The difficulties associated with NP-hard class of problems and the general large instances of the problems make exact approaches that often generate optimal solution not ideal to solve such problems; taking into account the resources and computational time required. Researchers have investigated the trade-off between finding a good solution at smaller computational time and finding an optimal solution, which at times is intractable or resource intensive. The uncertainties associated with stochastic OPPS optimization problems make the application of metaheuristics more ideal in applying it to large problem instances, since it is capable of finding a good solution in a much smaller computation time. Popular metaheuristic algorithms for large scale optimization includes: tabu search, genetic algorithm, simulated annealing, ant colony optimization, and particle swarm optimization.

2.14.2.1 Tabu search

Tabu search (TS) is a metaheuristic algorithm used to solve large combinatorial optimization problems. The process of TS was designed by Glover (1986). It optimizes or improves a solution by searching through a neighbourhood of solutions and selecting the best ones. TS classifies certain solutions as forbidden (taboo; where the name ‘tabu’ is derived from) to prevent the algorithm from selecting those solutions which is a strategy to avoid cycling of the algorithm. There are three TS specific concepts that improves it solution approach over other combinatorial optimization algorithms according to Bianchi *et al.*, (2009). These concepts are: best improvement, tabu lists, and aspiration criteria. Best improvement is an approach in TS algorithm in which each existing solution is replaced with its best neighbouring solution. This is a method for avoiding local optima, however it may result in cycling when each current solution is replaced. TS counteracts this problem by creating a tabu list. Tabu list is a list that stores attributes of recently visited solutions. The type of attribute saved is the 'move' made by the algorithm in arriving at a result. The algorithm is then restricted from selecting from this set of solutions with attributes on the tabu list. The aspiration criterion is a criterion check in the TS algorithm that, if met, permits a 'move' to a banned solution to be chosen. Such criterion, according to Glover (1990), can be set as follows: if the existing solution is worse than the newer one, then the tabu can be overridden.

TS has been explored by researchers to solve the OPPS problem. Lamghari and Dimitrakopoulos (2012) used TS and Variable Neighborhood Descent (VND) algorithms for solving the OPPS problem. In their implementation, a definite number of neighborhood was set and the algorithm was made to search the neighborhood until the optimal solution was found. The authors implemented their model on a copper and gold dataset to validate the effectiveness of the proposed model. In their conclusion, the authors stated that a near optimal solution was found at a reasonable computational time. Senécal and Dimitrakopoulos (2020) presented a TS that uses multi-neighborhood to solve the long term OPPS problem. In their approach, the authors considered multiple processing streams under mineral uncertainty. The objective function from their model maximizes the discounted cash flow and penalizes deviations from production targets.

2.14.2.2 Genetic algorithm

Genetic algorithm (GA) has been researched extensively in combinatorial optimization in the area of Finance, Supply Chain Management, and Information Technology. However, not many research and applications of GA has been made in the area of stochastic open pit production scheduling to tackle all the related complexities. Alipour *et al.* (2017) presented a Genetic Algorithm (GA) approach for the OPPS problem. The researchers used an initial population of 20 with each population consisting of a 10 x 20 matrix in the GA model which represents the total blocks in the copper orebody. The initial population was then normalized to cater for the various constraints associated with the OPPS problem, namely; the mining capacity, processing capacity and block precedence constraints. A fitness function to evaluate the population was also formulated. The OPPS problem was solved with the GA and the results were compared to solution from IBM CPLEX solver. The authors indicated that, the computation time needed to solve the optimization problem with GA was significantly lower than that of the IBM CPLEX solver. However, the optimal solution from the GA was 5% lesser than that from the IBM CPLEX solver. The researchers further emphasized GA as computationally efficient but does not always give the optimal solution compared to LP and MILP with IBM CPLEX solver.

Another application of GA to the OPPS problem was presented by Alipour *et al.* (2020). In this application, the authors built on their earlier research in 2017. A 3D orebody model consisting of 10,529 blocks was used for a case study. The authors compared the GA solution to the solution from SimSched DBS software (Ota *et al.*, 2017) developed based on surface constrained stochastic life-of-mine production scheduling by Marinho de Almeida (2013). From the analysis by the authors, GA proved capable of solving not just a 2D OPPS problem but also a 3D OPPS problem. SimSched DBS obtained its solution in a shorter computation time than the GA. This was because SimSched DBS software mined blocks accumulated as surfaces which reduces the number of decision variables and level of selectivity for processing. The optimal solution from the GA was 4% better than that from the SimSched DBS software. The authors concluded at the time that, due to the size of the problem, any comparison to an exact optimization methodology was not possible. This further emphasizes the use of metaheuristics in attempting the OPPS problem. Grade uncertainties were not

considered in their research. The researchers through the case study demonstrated the viability of using a GA model to solve OPPS optimization problems.

Amponsah *et al.* (2021) also presented a GA model to solve a small-scale 3D OPPS problem. The researchers used a literal permutation encoding scheme from Gen *et al.* (2008) for chromosomes encoding. They compared their results to a MILP solution from CPLEX. In the authors' findings, the GA model's solution was within 10% of the MILP solution with CPLEX. This was partly because the MILP allowed for fractional block processing across multiple periods, which the GA did not. In extending Amponsah et al. (2021) research, a GA framework that allows for fractional block processing across multiple periods is presented in this research. The extended GA model, also considers grade uncertainties in its formulation and optimization.

2.14.2.3 Simulated annealing

"Simulated annealing (SA) algorithm is based on an analogy between the simulation of solid annealing and the problem of solving large combinatorial optimization problems," according to the researchers Van Laarhoven and Aarts (1987). Kirkpatrick *et al,* (1983) developed SA as a combinatorial optimization algorithm. SA optimization algorithm, in principle, is based on local search heuristics and employs a pre-defined neighbourhood search structure in the solution space. Kumral and Dowd (2005) proposed a SA algorithm approach to attempt the OPPS problem. The authors simulated the mineral deposit with Sequential Gaussian Simulation and averaged the simulated characteristics within specified block volumes to create the block model. The ultimate pit limit was then determined from the block model by pit-blend using LP and Lerchs-Grossman algorithm. The orebody model contained 2,773 blocks. SA provided a suitable and uniform mine production schedule in 30 minutes for this small scale problem.Goodfellow and Dimitrakopoulos (2013) also presented a SA optimization approach to the OPPS problem. Their model used a stochastic push-back design to adjust and minimize the deviation of materials sent to the waste and processing destinations. The authors implemented the SA formulation on a set of 20 block model simulations of a copper deposit. In the authors' numerical analysis, a total computation time of ~1.5 hours was taken to generate a solution. Aside the computational efficiency, the

authors also made a strong argument that the SA algorithm efficiently handled the multiple metals, slope zones and multiple destinations in the optimization problem.

2.14.2.4 Ant colony optimization

Ant colony optimization (ACO) is a population based metaheuristic algorithm. The concept that inspired ACO is based on the behaviour ants display when plying a route in search of food (Dorigo and Blum, 2005). Ants on their quest for food scan their nest in a random manner, and when a food source is found, the ant releases a chemical called pheromone on its way to the nest. This chemical serves as a way of communicating to other ants to ply the same route in search of food (Deneubourg *et al.*, 1990). The route with the highest pheromone concentration tends to be the preferred route or the shortest. Subsequently, the pheromone evaporates as time goes on. In combinatorial optimization problems, Dorigo *et al.* (1999) presented the ACO as an optimization algorithm for the Travelling Salesman Problem (TSP). An application of the ACO algorithm was proposed by Shishvan and Sattarvand (2015) for the OPPS problem. In their implementation, the authors used the Max-Min ant colony system and tested the model on a copper-gold deposit. The deposit consisted of 350,000 blocks. The ACO algorithm generated a 12% improvement in the initial mining schedule. The authors carried out a sensitivity analysis on the ACO parameters consisting of initial pheromone values, pheromone evaporation rate, and perturbation distance. In the author's findings, they stated that a higher initial pheromone value reduced the algorithms chances of exploring better solutions thereby generating poor results. In the analysis of the evaporation rate, the authors concluded that lower evaporation rate increases the time spent by the algorithm on poor solutions.

2.14.2.5 Particle swarm optimization

Particle swarm optimization (PSO) is a nature inspired metaheuristic optimization which was first proposed by Kennedy and Eberhart (1995). PSO algorithm performs the search process by using a population of individuals living in groups (Khan and Niemann-Delius, 2014). Khan and Niemann-Delius (2014) implemented the PSO algorithm for the OPPS problem. The authors used a continuous version of the PSO algorithm and a guaranteed convergence PSO algorithm. The authors' inspiration for this approach was the number of

blocks available in the ultimate pit limit of an open pit mine. Depending on the deposit size, this usually contain hundreds of thousands of blocks, therefore accounting for the authors' use of a continuous PSO to reduce the computational time. The precedence constraints in the OPPS problem were handled by normalization in the PSO algorithm. The algorithm checks and repair each solution to ensure feasibility at all times. The capacity constraints were handled by the application of a penalty method. The proposed model was implemented on two copper orebodies with 10,120 and 7,863 blocks. The model was successful in solving the OPPS problem with an optimality gap of 12.61% for the first case study and 4.80% for the second case study, respectively.

2.15 Stochastic open pit optimization in the presence of grade uncertainty

The conventional OPPS problem is optimized with a single interpolated orebody block model which does not account for grade uncertainties. As the conventional OPPS approach does not consider grade uncertainties, a true representation of the optimal NPV is rarely achieved. As reported by Sabour and Dimitrakopoulos (2011), due to uncertainties associated with mining projects, the mining industry in Canada lost in excess of 1.4 billion dollars in 1991. In the incorporation of uncertainties in the OPPS problem, the stochastic model takes several simulated orebody realizations as input with each orebody model having varying grades. The stochastic model then seeks to optimize for the maximum NPV and minimum waste management cost, while providing risk-based solution that tends to minimize deviations from operational targets.

Dimitrakopoulos and Ramazan (2008) introduced a stochastic integer programming (SIP) formulation that considered grade uncertainty. The authors elaborated that the SIP model considers multiple scenarios and generate a desirable outcome for a set of specified objectives which made its application to the OPPS problem preferable. The authors implemented their SIP model on two case studies; a gold deposit and a copper deposit. The case study with the gold deposit had 22,296 blocks. In the analysis of the results by the authors, the gold deposit case study was optimized in two stages with both optimizations totaling 42 hours in computational time with 14 simulated orebody realizations. The authors indicated that the SIP model with the simulated orebody realizations had a 9.7% increase in NPV over the traditional (conventional) mixed integer programming (MIP) model with a

single interpolated orebody block model. There was a similar outcome from the copper case study. The number of simulated orebody realizations for this case study was 20 and the authors recorded an increase in NPV of ~ 25% over the traditional MIP model's NPV. Mbadozie (2020) also presented a stochastic mixed integer linear programming (SMILP) formulation for oil sands production scheduling and waste management that considers grade uncertainty. The author used 20 orebody realizations to represent grade variability in the deposit. The results from the oil sands case study demonstrated that the SMILP schedule generated 16.85% improvements in NPV over the conventional schedule. These promising gains in NPV from stochastic production schedules form the basis of this research.

2.16 Summary and conclusions

Over many decades, researchers have formulated various mathematical models to solve the OPPS problem. Of these formulations and models, exact methodologies have been the dominant approach in the literature. These models achieve optimum results when applied to tractable problems. However, applying these models to large-scale and complex problems, particularly problems with many binary variables, may result in intractability or difficulty in attaining solutions in a reasonable runtime. In general, metaheuristics can achieve good, and sometimes optimal solutions to problem instances of reasonable size and in a shorter computation time (Bianchi *et al*., 2009). In this chapter, literature on these two optimization approaches were reviewed and discussed as they are applied to the OPPS problem. Genetic algorithm, which is a metaheuristic optimization algorithm, was further reviewed and its algorithmic formulation discussed in detail. GA has unique advantages in implementing it for the OPPS problem since it has a good balance between exploration and exploitation of the optimization search space.

CHAPTER 3

THEORETICAL FRAMEWORK

3.1 Background

This chapter elaborates on the theoretical framework proposed in this thesis for optimizing an open pit production scheduling problem with genetic algorithm. The conceptual framework and mathematical models are described in detail. The genetic algorithm representation of the problem is presented and theories on how to handle constraints are stated. The principal objective of this thesis is to formulate a genetic algorithm framework that optimizes both the conventional and stochastic open pit production scheduling problem to generate the maximum NPV. An integrated approach and formulation is therefore required to represent both the conventional and stochastic scenarios of the optimization problem.

The data used in the theoretical formulation of the problem is derived from the modeling of the oil sands data by Mbadozie (2020). The oil sands data used for the conventional formulation was modelled with Ordinary Kriging (Krige, 1951) estimation method. For the stochastic formulation, a set of equally probable orebody realizations were generated using Sequential Gaussian Simulation and the Geostatistical Software Library (Deutsch and Journel, 1998). As recommended by Albor and Dimitrakopoulos (2009), and Vallejo and Dimitrakopoulos (2019), a set of 20 orebody realizations were generated to capture the range of grade uncertainty for the oil sands data and used as input to the GA framework for stochastic optimization.

3.2 Model formulation for open pit production scheduling

The NPV of OPPS is based on the economic block value (EBV) of individual blocks in the orebody block model. The EBV of a block depends on its value and the costs incurred in mining and processing the block. The cost of mining a block is a function of the block's location in relation to how deep the block is from the surface and how far it is to its final destination. To calculate the NPV, the EBV is discounted since OPPS is undertaken over multiple periods. The discounted profit from block *n* is therefore given as the discounted

revenue generated from mining block *n* minus the discounted cost for extracting and processing block *n*. This is presented in Equation (3.1).

$$discounted\ profit_n^t = discounted\ revenue_n^t - discounted\ \cos t_n^t \quad \forall_n \in \{1,...N\}; \forall_t \in \{1,...T\}; \quad (3.1)$$

The notations used in the formulation have been classified as indices and sets, parameters and decision variables.

3.2.1 Indices and sets

$s \in \{1,....,S\}$	index for realizations
$n \in \{1,....,N\}$	index for blocks
$t \in \{1,....,T\}$	index for scheduling periods
$N = \{1,...,N\}$	set of all blocks in the model
$S = \{1,...,S\}$	set of all equally probable orebody realizations
$H_n(D)$	For each block, there is a set $H_n(D)$ defining the immediate predecessor blocks that must be extracted prior to extracting block *n* with safe slopes; where *D* is the total number of blocks in $H_n(D)$

3.2.2 Parameters

r	discount rate
o_n	ore tonnage in block *n*
$o_{n,s}$	ore tonnage in block *n* of realization *s*
w_n	waste tonnage in block *n*
$w_{n,s}$	waste tonnage in block *n* of realization *s*
dr	geological discount rate
v_n^t	revenue obtained by selling the final product within block *n* in period *t*, minus the extra discounted cost of mining all the material in block *n* as ore

$v_{n,s}^t$	revenue obtained by selling the final product within block n of realization s in period t, minus the extra discounted cost of mining all the material in block n as ore
q_n^t	cost of mining all the materials in block n in period t as waste
$q_{n,s}^t$	cost of mining all the materials in block n of realization s in period t as waste
Cl^t	lower bound of the mining capacity in period t
Cu^t	upper bound of the mining capacity in period t
Ql^t	lower bound of the processing capacity in period t
Qu^t	upper bound of the processing capacity in period t
g_n	average grade in ore portion of block n
$g_{n,s}$	average grade in ore portion of block n for realization s
$\underline{g}^t$	lower bound of the required average head grade in period t
$\overline{g}^t$	upper bound of the required average head grade in period t
pc_{o-}^t	penalty cost for lower ore tonnage target deviation in period t
pc_{o+}^t	penalty cost for upper ore tonnage target deviation in period *t*
pc_{g-}^t	penalty cost for lower grade target deviation in period *t*
pc_{g+}^t	penalty cost for upper grade target deviation in period *t*

3.2.3 Decision variables

$x_n^t \in [0,1]$	continuous variable representing the portion of block *n* to be extracted as ore and processed in period *t*
$y_n^t \in [0,1]$	continuous variable representing the portion of the block *n* to be mined in period *t*; fraction of *y* characterizes both ore and waste in the block

$b_n^t \in [0,1]$	b_n^t binary integer variable controlling the precedence of extraction of mining block *n;* b_n^t equal to one if extraction has started in period *t,* otherwise it is zero
$od_{s,+}^t \in [0,1]$	continuous variable representing the excess from the ore tonnage upper bound in period *t* for realization *s*
$od_{s,-}^t \in [0,1]$	continuous variable representing the shortage to the ore tonnage lower bound in period *t* for realization *s*
$gd_{s,+}^t \in [0,1]$	continuous variable representing the excess from the grade upper bound in period *t* for realization *s*
$gd_{s,-}^t \in [0,1]$	continuous variable representing the shortage to the grade lower bound in period *t* for realization *s*

3.3 Deterministic MILP formulation

In the conventional formulation of the OPPS, grade uncertainties are not considered and the main objective is to maximize the NPV of the mining operation subject to a set of constraints. The objective function and constraints are outlined in Equations (3.2) to (3.8). This MILP formulation is consistent with the research undertaken by Askari-Nasab et al. (2011).

3.3.1 Objective function

The objective function of the MILP model (Equation 3.2) is formulated to maximize the NPV of the mining operation. The objective function consists of two continuous decision variables for block n. The first decision variable x_n^t represents portion of block n to be processed in period t if it is ore. The decision variable y_n^t represents the portion of block n to be extracted in period t; fraction of y characterizes both ore and waste in the block. Using continuous decision variables allows for the fractional extraction of blocks in different periods.

$$Max \sum_{t=1}^{T} \sum_{n=1}^{N} \left(\frac{v_n^t \times x_n^t - q_n^t \times y_n^t}{(1+r)^t} \right) \tag{3.2}$$

Subject to:

3.3.2 Mining capacity constraints

Equations (3.3) and (3.4) define the mining capacity constraints for each period. Equation (3.3) defines maximum capacity for mining. This ensures that the total amount of material mined is less or equal to the stipulated capacity of mining equipment while Equation (3.4) defines the minimum capacity and controls the minimum amount of materials mined. These constraints are controlled by the continuous decision variable y_n^t and allows the mine planner to use different mining capacities in each period throughout the life-of-mine.

$$\sum_{n=1}^{N}(o_n + w_n) \times y_n^t \leq Cu^t \qquad \forall_t \in \{1,...T\}; \tag{3.3}$$

$$\sum_{n=1}^{N}(o_n + w_n) \times y_n^t \geq Cl^t \qquad \forall_t \in \{1,...T\}; \tag{3.4}$$

3.3.3 Processing capacity constraints

The processing capacity constraints aids the mine planner in ensuring a consistent feed throughout the mine life, resulting in a mine-to-mill operation that is well integrated. This is a soft constraint and depends on the availability of ore blocks. The processing objective may not be met in some periods depending on the orebody's ore grade distribution. In such circumstances, pre-stripping might be considered to ensure a consistent mill feed. This effectively forces the optimizer to mine waste in the early stages so that when ore production begins, the plant feed supply will be consistent and uniform. Equations (3.5) and (3.6) define the processing capacity of the mining operation. Equation (3.5) sets the upper bound and Equation (3.6) sets the lower bound for the amount of ore processed. These constraints are controlled by the continuous decision variable x_n^t and allows the mine planner to provide a uniform mill feed throughout the life-of-mine. In practice, the processing targets must be set with minimal periodic deviations to ensure maximum utilization of the mill.

$$\sum_{n=1}^{N}(o_n \times x_n^t) \leq Qu^t \quad \forall_t \in \{1,...T\}; \tag{3.5}$$

$$\sum_{n=1}^{N}(o_n \times x_n^t) \geq Ql^t \quad \forall_t \in \{1,...T\}; \tag{3.6}$$

3.3.4 Grade blending constraints

The goal of blending in production scheduling is to mine in such a way that the ore materials fulfil the processing plant's quality and quantity specifications. The grade blending constraints are essential constraints during production scheduling. These constraints ensure that an acceptable range of ore is sent to the mill at all times. Therefore, this grade range should be set between a lower and upper limit to facilitate blending of mill feed material. Equation (3.7) defines the upper limit of the ore grade and Equation (3.8) defines the lower limit of the ore grade to be sent to the mill. These constraints are controlled by the continuous decision variable x_n^t.

$$\sum_{n=1}^{N}(g_n - \bar{g}^t) \times (o_n \times x_n^t) \leq 0 \quad \forall_t \in \{1,...T\}; \tag{3.7}$$

$$\sum_{n=1}^{N}(g_n - \underline{g}^t) \times (o_n \times x_n^t) \geq 0 \quad \forall_t \in \{1,...T\}; \tag{3.8}$$

3.3.5 Block precedence constraints

Equations (3.9) to (3.11) enforce the block extraction precedence constraints. Binary integer decision variable, b_n^t, is used to control the precedence of block extraction. b_n^t is equal to one if the extraction of mining blocks has started by or in period *t*; otherwise it is zero. For each mining block *n,* Equation (3.9) check the set of immediate predecessor blocks in $H_n(D)$ that must be mined prior to mining block *n.* Equation (3.10) checks that extraction of mining block *n* can start only when the mining block has not been previously extracted. Equation (3.11) ensures that once extraction of block *n* starts, this block is available for extraction in subsequent periods.

$$b_n^t - \sum_{i=1}^{t} y_d^i \leq 0 \quad d \in H_n(D) \qquad \forall_n \in \{1,...N\}; \quad \forall_t \in \{1,...T\}; \tag{3.9}$$

$$\sum_{i=1}^{t} y_n^i - b_n^t \leq 0 \quad \forall_n \in \{1,...N\}; \quad \forall_t \in \{1,...T\}; \tag{3.10}$$

$$b_n^t - b_n^{t+1} \leq 0 \qquad \forall_n \in \{1,...N\}; \quad \forall_t \in \{1,...T-1\}; \tag{3.11}$$

3.3.6 Variables control constraints

Equation (3.12) ensures that the total ore material mined in any given scheduling period is less or equal to the sum of the ore, and waste materials mined in that period. Equations (3.13) and (3.14) ensures that the sum of the partials of block *n* extracted is at most one over all periods at the end of the mine life.

$$\sum_{n=1}^{N} (o_n \times x_n^t) \leq \sum_{n=1}^{N} ((o_n + w_n) y_n^t) \ \forall_t \in \{1,...T\}; \tag{3.12}$$

$$\sum_{t=1}^{T} y_n^t \leq 1 \qquad \forall_n \in \{1,...N\}; \tag{3.13}$$

$$\sum_{t=1}^{T} x_n^t \leq 1 \qquad \forall_n \in \{1,...N\}; \tag{3.14}$$

3.3.7 Non-negativity constraints

Non-negativity constraints monitor the decision variables to ensure they do not take negative values. Equation (3.15) defines the non-negativity of decision variables.

$$x_n^t, y_n^t, b_n^t \geq 0 \quad \forall_n \in \{1,...N\}; \ \forall_t \in \{1,...T\}; \tag{3.15}$$

3.4 Stochastic MILP formulation in the presence of grade uncertainty

The stochastic formulation for the OPPS problem considered in this research is modified after the formulation by Vallejo and Dimitrakopoulos (2019) and Mbadozie (2020). The approach for including grade uncertainty in the mining project stems from having multiple simulated orebody realizations generated through SGS which are equally probable and serve as input to the stochastic model. Equally probable orebody realizations mean each simulated realization can be a valid representation of the actual orebody. The simulated orebody

realizations capture the varying grade distribution that would not have been realized with a single interpolated block model based on a method like Kriging. Previous research from Albor and Dimitrakopoulos (2009), and Vallejo and Dimitrakopoulos (2019) have identified that, 20 simulated orebody realizations are adequate to capture the uncertainty in grade distributions.

3.4.1 Multi-objective function

The objective function for the stochastic model is derived from the average of all the simulated orebody realizations. Since these realizations are equally probable, each realization depicts varying grades for the orebody model. This can be assumed as having S number of schedules at the end of the optimization with each s schedule representing a probable solution. To simultaneously optimize with all the equally probable orebody realizations, an average of the revenue and cost from the realizations are taken into account in the objective function (Equation 3.16).

The multi-objective function has two components: 1) Maximize the NPV of the mining operation (Equation (3.16)); and 2) Minimize the cost of uncertainty associated with deviating from the operating targets, including ore tonnage and ore grade (Equation (3.17)). This is achieved by applying penalty costs and a geological risk discount rate to the ore tonnage and ore grade targets. Continuous deviation decision variables $od^t_{s,+}$, $od^t_{s,-}$, $gd^t_{s,+}$ and $gd^t_{s,-}$ as well as their respective penalty parameters pc^t_{o+}, pc^t_{o-}, pc^t_{g+} and pc^t_{g-} are used for minimizing deviations from ore tonnage and ore grade production targets defined by Equations (3.21) to (3.24). These are introduced in the second component of the objective function (Equation (3.17) to enable the optimizer to select realization blocks with ore tonnage and ore grade that minimizes deviations from the corresponding production targets simultaneously through a balancing act. For example, if the optimizer selects realization blocks with high grade, it will lead to a large ore tonnage deviation resulting from reduced ore reserve which is undesirable and vice versa.

Additionally, a geological risk discount rate (dr) is applied to the cost of deviation to defer the risk of not meeting production targets to later periods. From Equation (3.17), by apply the dr parameter as a denominator tied to periods, early periods have larger impact on the

minimization objective function value than later periods. This means the overall penalty value is higher in the earlier periods than in latter periods ensuring that early-year deviations from stated targets are lower than later-year deviations. Conceptually, the higher penalty in earlier periods drive the optimizer to limit deviations from the ore tonnage and ore grade targets early in the mine life and postpone extraction of areas with larger deviations until later periods when more geological understanding of the deposit becomes available.

$$Max \frac{1}{s} \sum_{s=1}^{S} \sum_{t=1}^{T} \sum_{n=1}^{N} \left(\frac{v_{n,s}^{t} \times x_{n}^{t} - q_{n,s}^{t} \times y_{n}^{t}}{(1+r)^{t}} \right) \tag{3.16}$$

Where S is set of all equally probable realizations.

$$Min \frac{1}{S} \sum_{s=1}^{S} \sum_{t=1}^{T} \left(\frac{pc_{o+}^{t} \times od_{s,+}^{t} + pc_{o-}^{t} \times od_{s,-}^{t}}{(1+dr)^{t}} \right) + Min \frac{1}{S} \sum_{s=1}^{S} \sum_{t=1}^{T} \left(\frac{pc_{g+}^{t} \times gd_{s,+}^{t} + pc_{g-}^{t} \times gd_{s,-}^{t}}{(1+dr)^{t}} \right) \tag{3.17}$$

Equations (3.16) and (3.17) can be combined together as a single objective function as shown in Equation (3.18).

$$Max \frac{1}{s} \sum_{s=1}^{S} \sum_{t=1}^{T} \sum_{n=1}^{N} \left(\begin{array}{l} \left(\frac{v_{n,s}^{t} \times x_{n}^{t} - q_{n,s}^{t} \times y_{n}^{t}}{(1+r)^{t}} \right) - \\ \frac{1}{N} \left(\frac{pc_{o+}^{t} \times od_{s,+}^{t} + pc_{o-}^{t} \times od_{s,-}^{t}}{(1+dr)^{t}} \right) - \\ \frac{1}{N} \left(\frac{pc_{g+}^{t} \times gd_{s,+}^{t} + pc_{g-}^{t} \times gd_{s,-}^{t}}{(1+dr)^{t}} \right) \end{array} \right) \tag{3.18}$$

Subject to:

3.4.2 Mining capacity constraints

Equations (3.19) and (3.20) defines the mining capacity constraint for each period. Equation (3.19) ensures that the total blocks tonnage mined is equal to or less than the stipulated capacity of mining equipment while Equation (3.20) controls the minimum amount of materials mined. The tonnage of materials mined is the sum of the ore tonnage and waste

tonnage represented as $o_{n,s}$ and $w_{n,s}$ respectively. The continuous decision variable y_n^t controls this extraction process in each period.

$$\sum_{n=1}^{N}(o_{n,s}+w_{n,s})y_n^t \le Cu^t \quad \forall_t \in \{1,...T\};\ \forall_s \in \{1,...S\}; \tag{3.19}$$

$$\sum_{n=1}^{N}(o_{n,s}+w_{n,s})y_n^t \ge Cl^t \quad \forall_t \in \{1,...T\};\ \forall_s \in \{1,...S\}; \tag{3.20}$$

3.4.3 Processing capacity constraints

Equations (3.21) and (3.22) define the processing capacity of the mining operation for each period. Equation (3.21) sets the upper bound and Equation (3.22) sets the lower bound for the amount of ore processed. The deviation decision variables $od_{s,+}^t$ and $od_{s,-}^t$ are introduced to serve as buffers to the ore tonnage targets. These decision variables are penalized in the objective function (Equation (3.18)) to ensure that the ore tonnage targets are achieved with minimum deviation. These constraints are controlled by the continuous decision variables x_n^t, $od_{s,+}^t$ and $od_{s,-}^t$ in each period.

$$\sum_{n=1}^{N}(o_{n,s}\times x_n^t)-od_{s,+}^t \le Qu^t \qquad \forall_t \in \{1,...T\};\ \forall_s \in \{1,...S\}; \tag{3.21}$$

$$\sum_{n=1}^{N}(o_{n,s}\times x_n^t)+od_{s,-}^t \ge Ql^t \qquad \forall_t \in \{1,...T\};\ \forall_s \in \{1,...S\}; \tag{3.22}$$

3.4.4 Grade blending constraints

Equation (3.23) defines the upper limit of the ore grade and Equation (3.24) defines the lower limit of the ore grade to be sent to the mill in each period. The deviation decision variables $gd_{s,+}^t$ and $gd_{s,-}^t$ are introduced to serve as buffers to the ore grade targets. These decision variables are penalized in the objective function (Equation (3.18)) to ensure that the ore grade targets are achieved with minimum deviation. These constraints are controlled by the continuous decision variables x_n^t, $gd_{s,+}^t$ and $gd_{s,-}^t$ in each period.

$$\sum_{n=1}^{N} g_{n,s} \times (o_{n,s} \times x_n^t) - \sum_{n=1}^{N} \overline{g}^t \times (o_{n,s} \times x_n^t) - gd_{s,+}^t \leq 0 \quad \forall_t \in \{1,...T\}; \forall_s \in \{1,...S\}; \tag{3.23}$$

$$\sum_{n=1}^{N} g_{n,s} \times (o_{n,s} \times x_n^t) - \sum_{n=1}^{N} \underline{g}^t \times (o_{n,s} \times x_n^t) + gd_{s,-}^t \geq 0 \quad \forall_t \in \{1,...T\}; \forall_s \in \{1,...S\}; \tag{3.24}$$

3.4.5 Block precedence constraints

Equations (3.25) to (3.27) enforce the block extraction precedence constraints. Binary integer decision variable, b_n^t, is used to control the precedence of block extraction. b_n^t is equal to one if the extraction of mining blocks has started by or in period *t*; otherwise it is zero. For each mining block *n*, Equation (3.25) check the set of immediate predecessor blocks in $H_n(D)$ that must be mined prior to mining block *n*. Equation (3.26) checks that extraction of mining block *n* can start only when the mining block has not been previously extracted. Equation (3.27) ensures that once extraction of block *n* starts, this block is available for extraction in subsequent periods.

$$b_n^t - \sum_{i=1}^{t} y_d^i \leq 0 \;\; d \in H_n(D) \; \forall_n \in \{1,...N\}; \;\; \forall_t \in \{1,...T\}; \tag{3.25}$$

$$\sum_{i=1}^{t} y_n^i - b_n^t \leq 0 \;\; \forall_n \in \{1,...N\}; \;\; \forall_t \in \{1,...T\}; \tag{3.26}$$

$$b_n^t - b_n^{t+1} \leq 0 \qquad \forall_n \in \{1,...N\}; \;\; \forall_t \in \{1,...T-1\}; \tag{3.27}$$

3.4.6 Variables control constraints

Equation (3.28) ensures that the total ore material mined in any given scheduling period is less or equal to the sum of the ore, and waste materials mined for all realizations in that period. Equations (3.29) and (3.30) ensure that the sum of the partials of block *n* extracted is at most one over all periods at the end of the mine life.

$$\sum_{n=1}^{N} (o_{n,s} \times x_n^t) \leq \sum_{n=1}^{N} ((o_{n,s} + w_{n,s}) y_n^t) \; \forall_t \in \{1,...T\}; \; \forall_s \in \{1,...S\}; \tag{3.28}$$

$$\sum_{t=1}^{T} y_n^t \leq 1 \qquad \forall_n \in \{1,...N\}; \tag{3.29}$$

$$\sum_{t=1}^{T} x_n^t \leq 1 \qquad \forall_n \in \{1,...N\}; \tag{3.30}$$

3.4.7 Non-negativity constraints

Equation (3.31) defines the non-negativity constraints for the decision variables for mining, processing, extraction precedence, and ore tonnage and ore grade target deviations. These constraints enforce that none of these variables can take on negative values during the optimization process.

$$x_n^t, y_n^t, b_n^t, od_{s+}^t, od_{s-}^t, gd_{s+}^t, gd_{s-}^t \geq 0 \qquad \forall_n \in \{1,...N\}; \quad \forall_t \in \{1,...T\}; \forall_s \in \{1,...S\}; \tag{3.31}$$

3.5 Genetic algorithm problem representation

The starting point of the optimization problem in GA is the problem initialization which consists of the chromosome encoding phase. A multi-layer chromosome encoding technique was used in this research: (1) a literal permutation encoding scheme, and (2) a real number or continuous variable encoding scheme. Figure 3-1 shows a sample of the chromosome encoding represented in the GA. The literal permutation encoding was employed for the genes representing the period whiles real number encoding was used for the genes representing the fractions blocks extracted.

Block index	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*14*	*15*	~	*n*
Period	5	6	5	4	3	1	1	2	2	6	7	8	7	8	8	~	8
% Block	0.3	0.2	0.5	0.2	0.1	0.3	0.6	0.9	0.4	0.22	0.23	0.41	0.74	0.25	0.33	~	0.2

Figure 3-1: Sample chromosome encoding

In this research, every block is assumed to be mined over at most two periods. Therefore, the chromosomes were encoded in two halves as shown in Figure 3-2. The first and second halves represent the fractions of each block mined at different periods respectively. The constraints in Equations (3.29) and (3.30) are therefore satisfied in the chromosome

encoding when these two halves are reconciled at the end of the optimization. Every block in the model is mined at most once during the mine life.

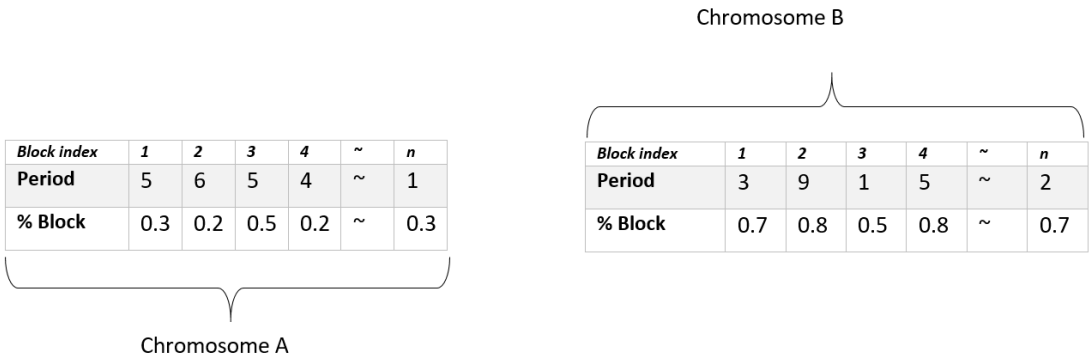

Figure 3-2: Sample chromosome encoded in two halves

Equation (2.3) was used in generating the genes in the chromosome representing the block fractions since this requires a continuous variable encoding. The other genes were generated using a Gaussian random distribution. The GA was implemented to optimize either a deterministic or stochastic production schedule based on input from the user. The objective function of the optimization was represented as a fitness function to test each solution in the population. The fitter population survives the current generation and proceeds in the iteration process.

3.5.1 Constraints handling and representation

The major constraint in the OPPS problem that presents great levels of complexities is the precedence constraint. The precedence constraint determines the sequence of block extraction and ensures that blocks on the surface are extracted in the same or an earlier period to blocks directly beneath them. As shown in Figure 3-3 , Blocks 1, 2 and 3 must be extracted prior to extraction of Block 10 or in the same period as Block 10. In three-dimensional (3D) block representation, every block has at least nine different blocks forming its precedence.

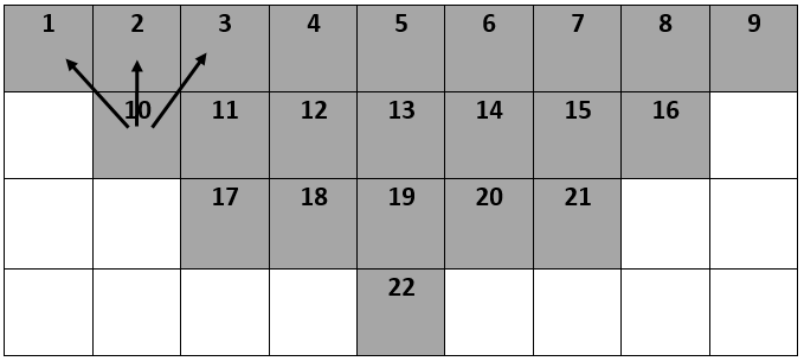

Figure 3-3: Block extraction precedence modified after Ben-Awuah and Askari-Nasab (2011)

In order to ensure that the precedence constraints are enforced as defined, a check-and-repair method is implemented in the GA. In each population, if a block cannot be mined in period *t* due to precedence constraints, it is moved to the next period or a period where the requirements of immediate predecessor blocks in $H_n(D)$ are not in violation. The entire population is then normalized to accept the current gene as a feasible solution for evaluation. For every violation of the normalized precedence constraints, the fitness function is penalized to ensure that the optimizer finds a feasible solution in each generation.

Capacity constraints are treated as knapsack problems. Knapsack problems are primarily resource allocation problems. The maximum allowable capacity is derived from the optimization problem since the scheduling is performed annually (periods). All the extracted blocks for that period should be less or equal to the maximum capacity for that period. Using sliding window technique (Cullenbine et al., 2011; Anirudh Sharma, 2020), an array containing the tonnage of every block scheduled in each period is created. The total tonnage of every period is checked against the maximum capacity from the optimization problem. When the maximum capacity for a period is reached, all subsequent blocks or block fractions that were originally in that period are moved to the next period. The population is then normalized afterwards so that the current population contains the right blocks that satisfy the capacity constraint for that period. The window is then slid to the next period and the steps above are repeated until the last period is reached.

3.5.2 Normalization

Due to the randomness associated with GA at every stage of the optimization process, the genes in each population tend to violate constraints when mutation or crossover occurs. There is therefore the need to normalize the population after each mutation and crossover to ensure that the constraints are satisfied (Denby and Schofield, 1994). This process is termed as normalization or regularization. Figure 3-4 shows an example of a double point crossover that violates the precedence constraints. As illustrated in Figure 3-4(A), before crossover occurs, both Parents 1 and 2 are feasible solutions satisfying the precedence constraints. That is, all blocks on the lower level are extracted in periods later than or equal to extraction periods of blocks above them. During crossover, the genes in Parents 1 and 2 representing Blocks 10, 11, 12 and 13 within the crossover point are swapped. Child 1 receives genes from Parent 2 whiles Child 2 receives genes from Parent 1. As highlighted in Figure 3-4(B), after the crossover, both Child 1 and Child 2 violate the precedence constraint because Block 10 in Child 1 and Block 18 in Child 2 are extracted in periods earlier than the blocks above them and therefore require normalization. This process ensures that a population with a feasible solution is kept at all times throughout the GA optimization process.

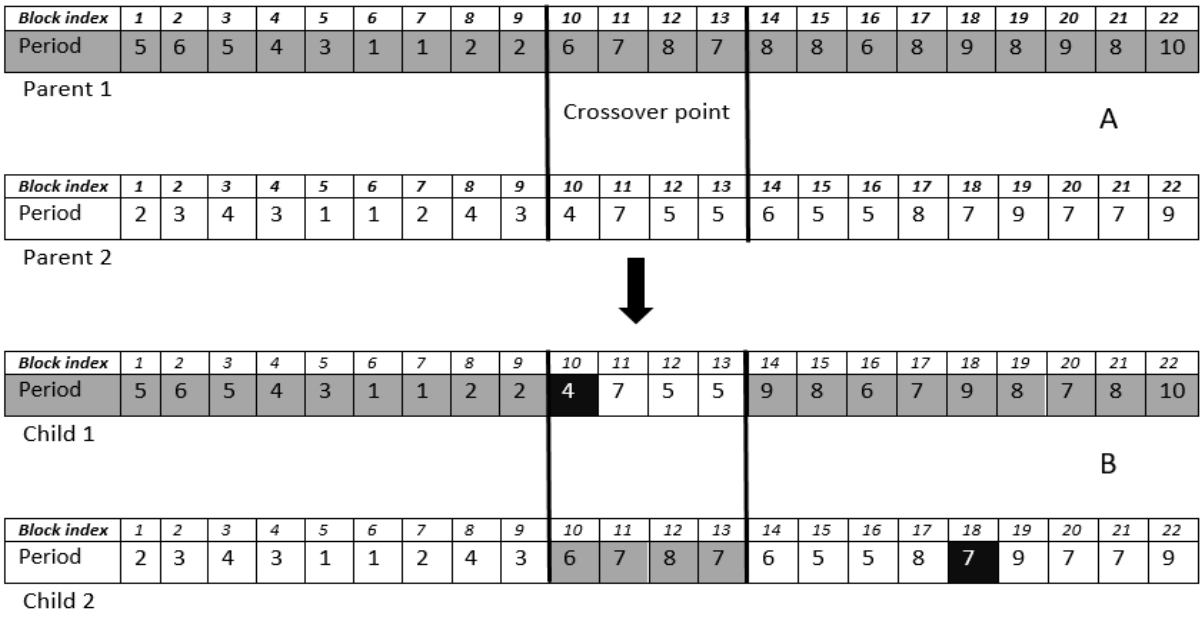

Figure 3-4: Sample crossover showing precedence constraint violation. (A) Illustrates a feasible solution before crossover. (B) Demonstrates a solution that violates the precedence constraint after crossover

3.5.3 Mutation and crossover strategy

Genetic operators such as crossover and mutation are key to the success of any genetic algorithm optimization process and as such finding the best strategy for them is always paramount. In this research, a 'smart' mutation was implemented to curtail the complexities in handling the partial extraction of blocks in the population. Figure 3-5 shows a sample chromosome with twenty genes. To represent a chromosome with n number of blocks or genes; the corresponding length of that chromosome is 2n. This method is used to implement the assumption that each block can be extracted in at most two periods. The first part of the chromosome represented as Chromosome A in Figure 3-5 shows portions of the blocks and corresponding periods the extraction occurs in; same for Chromosome B.

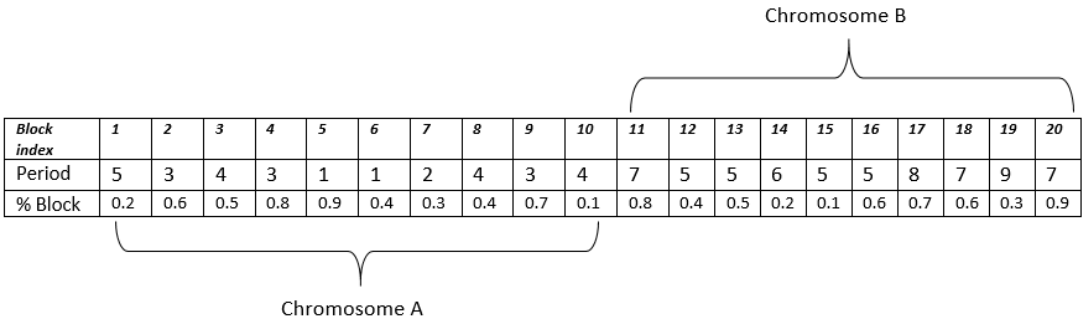

Figure 3-5: Sample chromosome representing the multi-part chromosome encoding

During the mutation process, a probability of mutation is applied to determine the gene that must mutate. In Figure 3-6, genes at block index 6, 7, 8 and 9 of Chromosome A will mutate per the probability of mutation applied. This however needs to occur in tandem with the corresponding genes in Chromosome B. The mutation algorithm keeps the index of the genes in Chromosome A and determines the corresponding position of the other genes in Chromosome B. When the mutation occurs in Chromosome A, a subsequent mutation and normalization takes place in Chromosome B. The genes representing the periods are mutated at random from a feasible set of periods that the block can be extracted in. Based on the precedence constraints. Given a feasible set of period (3, 2, and 4) for Block 6, a period is chosen at random and assigned to the block during the mutation for the period of that block. The mutation for the fractions of blocks that should be extracted is determined by Equation (2.3). The mutation algorithm again keeps the index of the gene representing the fraction of

the block to be extracted in Chromosome A and determines the corresponding position of the other gene in Chromosome B. The double point crossover used in this research also employs the same chromosome representation and index retention approach. Figure 3-7 shows a sample chromosome after mutation showing the result of mutation for Chromosome A and Chromosome B. Table 3-1 shows the pseudo code for the proposed mutation strategy used to handle the block extraction by the GA.

Figure 3-8 shows the flow chart for the GA optimization process and sub processes. A two-step GA framework was implemented; where based on the data provided and the input from the user, the framework will decide whether the problem is stochastic or conventional before proceeding to evaluate the fitness function for each population.

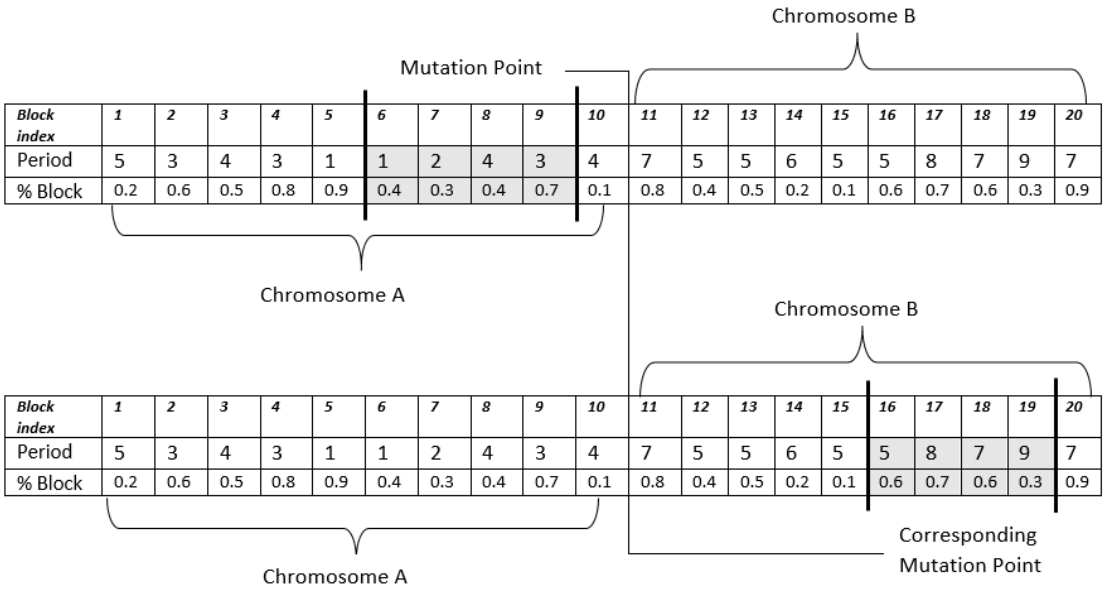

Figure 3-6: Sample chromosome before mutation showing the mutation point of Chromosome A and Chromosome B

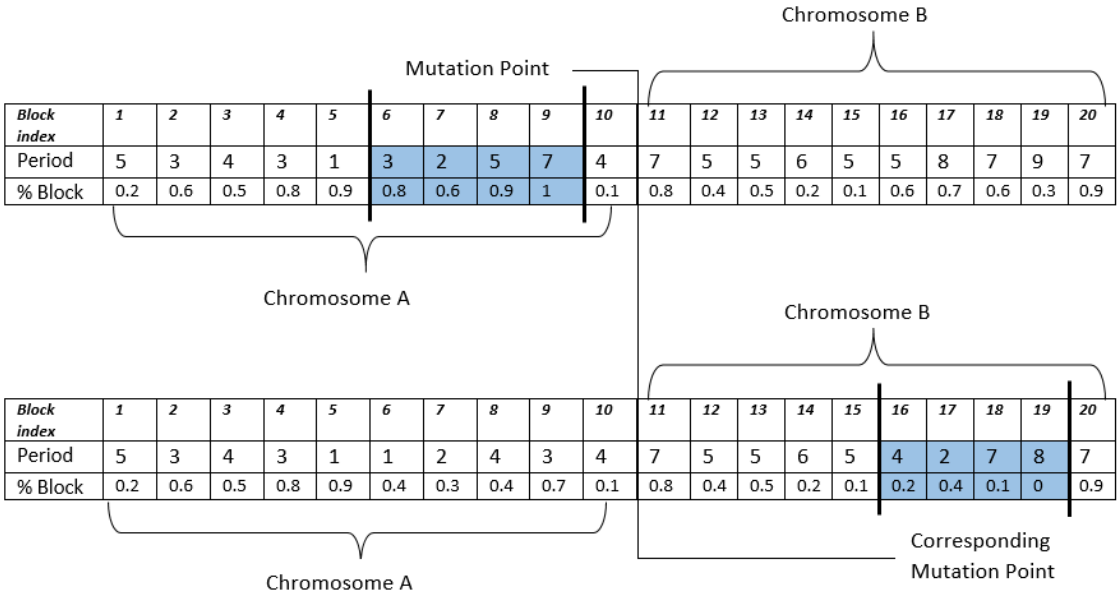

Figure 3-7: Sample chromosome after mutation showing the result of mutation for Chromosome A and Chromosome B

Table 3-1: Pseudo code for the proposed mutation strategy

Pseudo Code for proposed mutation strategy
Start ***get*** *chromosomeLength;* ***get*** *popabilityOfMutation;* ***get*** *numberOfBlocks* *Select number of individual to be mutated based on the propablilityOfMutation.* *split the chromosome into two halves A and B based on numberOfBlocks* ***While*** *n = Number of individuals to be mutated* *Get the index a of the individual in chromosome A and corresponding index b in chromosome B* *Perform mutation on gene n at a in chromosome A and gene n at b in chromosome B* ***EndWhile*** *Combine chromosome A and B after mutation and return to the main generation* ***End***

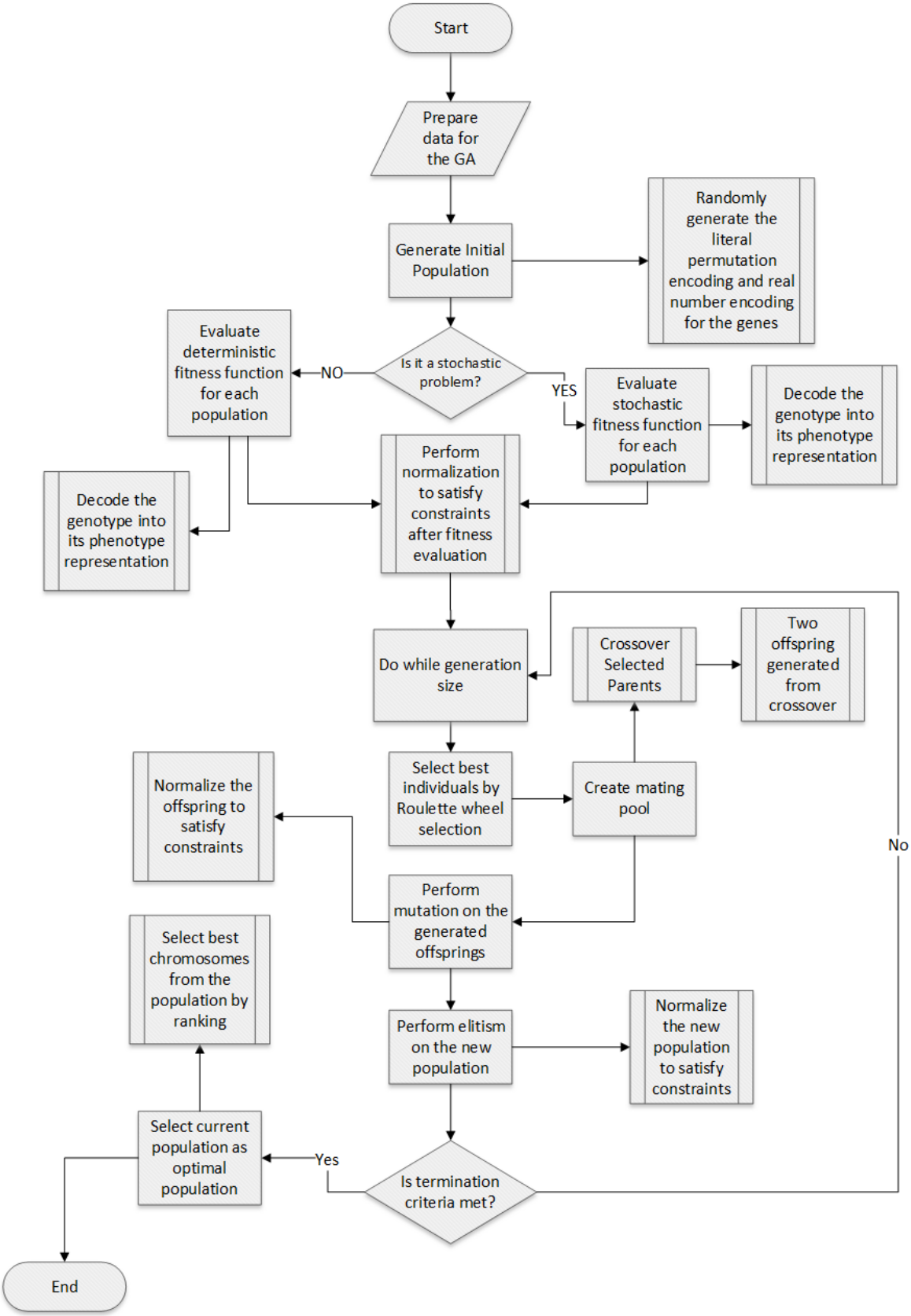

Figure 3-8: Proposed GA optimization process and sub processes

3.6 Summary

In this chapter, the theoretical modelling for the open pit production scheduling problem and the GA framework were presented. In this research, the deterministic model formulation and stochastic formulation in the presence of grade uncertainty were outlined. This chapter also presented the relationship between the theoretical MILP formulation of the problem and the GA representation. Theoretical concepts on how to handle soft and hard constraints in the problem with the GA framework were discussed and a proposed mutation and crossover strategy was adopted. These representations and theoretical adoptions were made to adequately define and model the OPPS problem. This will ensure the research objectives are achieved, thus maximising the NPV and producing a practical and uniform production schedule.

CHAPTER 4

APPLICATION OF GA FRAMEWORK AND DISCUSSION OF RESULTS

4.1 Background

Chapter 4 presents the application of the GA framework and discussion of results. The theoretical formulation and methodology presented in the previous chapter were implemented for oil sands deposit dataset. Experimental analysis and verification of the model are explored and discussed in this chapter. Two oil sands case studies are used as the basis to test and verify the formulated GA framework in comparison with the MILP model.

4.2 Verification of the GA framework

As stated by Bianchi et al. (2009), metaheuristics are capable of finding good, and sometimes optimal, solutions to problem instances of realistic size in a generally smaller computation time. It is paramount, however, to verify the results generated by these algorithms since they are not based on exact methodologies. The premise for verifying the results from the GA framework in this research is the objective function and the economic and technical constraints outlined in Chapter 3. A good model is not one that just maximises the objective function, but one that respects all the set constraints. A comparative analysis was conducted on the results from the GA with results from an implementation of the same case study using the MILP model with CPLEX. As observed from the literature, MILP models guarantee optimal solutions to tractable problems and, as such, form a good basis for validating the results generated by the GA framework. Case Study 1 was explicitly compared with a similar implementation of the MILP and SMILP models solved with CPLEX by Mbadozie (2020). This was because the MILP and SMILP models were tractable on Case Study 1 and therefore provided a means by which the results from the GA model could be compared. For Case Study 2, however, the SMILP integer solution was intractable with CPLEX. Therefore, CPLEX was used to solve the relaxed LP to estimate the optimization gap for the GA results.

Equation (4.1) by IBM ILOG CPLEX Inc (2017) was used to ascertain the optimal difference between the GA solution and the relaxed LP solution by CPLEX.

$$\frac{|bestbound - bestinteger|}{(1e-10+|bestinteger|)} \tag{4.1}$$

The bestbound in Equation (4.1) for an optimization problem refers to the objective function value at which a feasible optimal solution could potentially exist (IBM ILOG CPLEX Inc., 2017). In the case of an intractable integer problem, the bestbound is the optimal solution of the relaxed LP. This is the case because the relaxed problem does not have a bestinteger solution. Therefore, in determining the gap for the GA solution using Equation (4.1), the relaxed objective function value is represented as the bestbound and the solution for the GA as the bestinteger to compute the optimality gap.

4.3 Case study descriptions and scenarios

The GA model was implemented for two different case studies; the first case study with 4476 blocks and the second with 1569 blocks. Two scenarios were implemented for both case studies: (1) A deterministic model with GA (DGA), and (2) A stochastic model with GA (SGA). The orebody model for the DGA was based on Ordinary Kriging (Krige, 1951) which did not consider grade uncertainty. The SGA scenario considered grade uncertainty through equally probable orebody realizations generated using Sequential Gaussian Simulation for orebody modeling. The oil sands dataset used in this research was obtained from (Ben-Awuah and Askari-Nasab, 2011). Table 4-1 shows the block model data for both case studies and Table 4-2 shows the economic parameters for both case studies. The mining and processing requirements for both case studies are show in Table 4-3. In Table 4-4, the risk parameters associated with the stochastic scenario of the case studies are outlined. Table 4-5 and Table 4-6 show the number of decision variables and the size of constraint matrix for Case Study 1 and Case Study 2 respectively based on the MILP model. The problem size and number of decision variables for the GA model for Case Study 1 and Case Study 2 are outlined in Table 4-7 and Table 4-8 respectively. The production schedule for Case Study 1 was scheduled over ten periods whereas Case Study 2 was developed over twenty periods.

The primary focus for the GA was to generate a uniform and practical schedule while respecting the constraints for all periods in the schedule. The DGA and SGA were implemented in a MATLAB environment (MathWorks Inc., 2020) on a Lenovo ThinkPad computer with Intel(R) Core(TM) i7-8565U CPU @ 1.80GHz and 16 GB of RAM. Figure 4-1 presents the methodology for verifying the solution generated by the GA on Case Study 1 by comparing it with the implementation from MILP and SMILP models with CPLEX.

Table 4-1: Block model data from oil sands datasets for case studies

Block model data	**Case Study 1**		**Case Study 2**	
Parameter (Unit)	**(DGA) Value**	**(SGA) Value**	**(DGA) Value**	**(SGA) Value**
Total block tonnage (Mt)	318	318	3539	3539
Total ore tonnages (Mt)	126.56	128	1041	1141
Block dimensions (m x m x m)	50 x 50 x 15	50 x 50 x 15	300 x 300 x 15	
Mine life (Years)	10		20	
Number of blocks	4476		1569	

Table 4-2: Economic parameters for case studies

Economic parameters	
Parameter (Unit)	**Value**
Mining cost ($/tonne)	4.60
Processing cost ($/tonne)	5.03
Selling price ($/bitumen %mass)	4.50
Economic discount rate (%)	10

Table 4-3: Mining and processing requirements for case studies

	Case Study 1		**Case Study 2**	
Parameters (Unit)	**Min value**	**Max value**	**Min value**	**Max value**
Mining capacity (Mt/year)	25	32	100	150
Processing capacity (Mt/year)	10	14	25	50

Ore bitumen grade (%m)	7	16	7	16

Table 4-4: Risk parameters for stochastic scenario

Parameters (Unit)	Value
Number of realizations	20
Cost of shortage in ore production ($/tonne)	5
Cost of excess in ore production ($/tonne)	10
Cost of shortage in ore bitumen grade ($/%m)	2.5
Cost of excess in ore bitumen grade ($/%m)	1.5

Table 4-5: Problem size for Case Study 1 based on MILP model

Description	Value
Number of decision variables	671400
Number of integer variables	44760
Number of continuous variables	626640
Size of constraints matrix	484624 by 671400

Table 4-6: Problem size for Case Study 2 based on MILP model

Description	Value
Number of decision variables	472300
Number of integer variables	31380
Number of continuous variables	440920
Size of constraints matrix	350256 by 472300

Table 4-7: Problem size for Case Study 1 based on the GA model

Description	Value
Number of decision variables	26856
Number of integer variables	8952
Number of continuous variables	17904

Table 4-8: Problem size for Case Study 2 based on the GA model

Description	Value
Number of decision variables	9576
Number of integer variables	3192
Number of continuous variables	6384

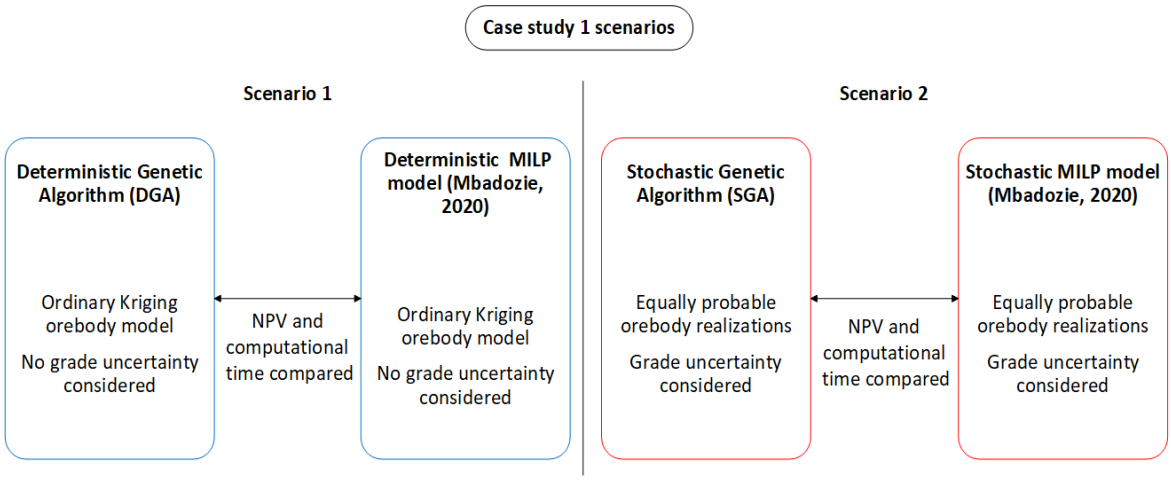

Figure 4-1: Case Study 1 scenarios comparisons

4.4 Case Study 1 implementation

In order for a production schedule to be carried out, the mineral resources have to be modelled into a three-dimensional block-by-block representation to serve as input to an

optimization algorithm. Each block in this three-dimensional block model contains the various attributes of the mineral present in the orebody model. These minerals can be classified as ore or waste based on the present economics. The block model for this case study consisted of 4476 blocks from an oil sands dataset. A matrix that consisted of the size $m \times n$ was created. Where m denoted the population size and n denoted the number of genes in each population, the genes represented the number of blocks in the block model. Each n gene in the population had three main attributes that defined it; (1) the index of the gene representing the block number in the block model; (2) the portion of the gene to be extracted either in part or completely and (3) the period the gene or block will be extracted in. The GA input parameters used for this case study are shown in Table 4-9.

Table 4-9: Proposed GA input parameters

GA parameter	Description
Population size	20
Selection type	Roulette wheel
Crossover type	Double point
Probability of mutation	0.2
Probability of crossover	0.85
Probability of elitism	0.2
Maximum generations	1000

4.4.1 Scenario 1 results and discussion: DGA

The scenario presented here is based on the DGA. The orebody model was based on Ordinary Kriging estimation. There was no consideration of grade uncertainty. This scenario was used as the base to test the proposed DGA. The production schedule was limited by the processing and mining requirement in Table 4-3. The focus of this experiment was to test the effectiveness of the DGA model on the OPPS problem, to optimize the NPV, achieve a uniform and practical schedule as well as respecting the mining and processing requirements. The production schedule results generated by the DGA are shown in Table 4-10. Figure 4-2

and Figure 4-3 shows the cross sectional view and the plan view of the extraction sequence generated by the DGA for the production schedule respectively. It can be seen from Figure 4-2 that, the DGA model enforced the precedence constraints set in the optimization problem; blocks were mined according to their precedence and scheduled appropriately. Blocks on the lower levels were mined in later periods as opposed to blocks on the surface.

Table 4-10: Scheduling results for the DGA

Period	Total tonnage (Mt)	Ore tonnage (Mt)	Average ore bitumen grade (%m)
1	24.41	8.51	12.87
2	31.93	13.32	12.32
3	31.94	13.54	12.00
4	31.90	13.96	11.31
5	31.93	13.91	11.33
6	30.91	13.91	10.54
7	31.93	13.90	10.81
8	31.90	13.94	9.92
9	31.27	13.93	8.76
10	9.13	6.70	7.92

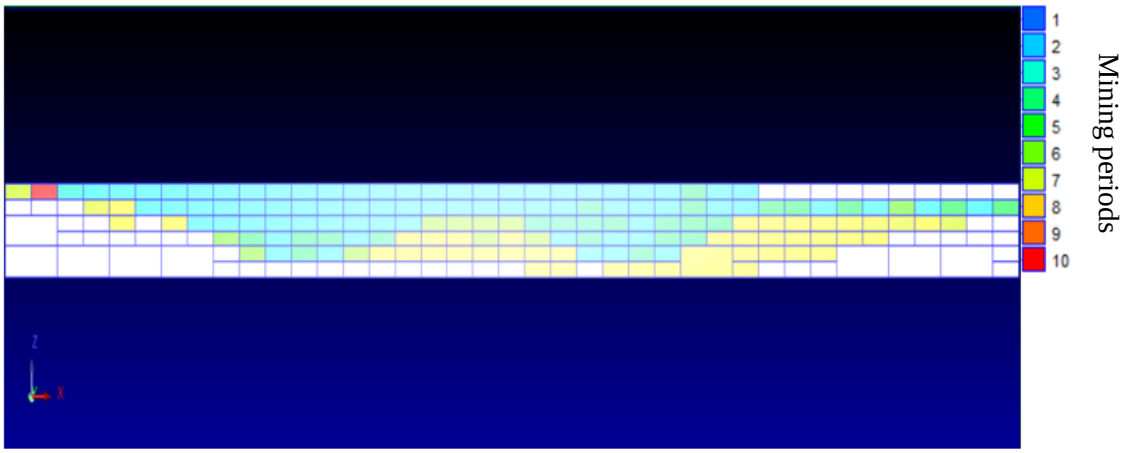

Figure 4-2: Cross sectional view of the block extraction sequence by the DGA

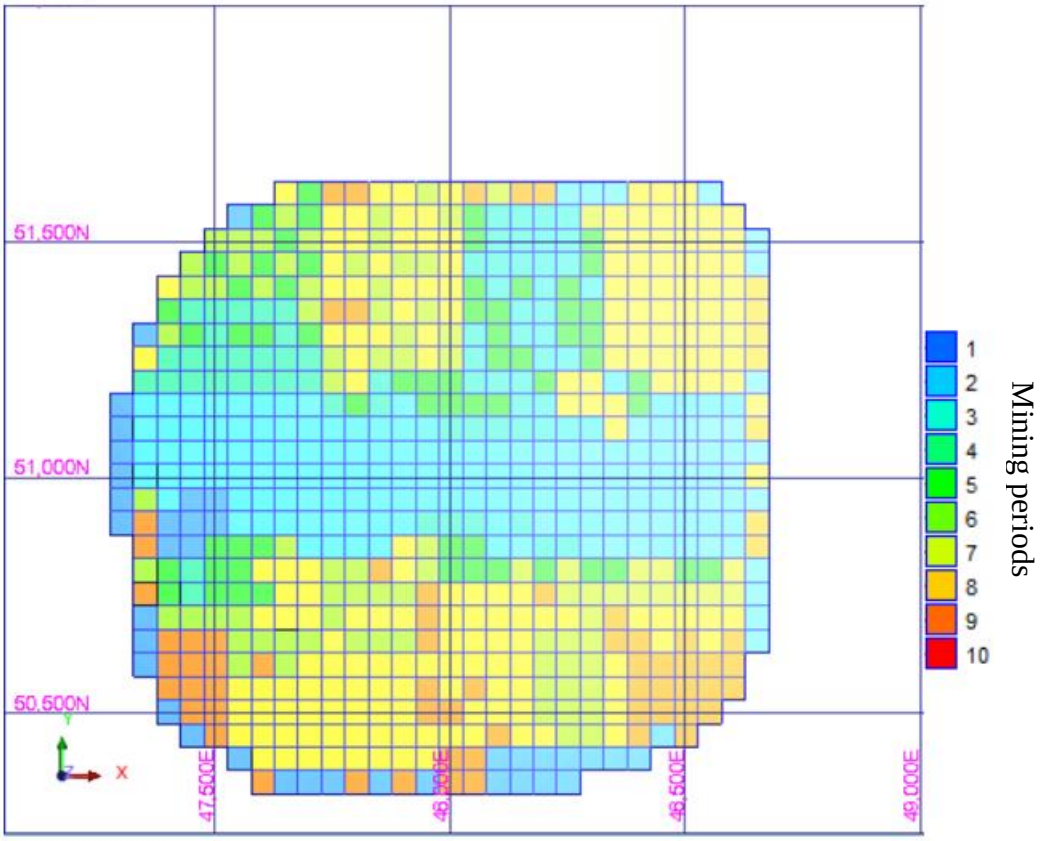

Figure 4-3: Plan view of the block extraction sequence by the DGA on Bench 3

4.4.1.1 Comparative analysis: MILP model with CPLEX and DGA results

The results from the DGA for Scenario 1 model was compared with a similar implementation from the MILP model with CPLEX at 0% optimality gap. The NPV generated from the proposed DGA and MILP model with CPLEX were $1,830 M and $1,929 M respectively. However, the total time taken for the MILP to generate its results was 4.1 hours whereas the DGA generated its results in 1.9 hours. The DGA optimal solution was achieved in a time that is 53.6% better than the MILP model solution with CPLEX. Although the MILP model with CPLEX optimal solution was better than the DGA, the NPV of the DGA solution was 5.1% less than that of the MILP solution. Table 4-11 shows the time duration, optimality gap and NPV comparison of the MILP model with CPLEX and DGA. It can be seen from Figure 4-4A that the DGA respected the maximum annual mining capacity constraint which was set at 32 Mt across all the scheduling periods. Although less material was extracted in

the first period, the extraction gradually ramped up and was uniform for the subsequent periods until declining in the last period. The minimum capacity for the first and last periods were unconstrained to allow for flexibility in the mining. The maximum annual processing capacity of 14 Mt was respected by the DGA as seen in Figure 4-4A. The DGA was able to generate a uniform schedule over the mine life. The total tonnage and ore tonnage generated by the DGA are shown in Table 4-11. Figure 4-5 shows the graph of the average ore bitumen grade for the DGA and the MILP with CPLEX production schedules. It can be seen from Figure 4-5 that there is a gradual decline of the ore bitumen grade as the mine life progresses, which ultimately influences the NPV.

Table 4-11: Solution comparison between the MILP model with CPLEX and the DGA

Parameter	MILP model with CPLEX	DGA
Number of blocks	4476	4476
Tonnage mined (Mt)	287	285
Ore processed (Mt)	121	124
NPV ($M)	1,929	1,830
Runtime (hours)	4.1	1.9
CPLEX Optimality gap (%)	0	-

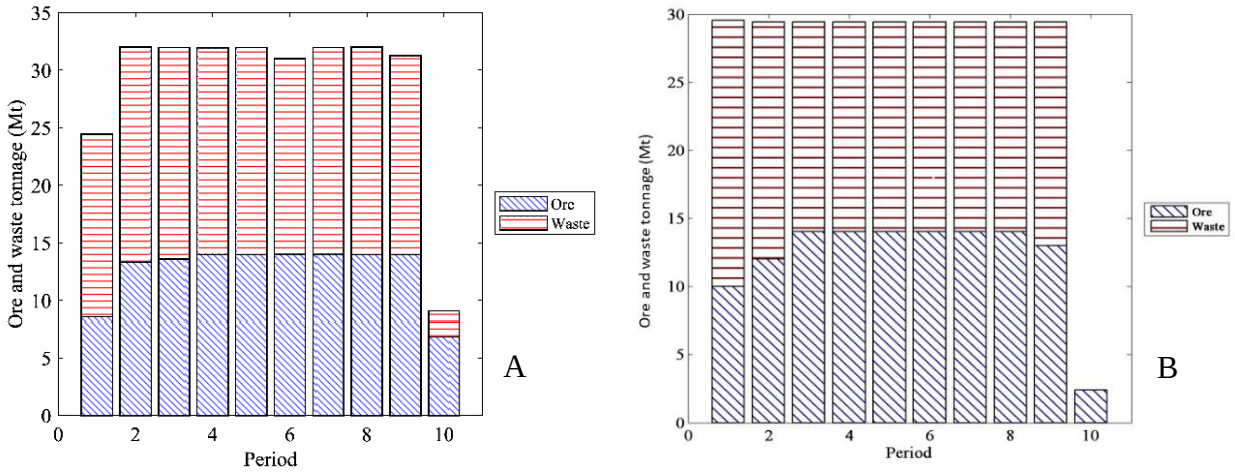

Figure 4-4: Total tonnages mined. (A) Illustrates the total tonnage mined by the DGA and (B) illustrates the total tonnage mined from the MILP model with CPLEX (Mbadozie, 2020)

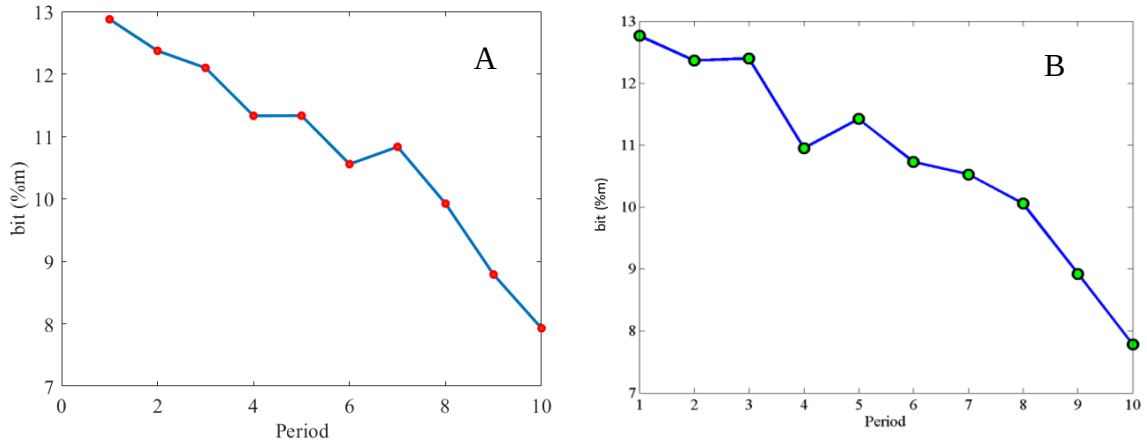

Figure 4-5: Average ore bitumen grade from the DGA illustrated in (A) and average ore bitumen grade from the MILP model with CPLEX illustrated in (B)

4.4.2 Scenario 2 results and discussion: SGA

For Scenario 2, the motivation for the SGA model was to ascertain the impact of grade uncertainty on the production schedule. In order to achieve this, multiple simulated orebody realizations were used as the input to the optimization problem. The technical and economic requirements for scenario 1 were maintained in this scenario. However, risk parameters in Table 4-4 were applied to the objective function as described in Chapter 3.4 to manage the risk associated with the deviations from the production targets. Table 4-12 shows the scheduling results. Figure 4-8 and Figure 4-9 show the extraction sequence of the SGA.

Table 4-12: Scheduling results for the SGA

Period	Total tonnage (Mt)	Ore tonnage (Mt)	Average ore bitumen grade (%m)
1	24.44	9.71	11.67
2	31.92	13.30	11.80
3	31.91	13.51	11.61
4	31.93	13.53	11.72
5	31.91	13.84	11.67
6	30.90	13.90	11.70

7	31.93	13.93	11.52
8	31.94	13.98	11.65
9	31.22	12.9	11.41
10	9.10	5.70	11.73

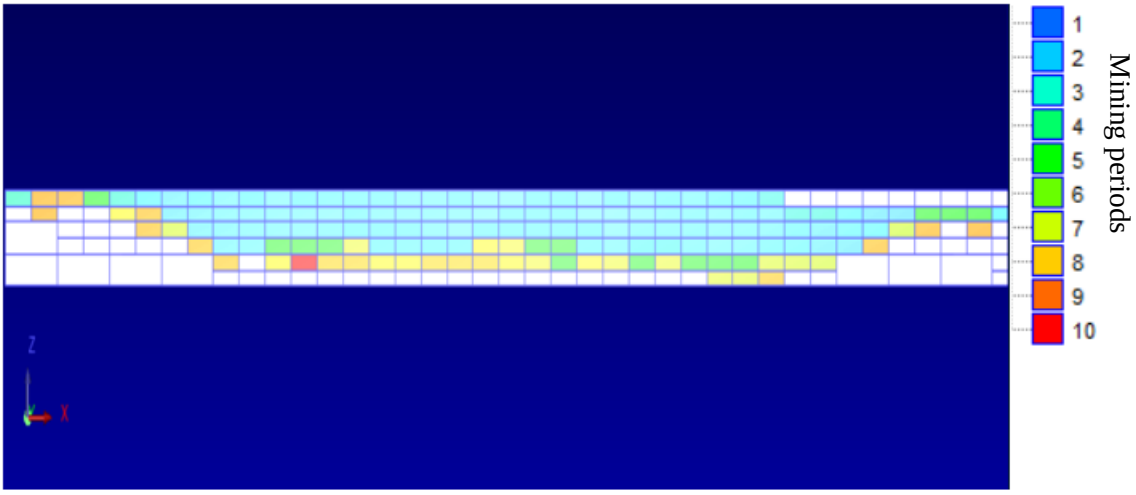

Figure 4-6: Cross sectional view of the block extraction sequence by the SGA

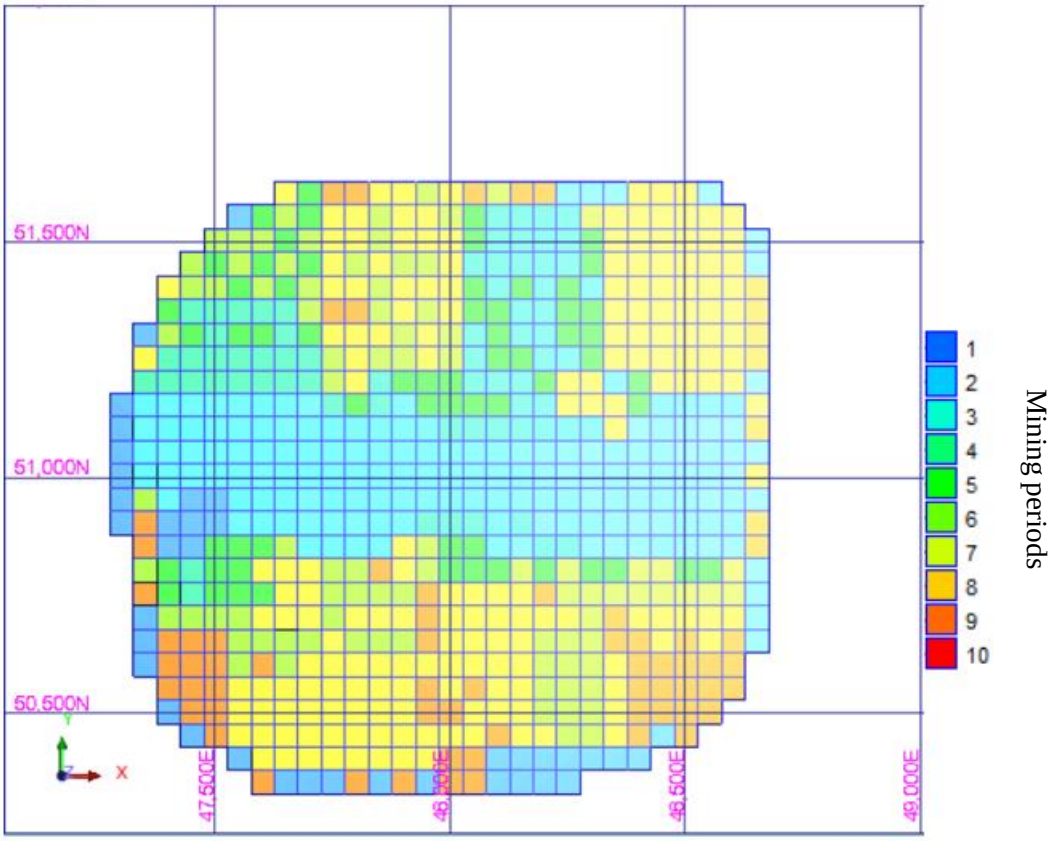

Figure 4-7: Plan view of the block extraction sequence by the SGA on Bench 3

4.4.2.1 Comparative analysis: SMILP model with CPLEX and SGA results

The impact of grade uncertainty on the production schedule is evident based on the NPV generated from the proposed SGA and SMILP model with CPLEX. The NPV generated from the SGA and SMILP model with CPLEX were $2,128 M and $2,248 M respectively. The NPV for the SGA was 5.3% less than that for the SMILP model with CPLEX. The optimality gap for the SMILP model with CPLEX was set at 5%. The total time taken for the SMILP to generate its results was 11.70 hours whereas the SGA generated its results in 2.9 hours. Table 4-13 shows the solution comparison between the SMILP model with CPLEX and the SGA. The impact of grade uncertainty is evident in the NPV generated by the stochastic schedule. The NPV generated by the SGA schedule was 16.3% better than the NPV from the DGA schedule. In Figure 4-8, a comparison between the SGA and the SMILP model with CPLEX is shown. The capacity constraints were respected by the SGA as seen

in Figure 4-8A. For the first and last periods, the minimum capacity were unconstrained to allow for flexibility in the mining. Figure 4-9 shows the average ore bitumen grade comparison between the SGA and SMILP model as well as comparison with individual orebody realizations. From Figure 4-10, it can be observed that, the stochastic model maintained a balanced average grade throughout the mine life, which accounted for the improvement in NPV compared to the DGA's average grade, which declined gradually as the mine life progressed. The significant difference in the average bitumen grade was mainly due to the different reserve modelling techniques used in the cases.

Table 4-13: Solution comparison between the SMILP model with CPLEX and the SGA

Parameter	SMILP model with CPLEX	SGA
Number of blocks	4476	4476
Tonnage mined (Mt)	290	287
Ore processed (Mt)	124	125
NPV ($M)	2,248	2,128
Runtime (hours)	11.7	2.9
CPLEX Optimality gap	5%	-

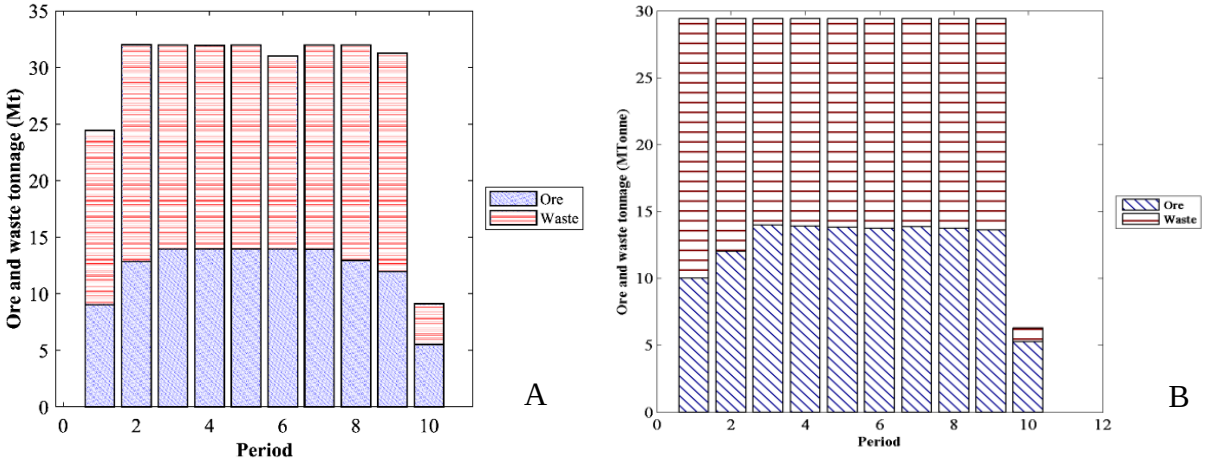

Figure 4-8: Total tonnages mined. (A) Illustrates the total tonnage mined by the SGA and (B) illustrates the total tonnage mined by the SMILP model with CPLEX (Mbadozie, 2020)

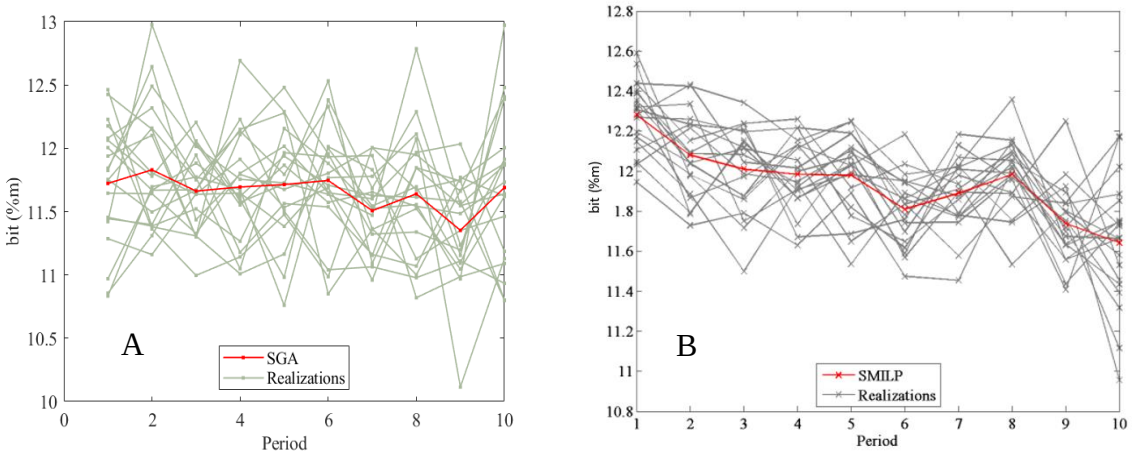

Figure 4-9: Average ore bitumen grade from the SGA illustrated in (A) with 20 realizations and the average ore bitumen grade from the SMILP model with CPLEX illustrated in (B) with 20 realizations

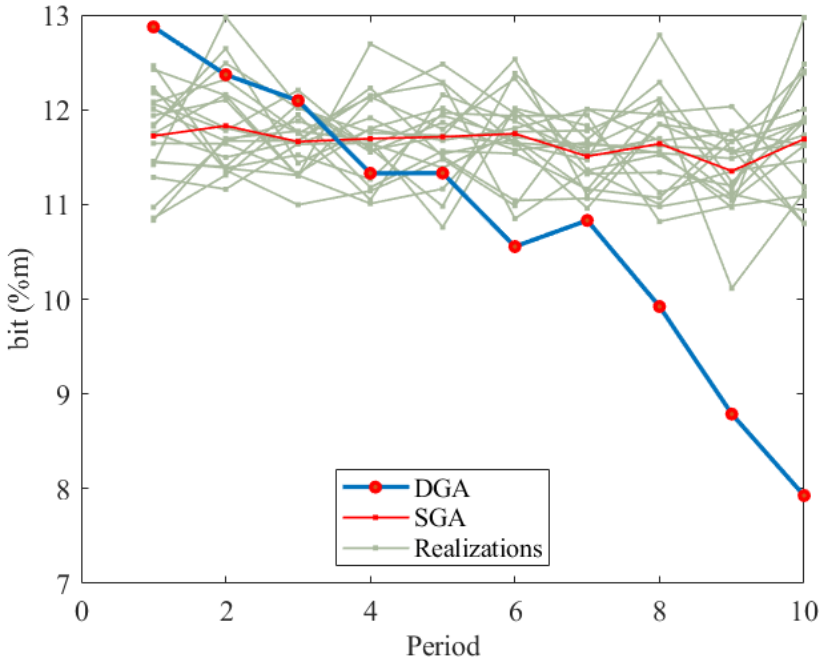

Figure 4-10: Average ore bitumen grade comparison for DGA, SGA and 20 realizations

4.5 Conclusions for Case Study 1

In this case study, the GA framework was implemented on two different scenarios. The first scenario was a deterministic model (DGA) and the second was a stochastic model(SGA). Both scenarios were compared to similar implementation from MILP models with CPLEX. The GA framework proved to be capable of solving the OPPS problem. The DGA model was computational efficient achieving its solution in a time which was 53.6% better than the MILP model with CPLEX in comparison. The SGA also had a 75.2% better computational time than the SMILP model with CPLEX. However, in both scenarios the NPV from the MILP and SMILP models were slightly higher than the DGA and SGA models.

4.6 Case Study 2 implementation

Case Study 2 featured an oil sands dataset consisting of 1569 blocks in the geological block model, detail of this is seen in Table 4-1. The GA implementation outlined for Case Study 1 was subsequently applied here. Case Study 2 also had two scenarios. Scenario 1 was a deterministic scenario with no consideration of grade uncertainty. Scenario 2 was the stochastic scenario which considered grade uncertainty through 20 simulated equally orebody realizations. The DGA was applied in Scenario 1 and the SGA in Scenario 2. For Scenario 1, with a set optimality gap of 10%, the MILP model generated an NPV of $10,175 M after 226 hours. The DGA generated an NPV of $9,142 M at 12.9% gap after 1.3 hours. This further emphasizes the application of metaheuristics to NP-hard combinatorial optimization problems. The optimal solution for Scenario 2 was intractable using the SMILP model with CPLEX. There was no integer solution generated by the SMILP model for Scenario 2 after 28 days of computation (Job scheduling policies - Compute Canada Document, 2022). However, the GA model was capable of generating a solution in 1.5 hours for the SGA. To verify the solution generated by the GA model, relaxed LP forms of the MILP and SMILP problems were solved with CPLEX. The solution generated from the relaxed LP problem was compared with the solution from the GA model. Equation (4.1) was used to compute and determine the gap for the GA solution relative to the best bound of the relaxed LP solution.

4.6.1 Scenario 1 results and discussion: DGA

The scenario presented here consisted of the DGA model with the orebody estimation based on Ordinary Kriging. There was no consideration of grade uncertainty. The production schedule was limited by the processing and mining requirement in Table 4-2 and Table 4-3. The scheduling results from the DGA for this scenario are shown in Table 4-14, Figure 4-11 and Figure 4-12. Figure 4-11A shows the tonnages of ore and waste mined per period by the DGA whiles Figure 4-11B shows the tonnages of ore and waste mined per period by the MILP model with CPLEX. The mining and processing capacities were set at 150 Mt and 50 Mt respectively. The DGA respected the capacity constraints as seen in Figure 4-11. The average ore bitumen grade for the production schedule by the DGA and the MILP model with CPLEX is shown in Figure 4-12A and Figure 4-12B respectively.

The objective function value generated by the relaxed LP was 11,306 and that for the DGA was 9,846. Using Equation (4.1) gives us a gap of 12.9%. This therefore means the DGA solution is in the worst case scenario at 12.9% of the optimal solution to the MILP model if it exists since the relaxed LP solution is the upper bound to it. The NPV generated from the DGA was $9,142 M. The solution comparison, optimality gap and runtime are shown in Table 4-15.

Table 4-14: Scheduling results generated by the DGA

Period	Total tonnage (Mt)	Ore tonnage (Mt)	Average ore bitumen grade (%m)
1	149.01	29.86	8.42
2	149.21	33.89	8.38
3	148.20	37.88	8.46
4	148.56	39.94	8.29
5	149.52	44.78	8.46
6	149.41	44.79	8.56
7	149.06	44.89	8.24
8	148.97	44.84	8.58
9	148.96	44.92	8.32

10	148.95	44.83	8.50
11	149.08	44.90	8.47
12	148.88	44.97	7.84
13	148.94	44.88	8.62
14	148.86	44.73	8.47
15	149.46	47.78	8.64
16	149.62	44.42	8.24
17	148.95	43.56	8.49
18	149.94	43.96	8.07
19	148.90	43.71	8.24
20	147.64	19.01	8.04

Table 4-15: Solution comparison between the DGA and MILP model with CPLEX

Parameter	**MILP model with CPLEX**	**DGA**
Number of blocks	1569	1569
Tonnage mined (Mt)	2995	2980
Ore processed (Mt)	934	832
NPV ($M)	10,175	9,142
Runtime (hours)	226	1.3
Optimality gap from relaxed LP (%)	10.0	12.9

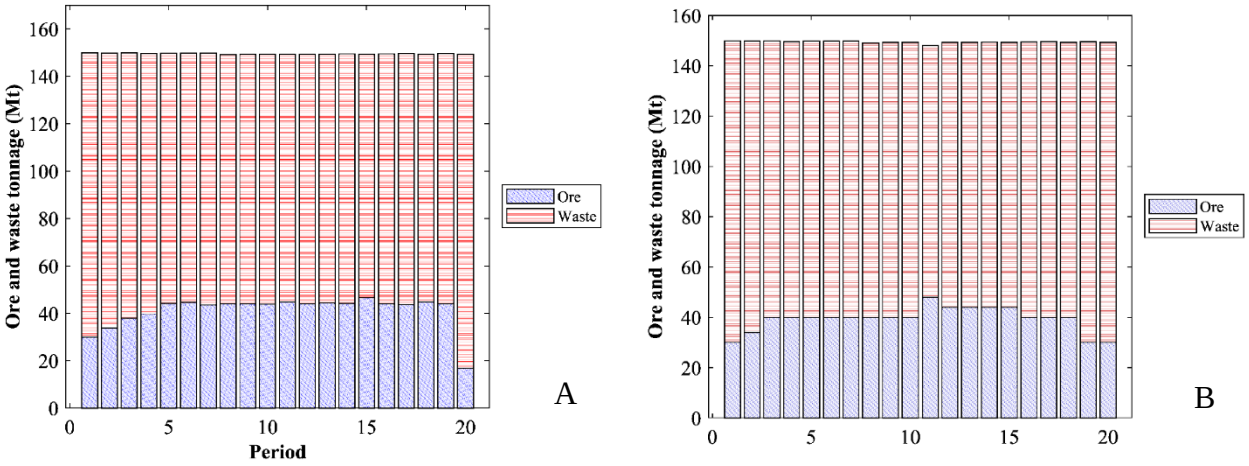

Figure 4-11: Total tonnages mined. (A) Illustrates the total tonnage mined by the DGA and (B) illustrates the total tonnage mined by the MILP model with CPLEX

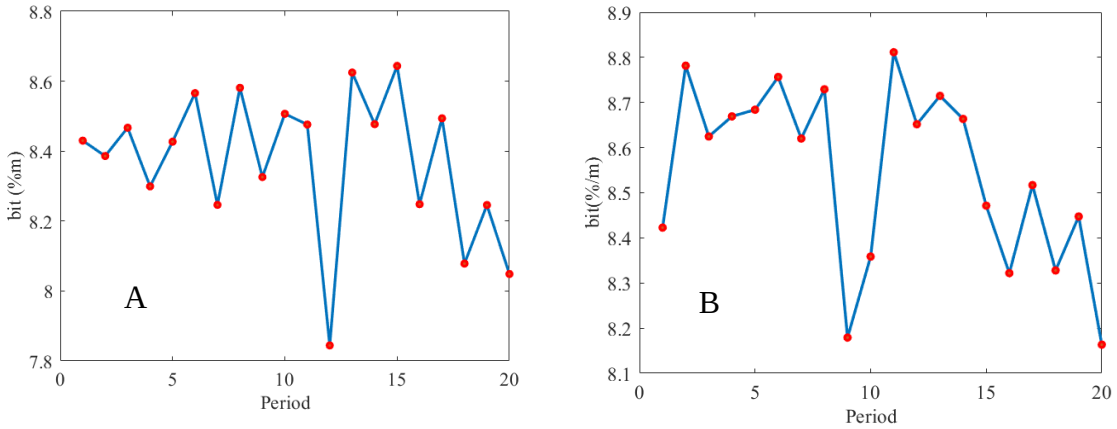

Figure 4-12: Average ore bitumen grade from the DGA illustrated in (A) and the average ore bitumen grade from the MILP model with CPLEX illustrated in (B)

4.6.2 Scenario 2 results and discussion: SGA

The SGA scenario for this case study considered grade uncertainty. The block model data used here consisted of multiple simulated orebody realizations from the oil sands dataset modelled and simulated by Mbadozie (2020). This block model data was used as the input to the optimization problem. The technical and economic constraints outlined in Table 4-2, Table 4-3 and Table 4-4 were applied. The scheduling results from the SGA for this scenario

are shown in Table 4-16, Figure 4-13, and Figure 4-14. Figure 4-13 shows the tonnages of ore and waste mined per period by the SGA. The SGA respected the capacity constraints as seen in Figure 4-13. The average ore bitumen grade for the production schedule is shown in Figure 4-14. Figure 4-15 illustrates the comparison between the average ore bitumen grade generated by the SGA model with the 20 different realizations and the DGA model. It can be seen from the figure that the average ore bitumen grade for the DGA was lower compared to the SGA model. This is as a result of the different reserve modelling approaches used in the cases. This resulted in an increase in the optimal solution for the SGA model. The SGA model maintained a balanced ore bitumen grade across the scheduling periods in comparison to the 20 individual realizations.

The objective function value generated by the relaxed LP was 12,810 and that for the SGA was 11,629. Using Equation (4.1) gives us a gap of 10.6%. This therefore means the SGA solution is in the worst case scenario at 10.6% of the optimal solution to the SMILP model if it exists since the relaxed LP solution is the upper bound to it. The NPV generated from the SGA was $10,045 M. The solution comparison, optimality gap and runtime are shown in Table 4-17.

Table 4-16: Scheduling results for the SGA

Period	Total tonnage (Mt)	Ore tonnage (Mt)	Average ore bitumen grade (%m)
1	149.91	29.98	9.80
2	149.84	33.96	10.14
3	149.94	37.92	9.79
4	149.62	39.98	9.53
5	149.72	49.95	9.94
6	149.87	49.93	10.04
7	149.82	49.82	9.56
8	149.11	49.91	10.16
9	149.32	49.92	9.93
10	149.33	49.90	9.82
11	149.28	49.96	9.79

12	149.33	49.89	9.39
13	149.36	49.90	9.81
14	149.41	49.93	10.22
15	149.28	47.92	10.05
16	149.46	44.86	9.78
17	149.64	44.94	10.30
18	149.25	44.86	9.84
19	149.55	44.92	9.62
20	149.30	28.64	9.17

Table 4-17: Solution comparison between the SGA and the SMILP model with CPLEX

Parameter	**SMILP model with CPLEX**	**SGA**
Number of blocks	1569	1569
Tonnage mined (Mt)	-	2990
Ore processed (Mt)	-	897
NPV ($M)	-	10,054
Runtime (hours)	*Terminated after 28 days	1.5
Optimality gap from relaxed LP (%)	101.0	10.6

* Job scheduling policies - Compute Canada Cedar Cluster

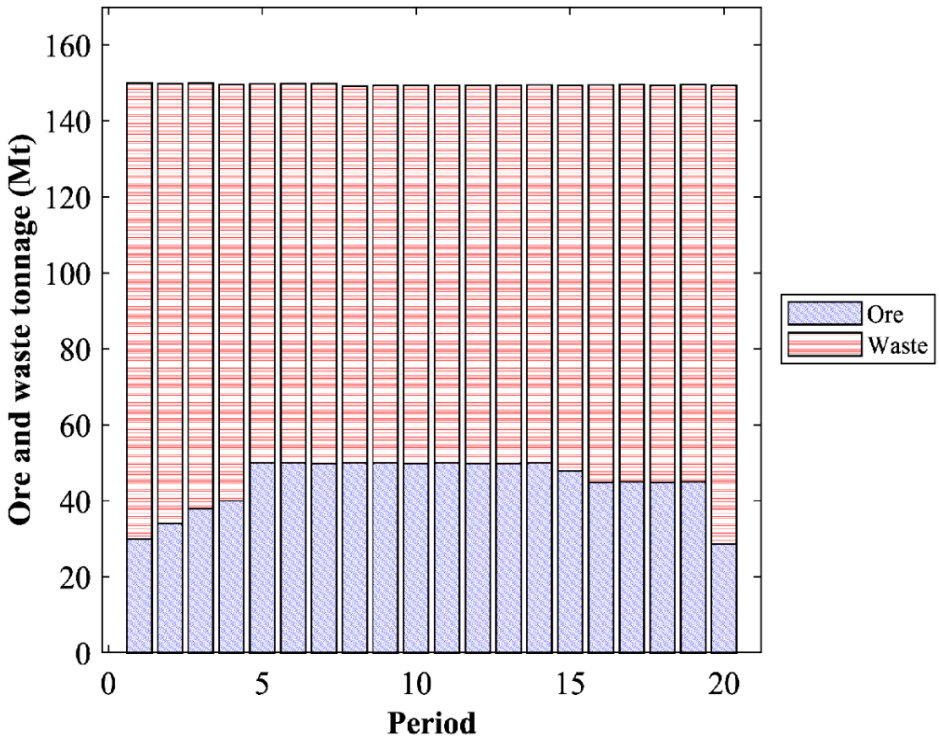

Figure 4-13: Total tonnage mined with the SGA

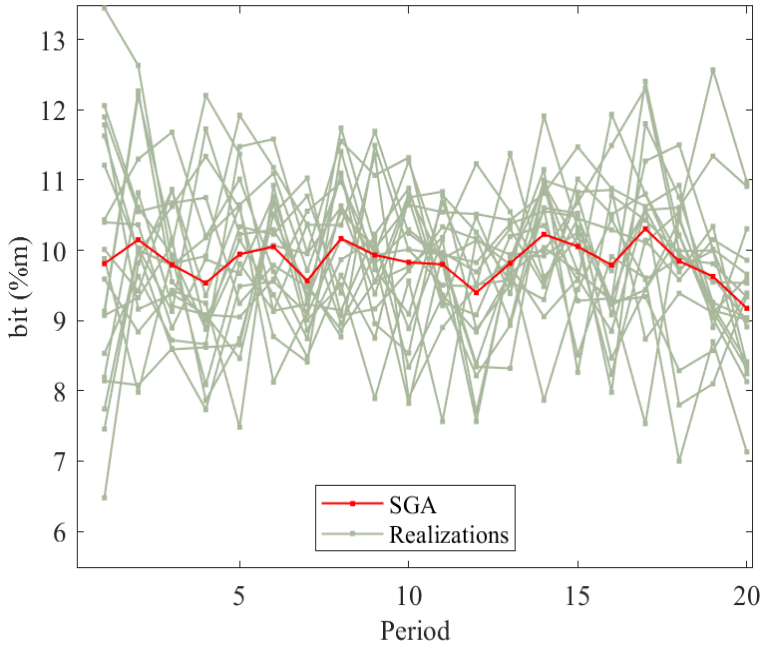

Figure 4-14: Average ore bitumen grade with the SGA and 20 realizations

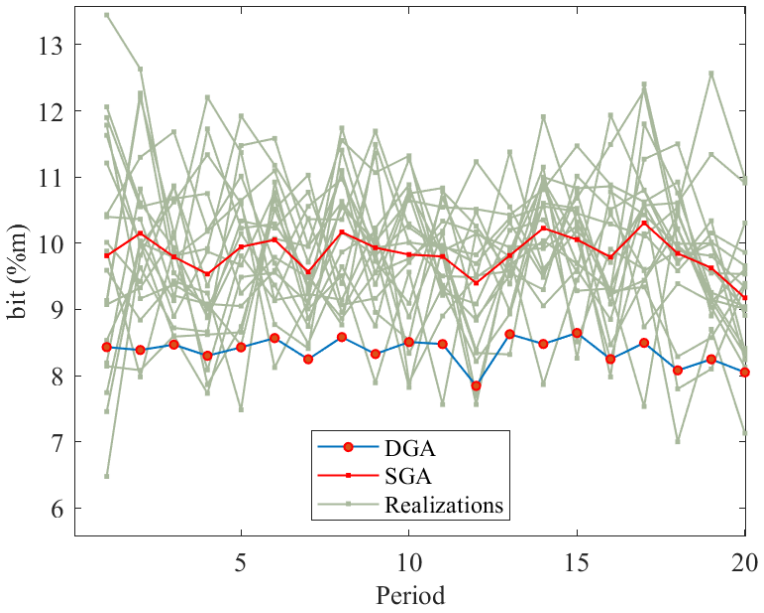

Figure 4-15: Average ore bitumen grade comparison for DGA, SGA and 20 realizations

4.7 Conclusions for Case Study 2

In this case study, the GA framework was implemented on two different scenarios. The first scenario was a deterministic model and the second was a stochastic model. For Scenario 1, with a set optimality gap of 10%, the MILP model generated an NPV of $10,175 M after 226 hours. The DGA generated an NPV of $9,142 M at 12.9% gap after 1.3 hours for this scenario. The SMILP model with CPLEX for Scenario 2 was terminated after 28 days at 101% gap. However, the SGA model was capable of generating results in 1.5 hours at 10.6% gap. In order to determine the optimality gap for the GA results, the relaxed LP forms of the MILP and SMILP problems were solved with CPLEX and compared with the GA results. The DGA and SGA results were in the worst case scenarios, 12.9% and 10.6% to the relaxed LP results for the MILP and SMILP models respectively.

CHAPTER 5

SUMMARY, CONCLUSIONS AND RECOMMENDATIONS

5.1 Summary of the research

Metaheuristics have proven to be computationally efficient in solving different combinatorial optimization problems throughout the literature. This has led to the interest of researchers in applying metaheuristics to mine planning and optimization. There are different aspects of mine planning; one of such aspects is the open pit production scheduling (OPPS) problem. The OPPS problem primarily determines the time and sequence of extraction of ore and waste from a mine to maximize the NPV of the mining operation. Exact optimization methodologies based on mathematical programming models have been well researched and applied to the mining production scheduling problem. However, these models as identified in the literature either tends to be intractable for large-scale and complex problems, or requires large amount of computing resources to generate solution for large-scale tractable problems. In the OPPS problem, a major factor that can affect the overall NPV and prevent the mining project from achieving its production targets is grade uncertainty. The ability to have a production scheduling framework that incorporates grade uncertainty while being computationally efficient in optimizing the OPPS problem has been a challenge in the literature for some time now.

This research has formulated and verified a metaheuristic optimization framework based on GA to optimize the NP-hard OPPS problem while incorporating grade uncertainty. Results from the research indicates that the GA model is capable of generating ‘good’ solutions and is computationally efficient in solving the OPPS problem. A summary of the research methodology and developed framework is presented in Figure 5-1.

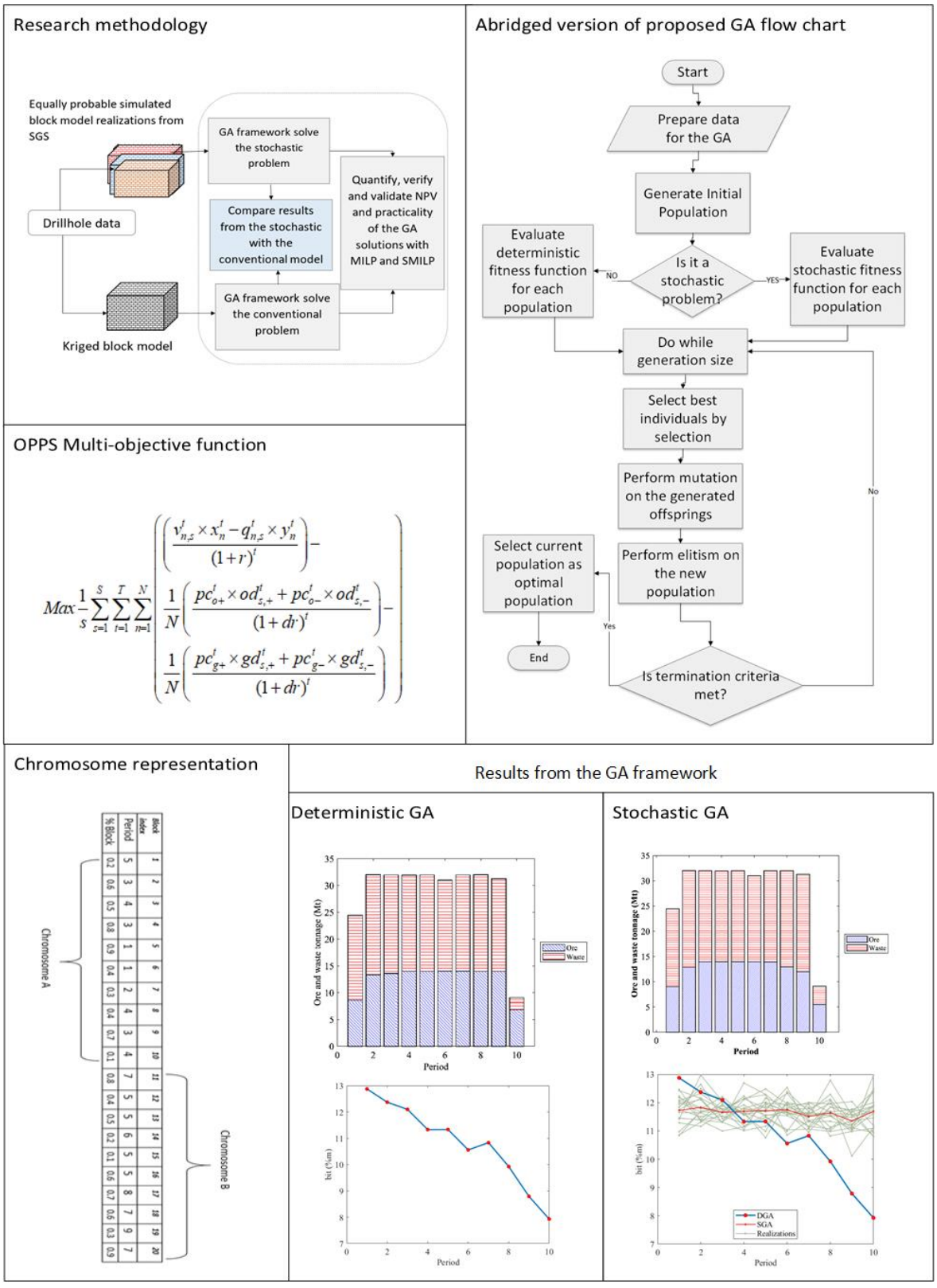

Figure 5-1: Summary of research method and framework developed

MATLAB programming platform was used in the definition and implementation of the GA framework. The main components of the GA framework comprise the objective function and the set of operational and technical constraints. The GA framework was implemented for two case studies and the results were compared to similar implementations with a Mixed Integer Linear Programming (MILP) model and a Stochastic Mixed Integer Linear Programming (SMILP) model to assess the results in terms of practicality, NPV and computational efficiency.

5.2 Conclusions

In this research, a GA framework for solving the OPPS problem was presented and evaluated with two case studies. A multiple chromosome-encoding scheme was proposed and implemented to represent the blocks and periods of extractions. Deterministic and stochastic production scheduling scenarios were investigated in this research. The deterministic production schedule was based on Ordinary Kriging block model, and the stochastic production schedule was based on SGS orebody realizations. The equally probable simulated orebody realizations capture the varying grade distribution in the deposit to allow for consideration of grade uncertainty. Due to the multiple chromosome encoding scheme, the GA was capable of fractional block processing. In the implementation of the GA framework, the relaxed LP solution was used as an upper bound to estimate the optimality gap for the GA solution. The solutions from the GA were compared with that from mathematical programming models with CPLEX solver to assess the practicality of the generated schedules as well as the NPV and computational efficiency.

For the deterministic production scheduling in Case Study 1 Scenario 1, the NPV of the DGA schedule was 5.1% less than that of the MILP model with CPLEX schedule while there was a 53.7% improvement in computational time comparing the DGA solution runtime to that of the MILP model with CPLEX solution runtime. For the second scenario based on stochastic production scheduling, while the NPV of the SGA schedule was 5.3% less than that of the SMILP model with CPLEX schedule, there was 75.2% improvement in computational efficiency comparing the SGA solution runtime to that of the SMILP model

with CPLEX solution runtime. It is also important to note that the NPV generated by the SGA schedule was 16.3% better than the NPV from the DGA schedule for Case Study 1.

For Case Study 2 Scenario 1, with a set optimality gap of 10%, the MILP model generated an NPV of $10,175 M after 226 hours. The DGA generated an NPV of $9,142 M at 12.9% after 1.3 hours. For Case Study 2 Scenario 2, the solution from the SMILP model with CPLEX was terminated after 28 days at 101% gap while the SGA generated solution in 1.5 hours at 10.6% optimality gap. In both case studies, the GA models generated the production schedule results significantly faster, although the NPVs were lower than that from the MILP and SMILP models with CPLEX solver.

In summary, the results generated by the GA were encouraging in the area of computational efficiency. In cases where the mathematical programming model solution runtime is lengthy or intractable, the GA proves to be capable of generating a 'good' solution at a reasonable computational time. Table 5-1 and Table 5-2 show summary comparisons of the case study results.

Table 5-1: Summary of Case Study 1 comparisons

Scenario 1		
Parameter	**MILP model with CPLEX**	**DGA**
Number of blocks	4476	4476
Tonnage mined (Mt)	287	285
Ore processed (Mt)	121	124
NPV ($M)	1,929	1,830
Runtime (hours)	4.1	1.9
CPLEX Optimality gap (%)	0	-
Scenario 2		
Parameter	**SMILP model with CPLEX**	**SGA**
Number of blocks	4476	4476
Tonnage mined (Mt)	290	287
Ore processed (Mt)	124	125

NPV ($M)	2,248	2,128
Runtime (hours)	11.7	2.9
CPLEX Optimality gap (%)	5	-

Table 5-2: Summary of Case Study 2 comparisons

Scenario 1		
Parameter	**MILP model with CPLEX**	**DGA**
Number of blocks	1569	1569
Tonnage mined (Mt)	2995	2980
Ore processed (Mt)	934	832
NPV ($M)	10,175	9,142
Runtime (hours)	226	1.3
Optimality gap from relaxed LP (%)	10.0	12.9
Scenario 2		
Parameter	**SMILP model with CPLEX**	**SGA**
Number of blocks	1569	1569
Tonnage mined (Mt)	-	2990
Ore processed (Mt)	-	897
NPV ($M)	-	10,045
Runtime (hours)	*Terminated after 28 days	1.5
Optimality gap from relaxed LP (%)	101.0	10.6

* Job scheduling policies - Compute Canada Cedar Cluster

5.3 Contributions of MASc. research

This research has designed and implemented a metaheuristic optimization framework based on GA to optimize the open pit production scheduling problem. The main contributions made by this research to the body of knowledge are as follows:

1. This research has developed an integrated GA framework that optimizes both the conventional and stochastic OPPS problem in the presence of grade uncertainty. It expands on the frontiers of stochastic mine planning optimization and creates the platform for developing specialized mine planning software packages.
2. The research extended the GA parameters namely; mutation and crossover to allow for adequate representation of the OPPS problem. This ensures the generated production schedule provides a practical mining environment during extraction.
3. The developed GA framework is computationally efficient in solving the stochastic OPPS problem with a reasonable runtime. This enables step-changes in the planning of open pit operations considering grade uncertainty.
4. The formulated GA framework employs a multiple chromosome encoding scheme to accommodate block processing over multiple periods. This improves the net present value of the resulting production schedule compared to the single chromosome encoding scheme.
5. The research has documented a workflow to assess the optimality of the GA solution in comparison with the relaxed LP results.

5.4 Recommendations

This research has made adequate contributions to the body of knowledge in the area of mining production scheduling with metaheuristic optimization framework based on GA. New efforts has been made in the chromosome encoding scheme of the GA initial population to suitably represent the open pit production scheduling problem. Despite these efforts, there is more room for improvement to build a robust GA framework that ultimately satisfy the many demands of the OPPS problem. The following areas and recommendations can be investigated to improve the GA framework and extend it to different aspects of mine planning:

1. Incorporate vertical bench advancement rate control to improve practicality of mining sequence.
2. Investigate alternative combinations of genetic parameters to improve the GA computational efficiency and solution quality.

3. Stockpiling was not considered in the GA framework. Future research should extend the GA model to include stockpile management.
4. This research considers grade uncertainty in the production scheduling optimization. However, parameters related to future economic data were all kept constant which means a change in these parameters will necessitate a re-optimization. Future research should consider the integration of economic uncertainty for the OPPS problem.
5. Experiment with the GA result as an initial solution to the MILP model with CPLEX to evaluate the improvement in NPV and the corresponding reduction in solution time when the two models are implemented in series.

BIBLIOGRAPHY

[1] Aarts, E.H. and van Laarhoven, P.J. (1987). Simulated annealing: a pedestrian review of the theory and some applications. In *Pattern recognition theory and applications*, pp. 179-192. Springer, Berlin, Heidelberg.

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

[3] Ahmadi, M. R. and Shahabi, R. S. (2018). Cutoff grade optimization in open pit mines using genetic algorithm. *Resources Policy*, **55**(March 2018), 184-191.

[4] Ahn, C. W. and Ramakrishna, R. S. (2003). Elitism-based compact genetic algorithms. *IEEE Transactions on Evolutionary Computation,* **7**(4)**,** 367-385.

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

[6] Alipour, A., Khodaiari, A. A., Jafari, A. and Tavakkoli-Moghaddam, R. (2020). Production scheduling of open-pit mines using genetic algorithm: a case study. *International Journal of Management Science and Engineering Management,* **15**(3)**,** pp. 176-183.

[7] Amponsah, S., Eme, P., Takouda, P.M and Ben-Awuah, E. (2021). *Genetic algorithm for open pit mine production scheduling optimisation problems*. In Proceedings of the 40th Conference on Application of Computers and Operations Research in the Minerals Industry (APCOM 2021). Johannesburg, South Africa, pp. 335-346.

[8] Amponsah, S., Takouda, P.M and Ben-Awuah, E. (2021). *A stochastic genetic algorithm for mining-related optimization problems*. In Proceedings of the 14th

International Conference on Multiple Objective Programming and Goal Programming (MOPGP 2021). 20-21 December 2021, Online.

[9] Anani, A.K., (2016). Applications of simulation and optimization techniques in optimizing room and pillar mining systems. Missouri University of Science and Technology. Pages 241

[10] Anirudh, S. (2020). Sliding window algorithm Available at https://redquark.org/cotd/sliding_window/. [accessed 1 Dec. 2020].

[11] Askari-Nasab, H., Awuah-Offei, K. and Eivazy, H. (2010). Large-scale open pit production scheduling using mixed integer linear programming. International Journal of Mining and Mineral Engineering, **2**(3), 185-214.

[12] Askari-Nasab, H., Pourrahimian, Y., Ben-Awuah, E. and Kalantari, S., 2011. Mixed integer linear programming formulations for open pit production scheduling. Journal of Mining Science, **47**(3), 338-359.

[13] Badiozamani, M.M. and Askari-Nasab, H., (2010). Lagrangian relaxation of the MILP open pit production scheduling formulation. *Mining Optimization Laboratory*, p.157.

[14] Ben-Awuah, E. and Askari-Nasab, H. (2011). Oil sands mine planning and waste management using mixed integer goal programming. *International Journal of Mining, Reclamation and Environment*, **25**(3)**,** 226-247.

[15] Ben-Awuah, E., Askari-Nasab, H., Awuah-Offei, K. and Awuah-Offei, K. (2012). Production scheduling and waste disposal planning for oil sands mining using goal programming. *Journal of Environmental Informatics,* **20**(1), 20-33.

[16] Ben-Awuah, E., Richter, O., Elkington, T. and Pourrahimian, Y., (2016). Strategic mining options optimization: Open pit mining, underground mining or

both. *International Journal of Mining Science and Technology*, ***26***(6), pp.1065-1071.

[17] Ben-Awuah, E., Askari-Nasab, H., Maremi, A. and Seyed Hosseini, N. (2018). Implementation of a goal programming framework for production and dyke material planning. *International Journal of Mining, Reclamation and Environment,* **32**(8)**,** 536-563.

[18] Bianchi, L., Dorigo, M., Gambardella, L. M. and Gutjahr, W. J. (2009). A survey on metaheuristics for stochastic combinatorial optimization. *Natural Computing,* **8**(2)**,** 239-287.

[19] Blickle, T. and Thiele, L. (1995). *A mathematical analysis of tournament selection*. In Proceedings of 6th International Conference on Genetic Algorithms, San Francisco, California, 95, pp. 9-15.

[20] Caccetta, L. and Hill, S. P. (2003). An application of branch and cut to open pit mine scheduling. *Journal of Global Optimization,* **27**(2-3), 349-365.

[21] Cantú-Paz, E., 1998. A survey of parallel genetic algorithms. *Calculateurs paralleles, reseaux et systems repartis*, **10**(2), 141-171.

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

[23] Dagdelen, K. (1985). Optimum multi-period open pit mine production scheduling by Lagrangian parameterization. PhD Thesis, University of Colorado, Colorado, Pages 325.

[24] Dagdelen, K. and Johnson, T. (1986). *Optimum open pit mine production scheduling by Lagrangian parameterization*. In Proceedings of 19th International Symposium on the Application of Computers and Operations Research in the Mineral Industry,

Society for Mining, Metallurgy & Exploration, Pennsylvania State University, USA, pp. 127-142 .

[25] Dasgupta, D. and Michalewicz, Z. (1997). Evolutionary algorithms— An Overview. In: Dasgupta, D., Michalewicz, Z. (eds) Evolutionary Algorithms in Engineering Applications. Springer, Berlin, Heidelberg, pp. 3-28. https://doi.org/10.1007/978-3-662-03423-1_1.

[26] Davis, L. (1985). *Applying adaptive algorithms to epistatic domains*. In Proceedings of 9th International Joint Conference on Artificial Intelligence, 85, pp. 162-164.

[27] Deb, K., Pratap, A., Agarwal, S. and Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE transactions on evolutionary computation*, **6**(2)**,** 182-197.

[28] Denby, B. and Schofield, D. (1994). Open-pit design and scheduling by use of genetic algorithms. *Transactions of the Institution of Mining and Metallurgy. Section A. Mining Industry*, **103**(1994), A21-A26.

[29] Deneubourg, J. L., Aron, S., Goss, S. and Pasteels, J. M. (1990). The self-organizing exploratory pattern of the argentine ant. *Journal of Insect Behavior*, **3**(2)**,** 159-168.

[30] Deutsch, C. and Journel, A. (1998). *GSLIB: Geostatistical Software Library.* Oxford Univ.Press, New York, 2nd ed, Pages 384.

[31] Dimitrakopoulos, R. and Ramazan, S. (2004). Uncertainty based production scheduling in open pit mining. *Transactions of the Society for Mining, Metallurgy, and Exploration*, ***316*** (3), 106-112.

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

[33] Dorigo, M. and Blum, C. (2005). Ant colony optimization theory: A survey. *Theoretical Computer Science*, **344**(2-3)**,** 243-278.

[34] Dorigo, M., Caro, G. D. and Gambardella, L. M. (1999). Ant algorithms for discrete optimization. *Artificial life*, **5**(2)**,** 137-172.

[35] Dumitrescu, D., Lazzerini, B., Jain, L. C. and Dumitrescu, A. (2000). *Evolutionary computation*, Boca Raton, FL CRC press.

[36] Eshelman, L. J., Caruana, R. A. and Schaffer, J. D. (1989). Biases in the crossover landscape. In *Proceedings of the 3rd International Conference on Genetic Algorithms*. Morgan Kaufmann,San Mateo,pp. 10-19.

[37] Falkenauer, E., (1999). The worth of the uniform [uniform crossover]. In Proceedings of the 1999 Congress on Evolutionary Computation-CEC99 (Cat. No. 99TH8406) (Vol. 1, pp. 776-782). IEEE.

[38] Feng, Y., Wang, G.-G., Deb, S., Lu, M. and Zhao, X.-J. (2017). Solving 0–1 knapsack problem by a novel binary monarch butterfly optimization. *Neural Computing and Applications*, **28**(7)**,** 1619-1634.

[39] Franco-Sepúlveda, G., Del Rio-Cuervo, J. C. and Pachón-Hernández, M. A. (2019). State of the art about metaheuristics and artificial neural networks applied to open pit mining. *Resources Policy*, 60, 125-133.

[40] Franco-Sepúlveda, G., Del Rio-Cuervo, J. C. and Pachón-Hernández, M. A. (2019). State of the art about metaheuristics and artificial neural networks applied to open pit mining. *Resources Policy*, 60, 125-133.

[41] Fouskakis, D. and Draper, D. (2002). Stochastic optimization: a review. International Statistical Review, 70(3), 315-349.

[42] Gen, M. and Cheng, R. (1997). Genetic Algorithms and Engineering Design. New York: John Wiley&Sons. Inc.

[43] Gen, M., Cheng, R. and Lin, L. (2008). *Network models and optimization: Multiobjective genetic algorithm approach*, Springer Science & Business Media.

[44] Geovia Dassault Systems. (2017). GEOVIA Whittle software (Version 4.7.1). Vancouver, BC, Canada: Geovia Dassault Systems.

[45] Gershon, M. E. (1983). Optimal mine production scheduling: evaluation of large scale mathematical programming approaches. *International journal of mining engineering,* **1**(4)**,** 315-329.

[46] Glover, F. (1986). Future paths for integer programming and links to artificial intelligence. *Computers & operations research,* **13**(5)**,** 533-549.

[47] Glover, F. (1990). Artificial intelligence, heuristic frameworks and tabu search. *Managerial and Decision Economics,* **11**(5), 365-375.

[48] Goldberg, D. E. (1990). A note on Boltzmann tournament selection for genetic algorithms and population-oriented simulated annealing. *Complex Systems,* **4**(4), 445-460.

[49] Goldberg, D. E. and Deb, K. (1991). A Comparative analysis of selection schemes used in genetic algorithms. *Foundations of Genetic Algorithms*, Morgan Kaufmann, San Mateo, California, 1, 69-93.

[50] Goldberg, D. E. and Holland, J. H. (1988). Genetic algorithms and machine learning. *Machine Learning **3***(2/3), 95–99.

[51] Goldberg, D. E. and Lingle, R. (1985). *Alleles, loci, and the traveling salesman problem*. In Proceedings of International Conference on Genetic Algorithms and their Applications, Carnegie-Mellon University, Pittsburgh, PA, pp. 154-159.

[52] Goodfellow, R. and Dimitrakopoulos, R. (2013). Algorithmic integration of geological uncertainty in pushback designs for complex multiprocess open pit mines. *Mining Technology,* **122**(2)**,** 67-77.

[53] Grefenstette, J., Gopal, R., Rosmaita, B. and Van Gucht, D. (1985). *Genetic algorithms for the traveling salesman problem*. In Proceedings of 1st International Conference on Genetic Algorithms and their Applications, Lawrence Erlbaum, 160, pp.160-168.

[54] Hansheng, L. and Lishan, K. (1999). Balance between exploration and exploitation in genetic search. *Wuhan University Journal of Natural Sciences,* **4**(1)**,** 28-32.

[55] Holland, J. H. (1992). Genetic Algorithms. *Scientific American,* **267**(1)**,** 66-73.

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

[57] Hu, X.-B. and Di Paolo, E. (2009). An efficient genetic algorithm with uniform crossover for the multi-objective airport gate assignment problem. *Multi-objective memetic algorithms.* Berlin, Heidelberg: Springer, pp. 71-89.

[58] IBM, ILOG (2017). CPLEX reference manual and software. Ver. 12.8, New York, USA Available at: https://www.ibm.com/docs/en/icos/12.8.0.0?topic=parameters-relative-mip-gap-tolerance [Accessed: 1 January 2022].

[59] Job scheduling policies - Compute Canada Document [online]. (2022). CC Doc. Available from: https://docs.alliancecan.ca/wiki/Job_scheduling_policies.

[60] Johnson, T. B. (1969). *Optimum open-pit mine production scheduling*. In proceedings of 8th International Symposium on the Application of Computers and Operations Research in the Mineral Industry, Salt Lake City, Utah, pp. 539-562.

[61] Kennedy, J. and Eberhart, R. (1995). *Particle swarm optimization*. In Proceedings of the IEEE International Conference on Neural Networks, Perth, Australia, 4, pp. 1942-1948.

[62] Khan, A. and Niemann-Delius, C. (2014) 'Production scheduling of open pit mines using particle swarm optimization algorithm', Advances in Operations Research. Edited by I. Kacem, 2014, p. 208502. doi:10.1155/2014/208502.

[63] Kirkpatrick, S., Gelatt, C. D. and Vecchi, M. P. (1983). Optimization by simulated annealing. *science*, **220**(4598)**,** 671-680.

[64] Koushavand, B., Askari-Nasab, H. and Deutsch, C. V. (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)**,** 451-459.

[65] Krige, D. (1951). A statistical approach to some basic mine valuation problems on the witwatersrand. *Journal of the Chemical, Metallurgical and Mining Society of South Africa*, ***52*** (6), 119-139.

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

[67] Kumar, M., Husain, M., Upreti, N. and Gupta, D., (2010). Genetic algorithm: Review and application. *International Journal of Information Technology and Knowledge Management*, **2**, 451−454.

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

[69] Lerchs, H. and Grossmann, I. (1965). Optimum design of open pit mines. *Canadian Institute of Mining*, 58 (3), 47-54

[70] Louis, S. J. and Rawlins, G. J. (1991). *Designer Genetic Algorithms: Genetic Algorithms*. In Proceedings of 4th International Conference on Genetic Algorithms, Morgan Kaufmann, San Mateo, pp. 53-60.

[71] Lin, L. and Gen, M. (2009). Auto-tuning strategy for evolutionary algorithms: balancing between exploration and exploitation. *Soft Computing*, **13**(2), 157-168.

[72] Marinho de Almeida, A., (2013). Surface constrained stochastic life-of-mine production scheduling. MSc Thesis, McGill University, Montreal. Pages 119.

[73] Mathworks Inc. (2020). MATLAB Software. Ver. R2020a, Massachusetts, USA.

[74] Mbadozie, O. (2020). Incorporating Grade Uncertainty in Oil Sands Mine Planning and Waste Management Using Stochastic Programming. MASc Thesis, Laurentian University, Sudbury. Pages 142.

[75] Mbadozie, O., Ben-Awuah, E., and Maremi, A. (2022). A stochastic mixed integer linear programming framework for oil sands mine planning and waste management in the presence of grade uncertainty. *CIM Journal*, **13**(1), 16–37. https://doi.org/10.1080/19236026.2021.2024959.

[76] Michalewicz, Z. (1995a). *Genetic algorithms, numerical optimization, and constraints*. In Proceedings of 6th International Conference on Genetic Algorithms, Morgan Kauffman, San Mateo, pp. 151–158.

[77] Michalewicz, Z. (1995b). *A survey of constraint handling techniques in evolutionary computation methods*. In Proceedings of 4th Annual Conference on Evolutionary Programming, MIT Press, Cambridge, MA, pp. 135–155.

[78] Milani, G. (2016). A Genetic algorithm with zooming for the determination of the optimal open pit mines layout. *The Open Civil Engineering Journal*, **10**(1), 301-322.

[79] Mirjalili, S. (2019). Evolutionary algorithms and neural networks theory and applications. *Studies in Computational Intelligence* 780. Springer, Berlin. doi: 10.1007/978-3-319-93025-1 Available at: http://www.springer.com/series/7092. [accessed 18 Dec. 2020]

[80] Mirjalili, S., Song Dong, J., Sadiq, A. S. and Faris, H. (2020). Genetic Algorithm: Theory, Literature Review, and Application in Image Reconstruction. In: Mirjalili, S., Song Dong, J., Lewis, A. (eds) Nature-Inspired Optimizers. Studies in Computational Intelligence, vol 811. Springer, Cham. https://doi.org/10.1007/978-3-030-12127-3_5. [Accessed 18 Dec. 2020].

[81] Myburgh, C. and Deb, K. (2010). *Evolutionary algorithms in large-scale open pit mine scheduling categories and subject descriptors*. In Proceedings of 12th Annual Conference on Genetic and Evolutionary Computation, Portland, Oregon, USA, pp. 1155–1162.

[82] Nakano, R. and Yamada, T. (1991). *Conventional genetic algorithm for job shop problems*. In Proceedings of 4th International Conference on Genetic Algorithms, Morgan Kaufmann, San Mateo, CA, pp. 474-479.

[83] Oliver, I., Smith, D. and Holland, J. R. (1987). *Study of permutation crossover operators on the traveling salesman problem. Genetic algorithms and their applications*. In Proceedings of 2nd International Conference on Genetic Algorithms: July 28-31, Massachusetts Institute of Technology, Cambridge, MA, Hillsdale, NJ: L. Erlhaum Associates, pp. 224–230.

[84] Ota, R., Martinez, L. and R&O Analytics, (2017). *SimSched Direct Block Scheduler: A new practical algorithm for the open pit mine production scheduling problem*. In: 38th APCOM, August 2017, Golden, CO, USA [online]. [Viewed 29 May 2022]. Available from: https://www.researchgate.net/publication/320179762_SimSched_Direct_Block_Scheduler_A_new_practical_algorithm_for_the_open_pit_mine_production_scheduling_problem

[85] Paithankar, A. and Chatterjee, S. (2019). Open pit mine production schedule optimization using a hybrid of maximum-flow and genetic algorithms. *Applied Soft Computing*, 81, 105507, doi: 10.1016/j.asoc.2019.105507.

[86] Richardson, J. T., Palmer, M. R., Liepins, G. E. and Hilliard, M. R. (1989). *Some guidelines for genetic algorithms with penalty functions*. In Proceedings of 3rd International Conference on Genetic Algorithms (held in Arlington), San Mateo: Morgan Kaufmann, pp. 191–197.

[87] Ruiseco, J. R., Williams, J. and Kumral, M. (2016). Optimizing ore–waste dig-limits as part of operational mine planning through genetic algorithms. *Natural Resources Research,* **25**(4)**,** 473-485.

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

[89] Semenkin, E. and Semenkina, M. (2012). *Self-configuring genetic algorithm with modified uniform crossover operator.* In Proceedings of International Conference in Swarm Intelligence, Berlin, Heidelberg.: Springer, pp. 414-421.

[90] Senécal, R. and Dimitrakopoulos, R. (2020). Long-term mine production scheduling with multiple processing destinations under mineral supply uncertainty, based on multi-neighbourhood Tabu search. *International Journal of Mining, Reclamation and Environment,* **34**(7)**,** 459-475.

[91] Sharma, P. and Gargi, R. (2014). Performance analysis of different selection techniques in genetic algorithm. *International Journal of Science and Research,* **3**(8), 2042-2046. Available at: www.ijsr.net [accessed 18 Dec. 2020].

[92] 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)**,** 825-836.

[93] Sivanandam, S. N. and Deepa, S. N. (2007). *Principles of soft computing (2nd Edition)*. John Wiley & Sons.

[94] Sivanandam, S. N. and Deepa, S. N. (2008). Genetic algorithms. *Introduction to genetic algorithms.* Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 15–37. doi: 10.1007/978-3-540-73190-0_2.[accessed 18 Dec. 2020].

[95] Syswerda, G. (1991). Scheduling optimization using genetic algorithms. L Davis (Ed.), Handbook of Genetic Algorithms, Van Nostrand Reinhold, New York, pp. 332-349.

[96] Tabesh, M. and Askari-Nasab, H. (2011). Two-stage clustering algorithm for block aggregation in open pit mines. *Mining Technology,* **120**(3)**,** 158-169.

[97] Tolouei, K., Moosavi, E., Tabrizi, A. H. B., Afzal, P. and Bazzazi, A. A. (2020). An optimisation approach for uncertainty-based long-term production scheduling in open-pit mines using meta-heuristic algorithms. *International Journal of Mining, Reclamation and Environment,* **35(**2)**,** 115-140.

[98] Tsutsui, S., Yamamura, M. and Higuchi, T. (1999). *Multi-parent recombination with simplex crossover in real coded genetic algorithms*. In Proceedings of 1st Annual Conference on Genetic and Evolutionary Computation, 1, pp. 657-664.

[99] Vallejo, M. N. and Dimitrakopoulos, R. (2019). Stochastic orebody modelling and stochastic long-term production scheduling at the KéMag iron ore deposit, Quebec, Canada. *International Journal of Mining, Reclamation and Environment,* **33**(7)**,** 462-479.

[100] Villalba Matamoros, M. E. and Kumral, M. (2018). Calibration of Genetic Algorithm Parameters for Mining-Related Optimization Problems. *Natural Resources Research,* **28**(2)**,** 443-456.

[101] Weisstein, Eric W (2022). "NP-Hard Problem." From MathWorld--A Wolfram Web Resource. https://mathworld.wolfram.com/NP-HardProblem.html. [accessed 18 May. 2022].

[102] Zeleny, M., (1980). Multiple objectives in mathematical programming: Letting the man in. *Computers & Operations Research*, ***7***(1-2), pp.1-4.

[103] Zhang, Y., Cheng, Y. and Su, J. (1993). Application of goal programming in open pit planning. *International Journal of Surface Mining and Reclamation*, **7**(1)**,** 41-45.