International Journal of Mining and Geo-Engineering

IJMGE

IJMGE 51-1 (2017) 47–52

DOI: 10.22059/ijmge.2017.62152

A genetic algorithm approach for open-pit mine production scheduling

Aref Alipour [a,*], Ali Asghar khodaiari [a], Ahmad Jafari [a], Reza Tavakkoli-Moghaddam [b,c]

[a] *Department of Mining Engineering, College of Engineering, University of Tehran, Tehran, Iran*
[b] *Department of Industrial Engineering, College of Engineering, University of Tehran, Tehran, Iran*
[c] *LCFC, Arts et Métier ParisTech, Centre de Metz, France*

Article History:

Received 23 Mon 2016,
Revised 10 November 2016,
Accepted 11 November 2016.

ABSTRACT

In an Open-Pit Production Scheduling (OPPS) problem, the goal is to determine the mining sequence of an orebody as a block model. In this paper, linear programing formulation is used to aim this goal. OPPS problem is known as an NP-hard problem, so an exact mathematical model cannot be applied to solve in the real state. Genetic Algorithm (GA) is a well-known member of evolutionary algorithms that widely are utilized to solve NP-hard problems. Herein, GA is implemented in a hypothetical Two-Dimensional (2D) copper orebody model. The orebody is featured as two-dimensional (2D) array of blocks. Likewise, counterpart 2D GA array was used to represent the solution space of an OPPS problem. Thereupon, the fitness function is defined according to the OPPS problem objective function to assess the solution domain. Also, new normalization method was used to handle the block sequencing constraint. A numerical study is performed to compare the solutions of the exact and GA-based methods. It is shown that the gap between GA and the optimal solution by the exact method is less than % 5; hereupon GA is found to be efficient in solving OPPS problem.

Keywords : *Open-pit mine, production scheduling, metaheuristics and genetic algorithm*

1. Introduction

Open pit mine projects typically run for several decades and optimization of a strategic plans is a crucial element for successful planning of projects. A commonly used criterion for comparison of different strategic plans for extraction of valuable material from the ground is the net present value (NPV). The NPV optimization of strategic plans for open pit mines has a long history and has been approached from a number of different viewpoints.

Production scheduling in open-pit is a complex mine planning problem. Several authors by inspiration of operation research problems such as knapsack have developed linear and integer programming model for solving this problem. The Precedence-Constrained Knapsack Problem (PCKP) is an extension of the classic knapsack problem. PCKP Extension is carried out where the knapsack can be filled in multiple periods, known in mining engineering as the OPPS problem. In mining application, items are blocks in open-pit that should be scheduled for extraction in time. The knapsack constraint represents production capacity constraints such as mining and processing capacity, and precedence restrictions represent operational block access considerations (blocks can be extracted only if all blocks above and within a prescribed cone have been extracted before in time); individually, it can be shown that these constraints do not pose computational challenges. In combination, however, they produce an NP-hard problem.

In engineering applications, most of combinatorial problems are NP-hard. Normally the optimal solution of such problems cannot be obtained within an acceptable computation time. Therefore, approximation methods have to be utilized in order to practically answer the large instances of the problem. Sometimes the approximation methods are colloquially called heuristics. They normally act by building new solutions or improving the available solutions by using a set of problem-specific knowledge [2].

Metaheuristic optimization methods are a higher class of heuristic searching algorithms that are widely used for solving many of NP-hard combinatorial optimization problems. Several books and survey papers have been published on the subject. A hung part of the literature on metaheuristics is experimental in nature, describing empirical results based on computer experiments with the algorithms. There are a wide variety of metaheuristics that can be classified in the search strategy type including the local and global searches. Another classification dimension is single solution vs. population-based searches. Notable examples of metaheuristics include Genetic Algorithm (GA), Particle Swarm Optimization (PSO), Ant Colony (AC), Artificial Bee Colony (ABC), Simulated Annealing (SA), Tabu Search (TS), Variable Neighborhood Search (VNS) [3]. GA is a well-known global search metaheuristic method that uses population-based characteristics to improve the multiple candidate solutions.

Some of the metaheuristics are used in open-pit mining production scheduling problem. First time Denby and Schofield is used GA in optimization of open pit mine production scheduling [4]. Kumral and Dowd proposed SA-based algorithm to solve this problem [5]. Application of ACO has been proposed by Sattarvand for long-term open-pit production planning [6]. Lamghari and Dimitrakopoulos used tabu search and variable neighborhood descent algorithm [7] and also PSO has been utilized by Khan Niemann-Delius [8]. Recently, imperialist competitive algorithm (ICA) is proposed by Mokhtarianm and Sattarvand for long-term production scheduling of open pit mines [9].

The real-life instances of the OPPS problem by using liner programing are intractable for exact solver such as CPLEX; indeed, the industry-scale instances of the problem cannot be solved by standard solvers in a reasonable time. In this paper, an efficient genetic algorithm is used to represent an adaptive solution domain and to solve this problem. Penalty and normalization methods is used for the handling

* Corresponding author. Tel.: +982188020403. *E-mail address:* aref.alipour@gmail.com (A. Alipour).

the capacity and sequencing constraints, respectively. Numerical study is performed to compare the solutions of exact mathematical method and GA-based method in a hypothetical open pit mine that is typically represented by 2-D blocks.

2. Genetic algorithm

In artificial intelligence field, a genetic algorithm (GA) is a heuristic search that mimics the process of natural evolution. This heuristic (also sometimes called a metaheuristic) is routinely used to generate useful solutions to optimize and search the problems [1]. Genetic algorithms belong to the larger class of evolutionary algorithms (EA), which generate solutions to optimization problems using techniques inspired by natural evolution, such as inheritance, mutation, selection, and crossover. In a genetic algorithm, a population of candidate solutions (called individuals, creatures, or phenotypes) to an optimization problem is evolved toward better solutions. Each candidate solution has a set of properties (its chromosomes or genotype) which can be mutated and altered; traditionally, solutions are represented in binary as strings of zeros and ones, but other encodings are also possible.

The evolution usually starts from a population of randomly generated individuals and is an iterative process, with the population in each iteration called a generation. In each generation, the fitness of every individual in the population is evaluated; the fitness is usually the value of the objective function in the optimization problem being solved. The more fit individuals are stochastically selected from the current population, and each individual's genome is modified (recombined and possibly randomly mutated) to form a new generation. The new generation of candidate solutions is then used in the next iteration of the algorithm. Commonly, the algorithm terminates when either a maximum number of generations has been produced, or a satisfactory fitness level has been reached for the population. A typical genetic algorithm requires:

- A genetic representation of the solution domain,
- A fitness function to evaluate the solution domain.

A standard representation of each candidate solution is as an array of bits. Arrays of other types and structures can be used in essentially the same way. The genetic representation and the fitness function are defined, a GA proceeds to initialize a population of solutions and then to improve it through repetitive application of the mutation, crossover, inversion and selection operators [10]. More detail about GA is presented in the following sections.

2.1. Initialization

Initially, many individual solutions are (usually) randomly generated to form an initial population. The population size depends on the nature of the problem, but typically contains several hundreds or thousands of possible solutions. Traditionally, the population is generated randomly, allowing the entire range of possible solutions (the search space). Occasionally, the solutions may be "seeded" in areas where optimal solutions are likely to be found.

2.2. Selection

During each successive generation, a proportion of the existing population is selected to breed a new generation. Individual solutions are selected through a fitness-based process, where fitter solutions (as measured by a fitness function) are typically more likely to be selected. Certain selection methods rate the fitness of each solution and preferentially select best solutions. Other methods rate only a random sample of the population, as the former the process may be very time-consuming. The fitness function is defined over the genetic representation and measures the quality of the represented solution.

2.3. Genetic operators

The next step is to generate a second generation population of solutions from those selected through genetic operators: crossover (also called recombination), and/or mutation. For each new solution to be produced, a pair of "parent" solutions is selected for breeding from the pool that was selected previously. By producing a "child (also called offspring)" solution using the above methods of crossover and mutation, a new solution is created which typically shares many of its "parents" characteristics. These processes ultimately result in the next generation population of chromosomes that is different from the initial generation. Generally, the average fitness will increase by this procedure for the population. Although crossover and mutation are known as the main genetic operators, it is possible to use other operators such as regrouping, colonization-extinction, or migration in genetic algorithms [10].

2.4. Termination

This generational process is repeated until a termination condition has been reached. Common terminating conditions are [10]:

- A solution is found that satisfies minimum criteria
- Fixed number of generations reached
- Allocated budget (computation time/money) reached
- The highest ranking solution's fitness is reaching
- successive iterations no longer produce better results
- Manual inspection

3. Computational study for mine production scheduling

Mine production scheduling concentrates on determining a block extraction sequence in such a way that maximize NPV of the venture under access and capacity constraints. Deterministic mine production scheduling problem can be formulated using the following notation [11]:

Suppose, T is the number of periods, N is the number of blocks, $\mathrm{V_{ij}}$ is present value of block j in period i, $\mathrm{d_j}$ is the ore mass in block j, $\mathrm{A, A'}$ are the processing capacity, v_j is the waste mass in block j and $\mathrm{C, C'}$ are the mining capacity. Where x_{ij} is a binary variable and $x_{ij}=1$ means that block j is extracted in period i.

$$\max \sum_{i=1}^{T}\sum_{j=1}^{N} V_{ij} x_{ij} \tag{1}$$

Access constraint: In order to access a block to be extracted, overlying blocks should be extracted earlier or in the same period of the block under consideration. For example, in Figure 1 in order to extract the yellow blocks all the blue blocks should be extracted earlier or in the same period of the yellow blocks. The number of overlying blocks, which have to be extracted earlier, depends on upon slope angles ensuring the safety of walls. This constraint can be formulated by the following inequality,

$$\sum_{i=1}^{T} x_{ki} \geq \sum_{i=1}^{T} x_{ji} \tag{2}$$

$\forall \ j$ Blocks overlying block k.

Mining capacity: Depending on the selected equipment, financial power of the company, the stripping ratio, ore body characteristics and demand, a mining capacity is chosen.

$$C' \leq \sum_{j=1}^{N} (d_j + v_j) x_{ij} \leq C, \forall i = 1, \ldots, T \tag{3}$$

Processing capacity: In the mining operation, both ore and waste material are extracted because of the access constraint. While the waste material is sent to the dumps, the ore is sent to the processing plant. The amount of ore material should satisfy processing capacity.

$$A' \leq \sum_{j=1}^{N} d_j x_{ij} \leq A, \forall i = 1, \ldots, T \tag{4}$$

Block conservation constraint: This constraint ensures that a block can only be extracted once.

$$\sum_{i=1}^{T} x_{ij} \leq 1, \forall j = 1, \ldots, N \qquad (5)$$

In this study, open pit mine is typically represented by 2-D blocks. Figure 2 illustrated the economical block model of hypothetical copper as a scenario. Mining operation was considered to run for 4 years, and the maximum/minimum mining capacity was assumed to be 25/18 blocks a year and also maximum/minimum processing capacities was considered to be 15/9 ore blocks a year, respectively, and the discount rate was presumed to be 4%.

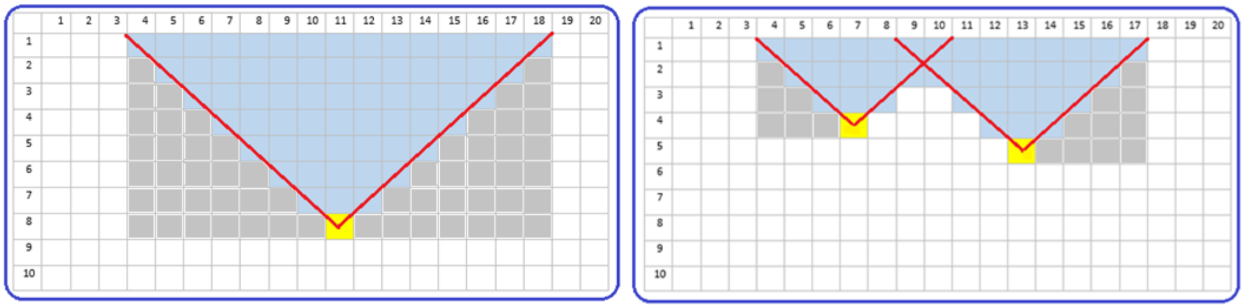
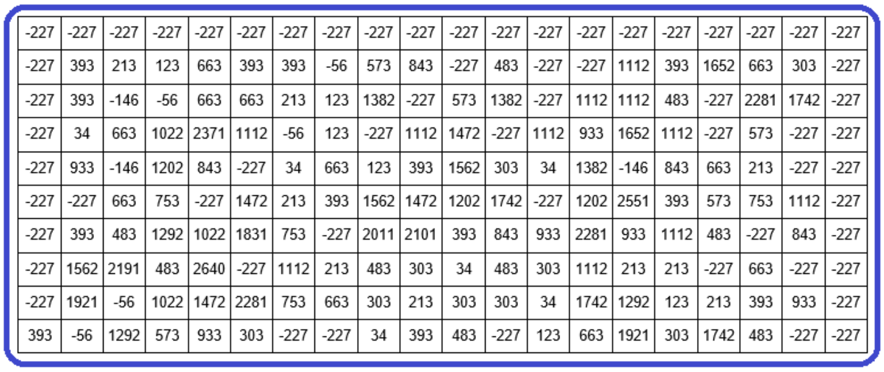


Figure1. The access constraint implies that the blue block should be extracted earlier or in the same period of the yellow block.



-227	-227	-227	-227	-227	-227	-227	-227	-227	-227	-227	-227	-227	-227	-227	-227	-227	-227	-227	-227
-227	393	213	123	663	393	393	-56	573	843	-227	483	-227	-227	1112	393	1652	663	303	-227
-227	393	-146	-56	663	663	213	123	1382	-227	573	1382	-227	1112	1112	483	-227	2281	1742	-227
-227	34	663	1022	2371	1112	-56	123	-227	1112	1472	-227	1112	933	1652	1112	-227	573	-227	-227
-227	933	-146	1202	843	-227	34	663	123	393	1562	303	34	1382	-146	843	663	213	-227	-227
-227	-227	663	753	-227	1472	213	393	1562	1472	1202	1742	-227	1202	2551	393	573	753	1112	-227
-227	393	483	1292	1022	1831	753	-227	2011	2101	393	843	933	2281	933	1112	483	-227	843	-227
-227	1562	2191	483	2640	-227	1112	213	483	303	34	483	303	1112	213	213	-227	663	-227	-227
-227	1921	-56	1022	1472	2281	753	663	303	213	303	303	34	1742	1292	123	213	393	933	-227
393	-56	1292	573	933	303	-227	-227	34	393	483	-227	123	663	1921	303	1742	483	-227	-227

Figure2. Economical block model of the hypothetical copper mine as a scenario.

4. A genetic solution representation for open-pit production scheduling

4.1. Regularization

The action of crossover and mutation operators normally deform the shape of the pits in generated mine schedules. This leads the resulting pits to violate access constraint and a regularization process is needed after each action. This method is known as normalization in research papers. First time Denby and co-authors [4] proposed their own normalization method, to ensure that the sequencing (access) constraints are satisfied. Thereafter, various normalization methods were applied to satisfy sequencing (access) constraint [5-6, 9]. Researchers have used exclusive styles with very few details given.

4.2. Implementation and evaluation

A genetic algorithm is used to solve the OPPS problem. Flowchart of the proposed procedure is demonstrated in figure 3. In the first step of algorithm, 20 random initial population is presented. Indeed a genetic representation of the solution domain consists of 200 blocks is generated, therefore, each population is a 10×20 matrix $P_{10\times 20}$ such that $P_{ij} = t$ means that block$_{ij}$ should be extracted in period t. An example of the initial random population used in the proposed GA is shown in figure 4, obviously, the initial populations do not satisfy the constraints. So each of them are normalized. In the normalization method, first we set $P_{ij} = 0$ if the block$_{ij}$ cannot be extracted because of the access constraint (slope angle and block precedency, for example in Figure 1 the black blocks cannot be extracted in any way); If $P_{ij} \succ 0$ then, if $P_{i-1j-1} = 0$ then $P_{i-1j-1} = 1$ also the same is done for P_{i-1j}, and P_{i-1j+1}. In the next step of normalization method, $P_{ij} = \max(P_{i-1j-1}, P_{i-1j}, P_{i-1j+1})$.

After this, we consider another constraint about the sum of the weights of blocks that can be extracted in each period. First, consider the blocks related to the first period ($P_{ij} = 1$), if they satisfy the constraints then nothing is done. Else, if the weights sum of the blocks in the first round is greater than the maximum mining capacity, then some of them are switched to 2, $P_{ij} = 2$, else if the the weights sum of blocks is less than the minimum mining capacity, then some of the other blocks are switched to 1 such that the above constraints remain satisfied. The same is done for other periods.

For the processing capacity constraint, a penalty method is considered. The value of the penalty of each period is calculated. Obviously, a feasible solution is the one with penalty equals to zero.

Next, crossover and mutation are done in 2000 iterations. In each iteration, one crossover and one mutation are done. In the crossover, two parents are chosen randomly from the population and crossed to generate two offspring. Then, these two offspring are normalized according to the normalization method. If these offspring were better than their parents, then they are substituted with the parents. Therefore, the population changes, but the number of population remains 20. In each mutation, a random population and some P_{ij} of that population are chosen randomly. Then, their values are changed with a random number between 0 and 4. Then this new offspring is normalized, if the new offspring is better than the initial one, it is substituted with the parent.

Note that an offspring is better than a parent if the value of the goal function of offspring is better than the parent and its penalty is less than the parent. After the iterations, the best solution is chosen among the populations. Other new iterations are done on this solution. In each of the new iterations, a random P_{ij} is chosen and if it is possible its value is changed to zero. In this approach, the penalty is reduced to zero. Then, the modified solution is reported. More details about proposed GA is presented in Table 1.

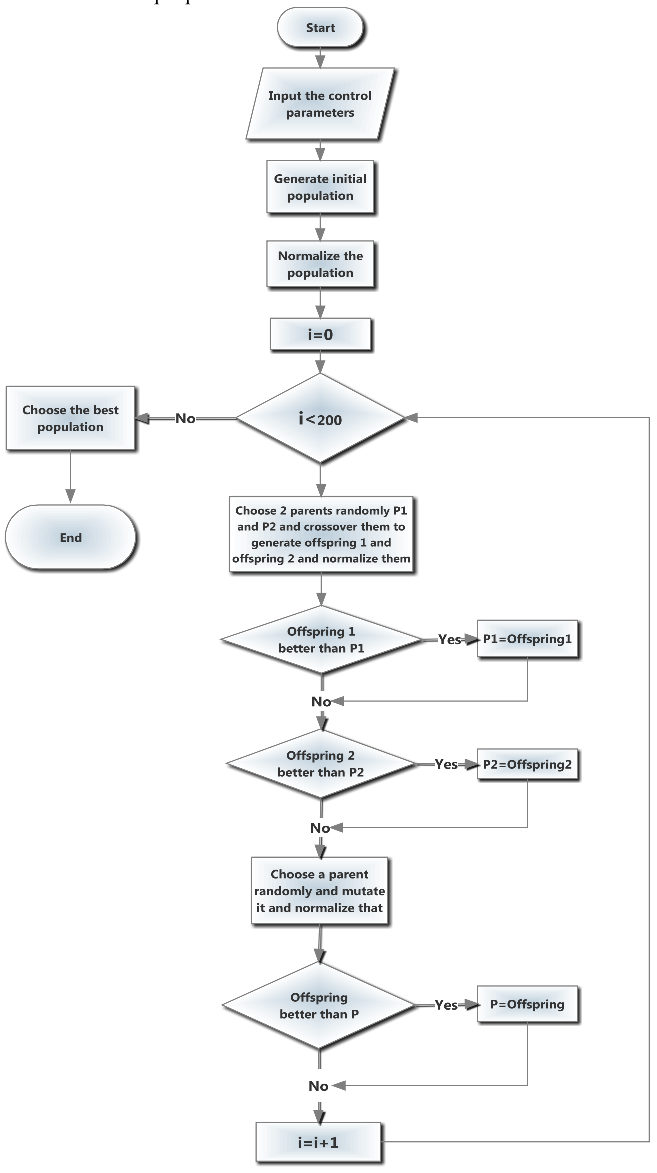
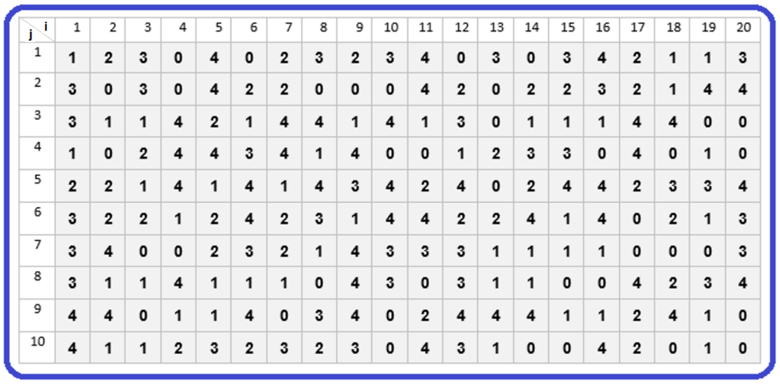

Start
Input the control parameters
Generate initial population
Normalize the population
i=0
i<200
No
Choose the best population
End
Choose 2 parents randomly P1 and P2 and crossover them to generate offspring 1 and offspring 2 and normalize them
Offspring 1 better than P1
Yes
P1=Offspring1
No
Offspring 2 better than P2
Yes
P2=Offspring2
No
Choose a parent randomly and mutate it and normalize that
Offspring better than P
Yes
P=Offspring
No
i=i+1

Figure 3. The flowchart of GA for open-pit mine production scheduling.



j \ i	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	1	2	3	0	4	0	2	3	2	3	4	0	3	0	3	4	2	1	1	3
2	3	0	3	0	4	2	2	0	0	0	4	2	0	2	2	3	2	1	4	4
3	3	1	1	4	2	1	4	4	1	4	1	3	0	1	1	1	4	4	0	0
4	1	0	2	4	4	3	4	1	4	0	0	1	2	3	3	0	4	0	1	0
5	2	2	1	4	1	4	1	4	3	4	2	4	0	2	4	4	2	3	3	4
6	3	2	2	1	2	4	2	3	1	4	4	2	2	4	1	4	0	2	1	3
7	3	4	0	0	2	3	2	1	4	3	3	3	1	1	1	1	0	0	0	3
8	3	1	1	4	1	1	1	0	4	3	0	3	1	1	0	0	4	2	3	4
9	4	4	0	1	1	4	0	3	4	0	2	4	4	4	1	1	2	4	1	0
10	4	1	1	2	3	2	3	2	3	0	4	3	1	0	0	4	2	0	1	0

Figure 4. An example of initial random population used in the modified GA.

Table 1: Details of proposed GA for OPPS problem.

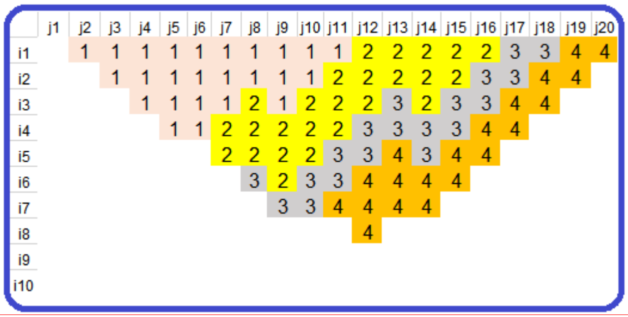
Table 1 : Details of proposed GA for OPPS problem.

Parameter	Description
Preparation method of Initial solution	Proposed method has used random initial solution according to a uniform distribution between [0, 4] scheduling period.
Selection method of generation	Elitism mechanism is used, new generation/solution is selected through a fitness-based process.
Population size	20
Multiple -point crossover rate	0.1
Mutation rate	0.05
Terminating criteria	Successive iterations no longer produce better results.
Optimum iteration	2000
Computational time	60 seconds

The results of production scheduling of numerical example solutions used by GA and CPLEX solver are illustrated in table 2. NPV's of CPLEX and GA solver were obtained 41565 and 39489 $, respectively in this research. The gap between the most solutions of GA and the exact solver is less than % 5. The number of calculated blocks for each period is listed in table 1. Mine block scheduling and sequencing by exact CPLEX solver and GA is demonstrated in Figure 5 and 6.

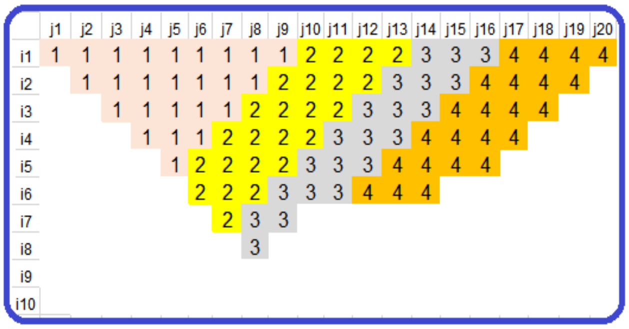
Table 2: Results of production scheduling solutions used by GA and CPLEX solver.

Calculated Item	Modified GA	CPLEX solver
NPV ($)	39489	41565
Number of blocks in ultimate pit limit	93	89
Number of blocks in the first period	25	25
Number of blocks in the second period	24	25
Number of blocks in the third period	21	19
Number of blocks in the fourth period	23	20



	j1	j2	j3	j4	j5	j6	j7	j8	j9	j10	j11	j12	j13	j14	j15	j16	j17	j18	j19	j20
i1		1	1	1	1	1	1	1	1	1	1	2	2	2	2	2	3	3	4	4
i2			1	1	1	1	1	1	1	1	2	2	2	2	2	3	3	4	4	
i3				1	1	1	1	2	1	2	2	2	3	2	3	3	4	4		
i4					1	1	2	2	2	2	2	3	3	3	3	4	4			
i5							2	2	2	2	3	3	4	3	4	4				
i6								3	2	3	3	4	4	4	4					
i7									3	3	4	4	4	4						
i8												4								
i9																				
i10																				

Figure 5. Production scheduling solutions using by CPLEX solver.



	j1	j2	j3	j4	j5	j6	j7	j8	j9	j10	j11	j12	j13	j14	j15	j16	j17	j18	j19	j20
i1	1	1	1	1	1	1	1	1	1	2	2	2	2	3	3	3	4	4	4	4
i2		1	1	1	1	1	1	1	2	2	2	2	3	3	3	4	4	4	4	
i3			1	1	1	1	1	2	2	2	2	3	3	3	4	4	4	4		
i4				1	1	1	2	2	2	2	3	3	3	4	4	4	4			
i5					1	2	2	2	2	3	3	3	4	4	4	4				
i6						2	2	2	3	3	3	4	4	4						
i7							2	3	3											
i8								3												
i9																				
i10																				

Figure 6. Production scheduling solutions using by GA.

5. Conclusion

The real-life instances of the OPPS problem are intractable for exact solvers such as CPLEX. In this paper, an efficient GA approach was presented for an OPPS problem.

GA coding is used to represent an adaptive solution domain to solve this problem. Also, a new normalization method was used for handling the block sequencing constraint. The good efficiency of the new coding method and the applied normalization mechanism demonstrated it to be a satisfaction for sequencing constraint.

To test the computational efficiency of the proposed GA, a hypothetical case study was run by the proposed heuristic and CPLEX exact solver. Numerical validation showed that the GA can obtain promising results.

The CPU time for this efficient GA to derive optimal solutions was relatively short given to the complexity of the OPPS problem. In other words, the capability of GA in obtaining near-optimal scheduling solution with a relatively small gap (less than % 5) is evident. Therefore, the proposed framework would be more practically useful for solving OPPS problem in large scale mines.

REFRENCES

[1] Gleixner, A.M. (2009). Solving large-scale open pit mining production scheduling problems by integer programming. Master's Thesis, Technische Universität Berlin, Zuse Institute

Berlin (ZIB).

[2] Dorigo, M.S. T.(2004). Ant Colony Optimization, Cambridge: MIT Press.

[3] Talbi, E.-G. (2009). Metaheuristics: from design to implementation, John Wiley & Sons.

[4] 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.

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

[6] Sattarvand, J. and Niemann-Delius, C. (2012). Long-term open-pit planning by ant colony optimization, Lehrstuhl für Rohstoffgewinnung über Tage und Bohrtechnik und Institut für Bergbaukunde III.

[7] 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.

[8] Khan, A. and Niemann-Delius, C. (2014). Production Scheduling of Open Pit Mines Using Particle Swarm Optimization Algorithm. Advances in Operations Research 2014.

[9] Mokhtarian Asl, M. and Sattarvand, J. (2016). An imperialist competitive algorithm for solving the production scheduling problem in open pit mine. Int. Journal of Mining & Geo-Engineering 50(1), 131-143.

[10] Whitley, D. (1994). A genetic algorithm tutorial. Statistics and computing 4(2), 65-85.

[11] Kumral, M. (2010). Robust stochastic mine production scheduling. Engineering Optimization 42(6), 567-579.