Shahrood University of Technology

Iranian Society of Mining Engineering (IRSME)

Journal of Mining and Environment (JME)

journal homepage: www.jme.shahroodut.ac.ir

THOMSON REUTERS

Ultimate Pit Limit Optimization using Boykov-Kolmogorov Maximum Flow Algorithm

Akisa David Mwangi[1, 2*], Zhang Jianhua[1], Huang Gang[1], Richard Muthui Kasomo[1], and Matidza Mulalo Innocent[1]

1. *School of Resources and Environmental Engineering, Wuhan University of Technology, Wuhan, Hubei, China*
2. *Mining, Materials and Petroleum Engineering Department, Jomo Kenyatta University of Agriculture and Technology, Nairobi, Kenya*

Article Info

Received 13 October 2020

Received in Revised form 06 December 2020

Accepted 10 December 2020

Published online 10 December 2020

DOI:10.22044/jme.2020.10170.1953

Keywords

Boykov-Kolmogorov algorithm

Maximum flow

Pseudo-flow

Push-Relabel

Ultimate pit limit

Abstract

The ultimate pit limit optimization (UPLO) serves as an important step in the mine planning process. Various approaches of maximum flow algorithms such as pseudo-flow and push-relabel have been used for pit optimization, and have given good results. The Boykov-Kolmogorov (BK) maximum flow algorithm has been used in solving the computer vision problems and has given great practical results but it has never been applied in UPLO. In this work, we formulate and use the BK maximum flow algorithm and the push-relabel maximum flow algorithm in MATLAB Boost Graph Library within the MATLAB software in order to perform UPLO in two case studies. Comparing both case studies for the BK maximum flow algorithm and push-relabel maximum flow algorithm gives the same maximum pit values but the BK maximum flow algorithm reduces the time consumed by 12% in the first case and 16% in the second case. This successful application of the BK maximum flow algorithm shows that it can also be used in UPLO.

1. Introduction

Over the years, there have been developments on solving the open pit optimization problem in the mining industry. This can be dated back from the manual methods [1] that were used, to the current mathematical algorithms applied in the modern computer technology [2]. Since the mining industry is capital-intensive [3] and operates with resources that are a product of mineral resource estimation of the geological block model [4], optimization of every operation is of essence so as to maximize the overall revenue from the mine. In the quest for maximizing the recovery of these ever-decreasing resources, optimization of these mining operations should be done safely [5]. Open-pit optimization plays a major role since it acts as an initial stage that assists in planning other major operations such as setting out the layout of the mine, determining the size of the processing plant and the mine equipment, and scheduling of the mining activities [6-8]. Owing to the vital role played by the Ultimate Pit Limit Optimization (UPLO) in the mining industry, this work focuses on the algorithms used for UPLO and gives an alternative method that can be applied in order to solve the UPLO problem. Some of the abbreviations used in this work are listed in Table 1.

Corresponding author: davidmwas06@gmail.com (A.D. Mwangi).

Table 1. List of some abbreviations used in this work and their meanings.

Abbreviation	Meaning
BK	Boykov-Kolmogorov
UPLO	Ultimate pit limit optimization
LG	Lerchs-Grossmann
EBM	Economic block model
EBVs	Economic block values
MatlabBGL	MATLAB Boost Graph Library

2. Literature Review

The UPLO methods using the mathematical models were introduced by Lerchs–Grossmann (LG) in the 1960s when they developed the graph theory [9, 10]. In their work, the main idea was to convert the block model into a directed tree such that the blocks represented the vertices of the graph and the directed arcs represented the relationship between the blocks with respect to the slope constraints of the pit. This algorithm was widely accepted over the years and was adopted in different mining softwares, and it is still being applied to date [2] but sometimes proves difficult to program and can consume a lot of time, especially with huge block models [11]. LG also introduced the 2D dynamic programming ultimate pit limit algorithm, although required improvement since the real mines are in 3D [12]. The 2D dynamic programming ultimate pit limit algorithm on the other hand was improved to 3D in a bid to overcome the flows from the 2D but still failed to give the optimal results [13]. The network flow for ultimate pit limit was introduced in 1968 [14], where a bipartite network was formed from the 3D block model. This formed the basis of application of maximum flow algorithms in UPLO. The moving cone algorithm in UPLO was also introduced [15], and was also widely accepted due to its simplicity, though it is a heuristic method. Improvements have also been made on the moving cone algorithm in an endeavour to make it more efficient by various researchers [16-19]. The floating cone II was introduced [20], and was later advanced to modified floating cone II [21]. The modified floating cone II was also later improved coming up with the floating cone III method [19]. Korobov also developed a cone-based method in order to solve UPLO but failed to give a true optimum solution [22]. This work was later improved in trying to eliminate some of the drawbacks of Korobovs' algorithm [22]. These cone-based algorithms have been incorporated in the mining industry in some of the optimization softwares [23] since they are simple to understand. One of their main drawbacks is that they are heuristic, and thus do not necessarily give the optimal results but rather close to the optimal results [24].

The UPLO problem has continued to be addressed by various researchers so as to improve the existing algorithms, and some of the proposed algorithms are as outlined in this work. The Ford Fulkerson algorithm was used in order to solve UPLO but in a 2D model [25]. The "one three–one two (13–12)" algorithm has also been proposed for UPLO but also in 2D [26]. The stochastic approach has also been applied successfully in UPLO [27]. A new network optimization method for solving UPLO has been introduced [28]. The Best Positive Inverted Truncated Cone (BPITC) algorithm for UPLO has also been introduced [29]. An algorithm in UPLO that seek to search for the largest pit with non-negative value in its objective function has also been developed [7]. The Genetic Algorithm (GA) [30] and the Parallel Genetic Algorithm (PGA) [31] for solving the UPLO problem have also been proposed. The use of artificial neural network in 3D was also proposed for UPLO [24]. Some maximum flow algorithms have also been successfully used in UPLO such as the Push-Relabel maximum flow [32, 33] and the pseudo-flow maximum flow [32]. Table 2 shows the trend of some of the UPLO methods from 1960s.

Time is a major factor involved in the mine industry since it helps in planning and performing different tasks within the shortest time possible. Maximum flow algorithms give optimal results since they are rigorous but most of them have not been incorporated in UPLO. Some of them such as the pseudo-flow maximum flow have been seen to immensely save on time when compared to the most commonly used Lerchs–Grossmann graph algorithm [11]. This paves the way for more research works to be done on the maximum flow algorithms with respect to their application in UPLO. This work, therefore, focuses on the maximum flow methods in relation to UPLO, and provides an alternative maximum flow algorithm to be applied in UPLO.

Table 2. A trend of some UPLO methods.

Period	UPLO Methods Used
Before 1960	Manual methods
1960-1970	Lerchs-Grossmann 2D dynamic programming
	Lerchs-Grossmann graph theory
	Network flow analysis
	Moving cone algorithm
	Linear programming
1970-1980	Parametric analysis
	Korobov algorithm
	Improved moving cone
	Maximum closure
1970-1990	3D dynamic programming
	Best-valued cross-section algorithm
	Modified tree graph algorithm
1990-2000	Corrected Korobov algorithm
	Improved graph theory
	Moving cone II
	Transportation algorithm
	Interactive parameterization techniques
	Artificial neural network
	Push-relabel maximum flow algorithm
2000-2010	Stochastic process
	Pseudo-flow maximum flow algorithm
	Floating slopes method
	Modified moving cone II
	Real option approach
	Best positive inverted truncated cone
2010-Date	Ford Fulkerson algorithm in UPLO
	Floating cone method III
	Network optimization
	One three–One two technique
	New graph-based algorithm
	Genetic algorithm
	Parallel genetic algorithm
	New artificial neural network

3. Maximum Flow Algorithms

These are network flow problems that maintain a feasible flow in the network and aim at producing a maximum flow from the source to the sink of the network. In UPLO, the maximum flow theory was introduced in 1968 by Johnson [14], whereby he transformed the 3D block model into a bipartite network. The converted block model was then solved as a network flow problem in order to obtain the maximum flow. In the bipartite network, every block in the block model is symbolized by one node. An imaginary dummy source *(s)* and all the ore blocks *(O)* are placed on one side, while an imaginary dummy sink *(t)* and all the waste blocks *(W)* are kept on the other side. Depending on the requirements of the slope, the ore blocks are connected to each waste block that must be mined so as to mine the ore block. After this connection, all the waste blocks are then connected to the sink node, while all the ore blocks are connected to the dummy source node [14]. The direction of the flow is determined by the direction of the arc, as shown in Figure 1. The arc capacities differ depending on the location of the arc. There is an unlimited capacity for the dummy source*(s)* to send; unlimited capacity for the dummy sink *(t)* to receive; arcs linking the ore blocks and the source node have capacities that correspond to the ore block values; arcs linking the sink node and the waste blocks have capacities that corresponds to the waste block values; and finally, arcs linking the waste blocks and the ore blocks have boundless capacities since any waste block that restricts an ore block to be mined has to be mined out at any cost so that the ore block can be mined as well [12].

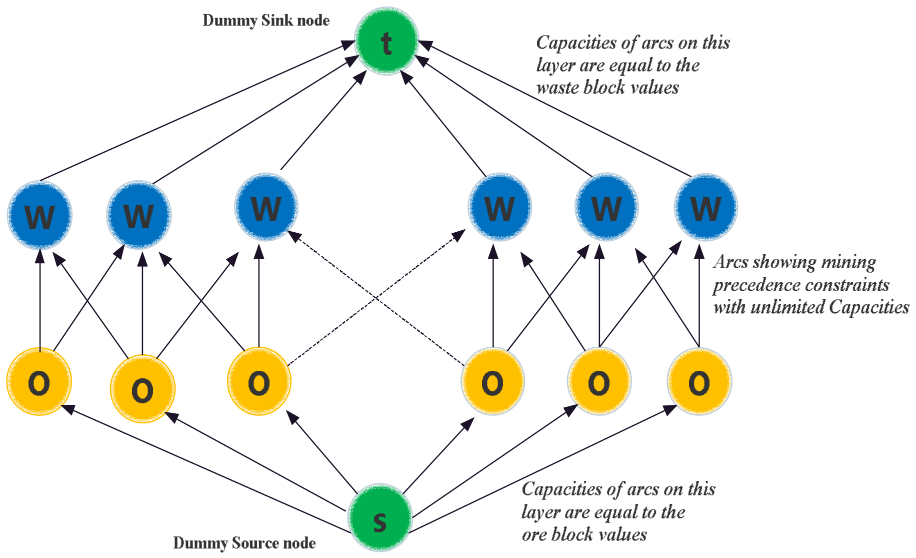

Dummy Sink node
t
Capacities of arcs on this layer are equal to the waste block values
W
O
Arcs showing mining precedence constraints with unlimited Capacities
s
Dummy Source node
Capacities of arcs on this layer are equal to the ore block values

Figure 1. Pit limit network design.

According to Verma and Batra [34], there are three main techniques of maximum flow algorithms:

3.1. Augmenting-Path

This method makes sure that the flow-conservation constraints and the capacity constraints are adhered, and thus maintain a feasible flow during the implementation of the algorithm. The algorithms in the augmenting path are normally primal feasible. The algorithm iteratively looks for source *(s)* to sink *(t)* paths with residual capacities pushing the arcs with least capacity until no more source *(s)* to sink *(t)* paths are found. Some of the augmenting path algorithms are the Dinic's algorithm [35] and the Boykov-Kolmogorov maximum flow algorithm [36].

3.2. Push-Relabel

This method conserves the capacity constraints but does not necessarily adhere to the flow-conservation constraints, and thus does not maintain a feasible flow during the implementation of the algorithm. The algorithm allows for an excess flow into a node called *surplus* thus has a flow called *preflow* [34]. Unlike the augmenting path, push-relabel is dual feasible [34]. As the name suggests, the algorithm works by pushing and relabelling the nodes. The source node is given a fixed label of zero '0', while the sink node is labelled 'n' representing the number of nodes and is also fixed. Initially, all the other nodes are also set as zero '0' and all the arcs from the source node to ore blocks are saturated. During the push process, the nodes with an excess flow push the flow to a neighbouring node that has a minor label, and if such a node does not exist, then that node is relabelled with an increase of one '1' to its value. This phase terminates when all the nodes with excess flow have labels greater than 'n'. The next phase then redirects the excess flow to the source node, thus converting the *preflow* into a maximum flow [34].

3.3. Hochbaum's Pseudo-flow

Just as the push-relabel algorithm, pseudo-flow conserves capacity constraints but does not necessarily adhere to the flow-conservation constraints, and thus does not maintain a feasible flow during the implementation of the algorithm and is also dual feasible. Unlike the push-relabel algorithm, pseudo-flow allows for an excess flow as well as deficit at the nodes, thus having a flow called *pseudo-flow* [37]. Initially, in this algorithm, all arcs coming from the source and those entering the sink are all saturated. A forest with numerous components is then induced by making sure that there are no cycles in the free arcs and the roots of these components are nodes with strict *surplus* or *deficit*. Iterations in the algorithm are done by searching for residual arcs from a node with *surplus* to one with *deficit* until none is found. The next phase then converts the pseudo-flow into a maximum flow [37].

In UPLO, the push-relabel and pseudo-flow maximum flow algorithms have made their applications. Pseudo-flow has been tested by various researches, who have given better results when compared to the Lerchs–Grossmann algorithm [2, 11, 37]. These maximum flow algorithms have been incorporated in some of the commercial mining optimization softwares. Push-relabel has been applied in Minemax's Scheduler [32], Mincom [33], etc. Pseudo-flow has been applied in Deswik, and has also been recently implemented in the Gemcom's Whittle optimization software [11]. From these adoptions into the optimization software, it can be seen that maximum flow has greatly improved the UPLO problem, solving and thus giving room for more research works to be done. This work looks at the augmenting path maximum flow option by applying the BK algorithm in UPLO since it has also proved to have better results in other applications such as the computer vision problems [34, 36].

4. Boykov-Kolmogorov (BK) Maximum Flow Algorithm

The BK algorithm was developed to improve the augmenting paths work in computer vision and proved to give better results experimentally when compared to other algorithms [36]. With the augmenting paths, a new breath-first search is normally started from the source (s) to sink (t) paths once all the pre-examined paths are exhausted. This search in UPLO can be achieved by scanning the blocks (vertices) within the block model. Since it can be very costly to repeat the whole search process every time, the BK algorithm builds two search trees, where one starts from the source and the other from the sink. The algorithm also reuses these trees in their search instead of starting afresh, thus saving on time [36]. The algorithm works by maintaining two search trees T and S, which are rooted at the sink node (t) and the source node (s), respectively. Moving away from the root node, the next node becomes the *child* of the previous node *parent*. At the beginning of the algorithm, the edges from the *parent* to the *children* in tree S are not saturated, and likewise, to the edges from the *children* to the *parent* in root T. At any given step, a node can either be in tree S, T or can be *free*, as shown in Equation (1).

$$T \subset N,\ \ t \in T,\ \ S \subset N,\ \ s \in S,\ \ S \cap T = \emptyset \qquad (1)$$

The nodes within the trees can be regarded as *active* or *passive* depending on their location, as shown in Figure 2.

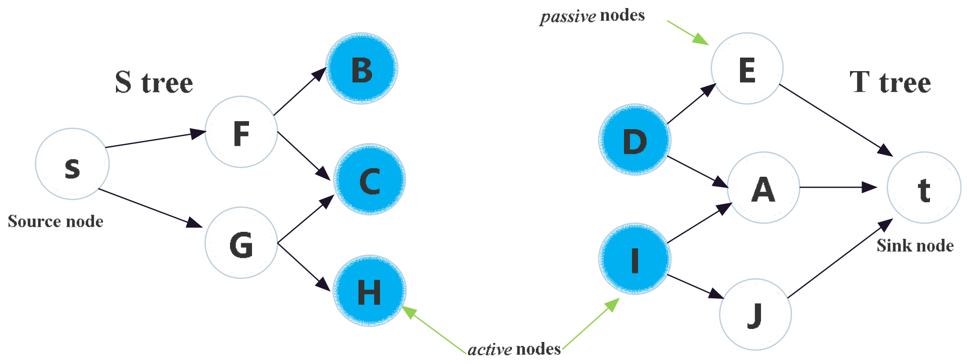


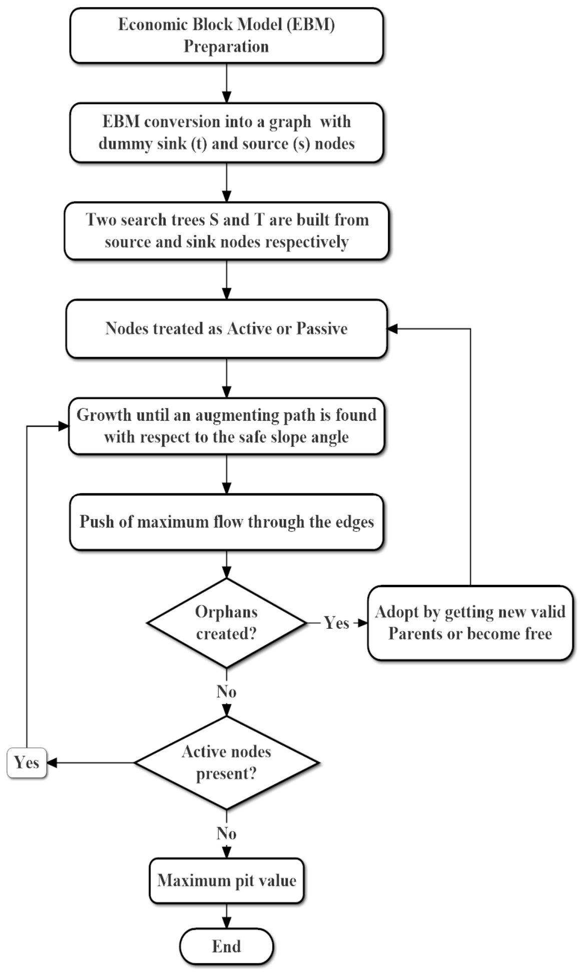
Figure 2. Active and passive nodes in the BK algorithm.

The *active* nodes are located at the edges of the trees since they are the ones that link to the new *free* nodes. Once they scan and link to the *free* nodes, they seize to be active, while the new linked nodes become the *active* nodes. Once an active node from one tree comes into contact with the *active* nodes from the other tree, an augmenting path is created. The algorithm iterates three main steps:

Growth step: this is the step where the tree grows by linking the active nodes in the tree to the free nodes whose edges are not saturated until an augmenting path is found. In UPLO, the blocks will be linking depending on the set of blocks that are required to be mined to pave the way for mining a certain ore block. The growth step terminates when an augmenting path from the source to the root is found. This is achieved by linking the *active* nodes of the two set of trees S and T.

Augmentation step: this step enhances the augmenting paths found in the growth step by trying to push maximum flow through the edges such that some edges becomes saturated. The saturation makes the edges invalid, thus rendering the nodes preceding the *orphans*. This then creates another set of trees whose roots are the *orphans* created in this step.

Adoption step**:** with creation of a forest of trees from the augmenting step, the adoption step restores the original set up of S and T trees. This is achieved by looking for new *parents* with non-saturated edges for the *orphans* from the same tree they has come from. If the *orphans* do not get new valid *parents*, they become free nodes. This step ends when all the *orphans* cease to exist and the original S and T trees are left. Once the adoption step ends, the algorithm starts again at the growth step until the time when all the *active* nodes phase out, thus achieving a maximum flow. A flowchart for the steps in the BK algorithm that is modelled for UPLO is shown in Figure 3.



Economic Block Model (EBM) Preparation
EBM conversion into a graph with dummy sink (t) and source (s) nodes
Two search trees S and T are built from source and sink nodes respectively
Nodes treated as Active or Passive
Growth until an augmenting path is found with respect to the safe slope angle
Push of maximum flow through the edges
Orphans created?
Yes
Adopt by getting new valid Parents or become free
No
Active nodes present?
Yes
No
Maximum pit value
End

Figure 3. The BK maximum flow algorithm flowchart.

4.1. 2D Illustration of BK Maximum Flow in Pit Optimization

If we consider a 2D block model shown in Figure 4, the BK maximum flow algorithm can be applied in pit limit optimization in order to calculate the maximum pit value. The blocks are labelled from A to J and their Economic Block Values (EBVs) given at the centre of each block. EBVs represent the values of the individual blocks in the economic block model. EBVs are calculated by considering the revenue from selling a single block and the cost of mining that block, as shown in Equation (2). This assists in the identification of the waste blocks (if EBV is negative or zero) and the ore blocks (if EBV is positive).

$$EBV = Revenue\,(block_i) - Cost\,(block_i) \quad (2)$$

A -2	B -1	C -1	D -1	E -2
F 0	G 1	H 3	I 6	J 0

Figure 4. A 2D block model.

The first step is to convert the block model to a graph and introduce the source node (s) and the sink node (t), both of which will act as the root nodes for trees S and T, respectively, as shown in Figure 5. The algorithm starts with making the root nodes active while the other nodes are *free*.

The algorithm then scans by looking for *free* nodes adjacent to the *active* nodes. The *active* nodes are then linked to the neighbouring *free* nodes, which are then activated, and the previous active nodes become passive, as shown in Figure 6.

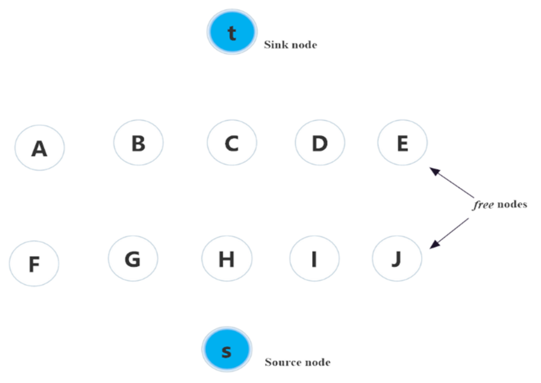


Figure 5. Graph representing the block model.

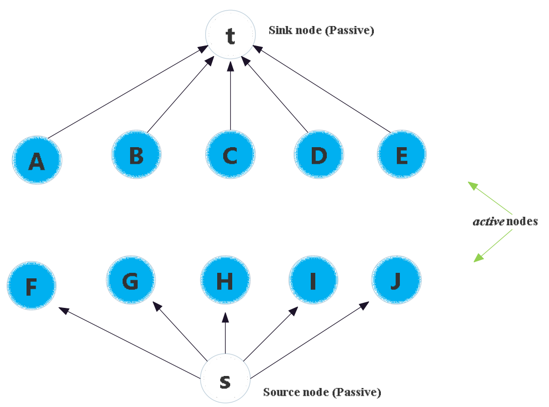


Figure 6. Graph showing the active nodes.

Assuming a maximum slope angle of 45°, the *active* nodes link the *free* neighbouring nodes, which in this case are the waste blocks that have to be mined for the ore block to be mined. The growth continues until the two trees join, as shown in Figure 7.

Since the BK maximum flow algorithm is an augmenting path method, it makes sure that the flow-conservation constraints and the capacity constraints are adhered to, such that:

i. The flow entering the ore block *i* from the source node *s* is equal to the total sum of the flow going out of the ore block to the overlying waste blocks (Equation 3), while the total sum of flow entering a waste node *j* is equal to the flow from the waste block to the sink node *t* (Equation 4).

ii. An edge $a_{ij\in A}$ cannot carry more than its capacity (Equation 5).

$$f_{si} - \sum_{j\in N} f_{ij} = 0 \quad for \quad each\ i \in N \quad (3)$$

$$\sum_{i\in N} f_{ij} - f_{jt} = 0 \quad for \quad each\ j \in N \quad (4)$$

$$f_{ij} \le C_{ij} for \quad each \quad a_{ij} \in A \quad (5)$$

where:

N	= Set of nodes within the graph representing the blocks within the economic block model;
A	= Set of arcs linking the nodes within the graph representing the relationship between the blocks;
a_{ij}	= An arc that links the ore blocks to the overlying waste blocks depending on the overall slope angle;
f_{si}	= Flow from the source node to the ore block;
f_{ij}	= Flow from the ore block to the waste block;
f_{jt}	= Flow from the waste block to the sink node;
C_{ij}	= Maximum capacity of flow through the arc.

The algorithm then augments the paths created by trying to push maximum flow through the edges such that some edges become saturated. This can result in the creation of *orphans* that have to be adopted by giving them new *parents* that have valid edges. The iteration process continues until the active nodes cease to exist, thus achieving a maximum flow, as shown in Figure 8.

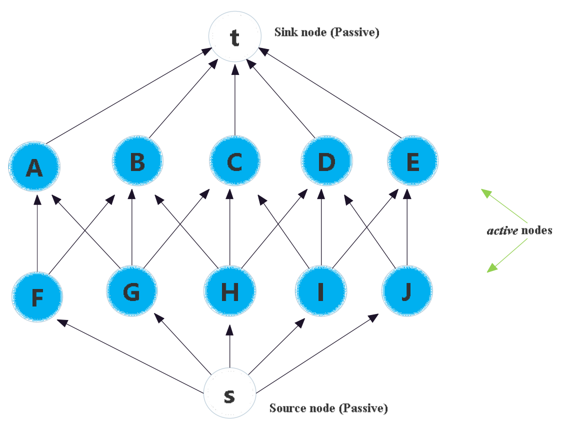


Figure 7. Graph showing the augmenting paths from source (s) to root (t).

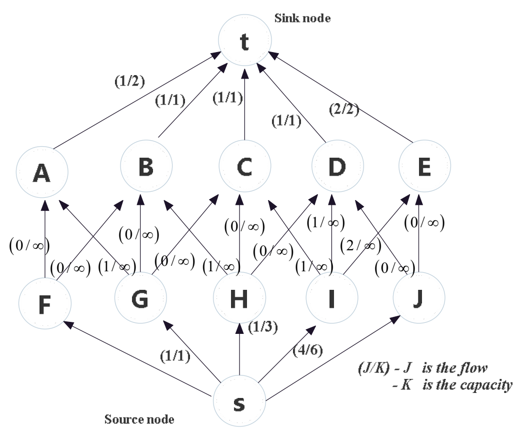


Figure 8. Maximum flow though the network.

The maximum pit value can be realized by separating the trees S and T using the invalid parts created by the saturated edges (in blue) linking the two trees. The saturated edges from the source node to the ore blocks are normally broken in order to avoid the support of the ore blocks to the overlying waste blocks, as shown in Figure 9. The other saturated edges from the waste blocks to the sink node are also broken in order to avoid the support of these waste blocks from the underlying ore blocks. This also makes sure that there are no outgoing arcs from the maximum closure. In this case, the ultimate pit contains the blocks B-C-D-E-H-I with a maximum pit value of 4, as shown in Figure 10. Trees S and T are marked in red and green, respectively.

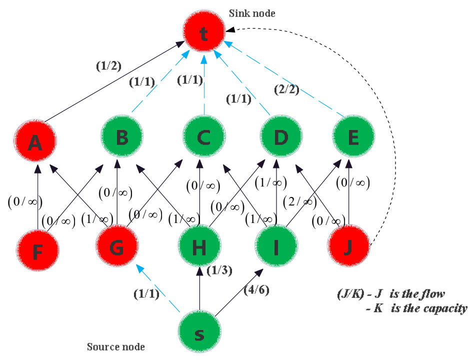


Figure 9. Network showing the saturated edges.

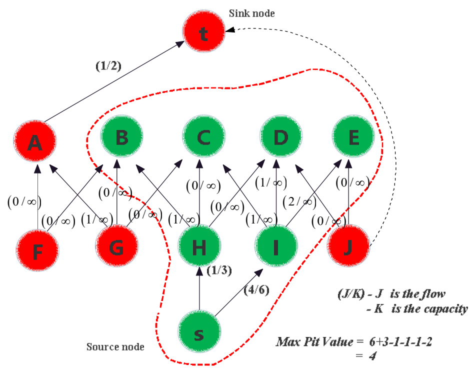


Figure 10. Maximum pit value from S tree.

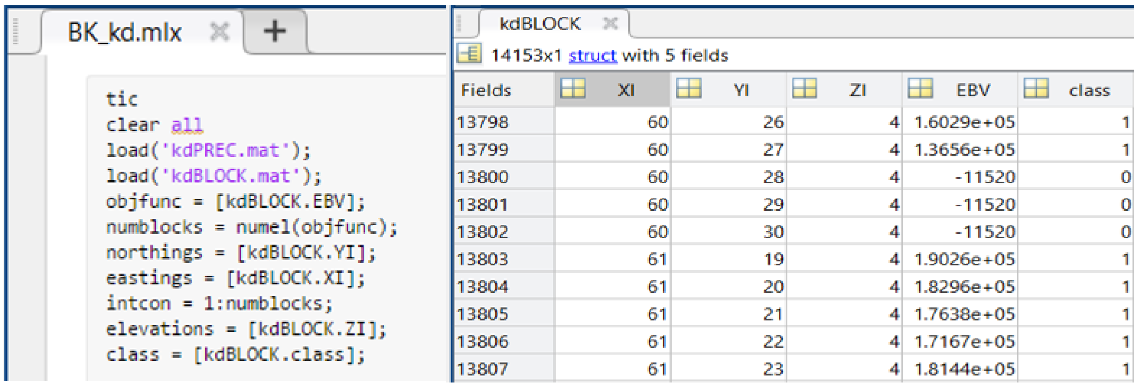
5. Application of BK Maximum Flow to Case Studies

BK maximum flow algorithm was then applied in the mining case studies in order to ascertain its applicability in the UPLO problem. The MATLAB environment was used in order to formulate a suitable model for application in UPLO. The MATLAB Boost Graph Library (MatlabBGL) package [38] was used since it contains different data structures as well as algorithms [39] that boost the graph library of the MATLAB environment. The BK maximum flow and push-relabel maximum flow are some on the algorithms contained in MatlabBGL [38]. They were well-formulated in this work to suit the UPLO problem. The implementation and testing of the BK maximum flow algorithm on the case studies was done on an Intel® Core™ i5-6200 CPU Dell computer with 16 GB of RAM.

The datasets used in the case studies came from Minelib (http://mansci-web.uai.cl/minelib/) [40], which is a public library that deals with open-pit mining-related problems. Minelib provides the datasets as well as their best-known solutions for ultimate pit limit and scheduling. Minelib uses the Hochbaum's Pseudoflow algorithm in order to solve the UPLO problem [40].

5.1. Case 1: Arizona's Copper Deposit (KD)

The dataset from the Arizona's copper deposit, classified as 'KD' in Minelib, was used in the application of BK maximum flow algorithm in UPLO. The block model contained 14,153 blocks that were 20 x 20 x 15 m in size. The block indices were provided so as to know the location of the blocks within the block model. The precedence blocks were computed at 45 degrees with 8 levels giving a total of 219,778 precedences. EBVs were also provided in order to help in calculating the ultimate pit value. Figure 11 shows a section of the MATLAB code used to input the data for this deposit and also a section of the data after it was imported in MATLAB. The data included the block indices, i.e. Easting (XI), Northing (YI), and Elevation (ZI); EBVs, and the class for distinguishing between the waste and ore blocks. The ore block precedencies were also loaded in a separate file.



```
tic
clear all
load('kdPREC.mat');
load('kdBLOCK.mat');
objfunc = [kdBLOCK.EBV];
numblocks = numel(objfunc);
northings = [kdBLOCK.YI];
eastings = [kdBLOCK.XI];
intcon = 1:numblocks;
elevations = [kdBLOCK.ZI];
class = [kdBLOCK.class];
```

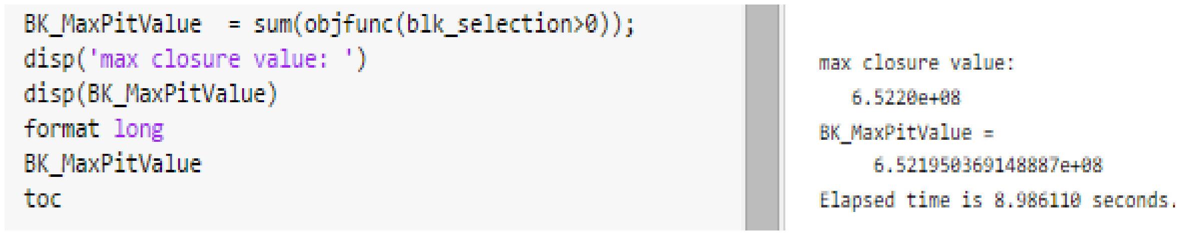
Fields	XI	YI	ZI	EBV	class
13798	60	26	4	1.6029e+05	1
13799	60	27	4	1.3656e+05	1
13800	60	28	4	-11520	0
13801	60	29	4	-11520	0
13802	60	30	4	-11520	0
13803	61	19	4	1.9026e+05	1
13804	61	20	4	1.8296e+05	1
13805	61	21	4	1.7638e+05	1
13806	61	22	4	1.7167e+05	1
13807	61	23	4	1.8144e+05	1

Figure 11. Sections of the input code and data used for the 'KD' deposit.

The ultimate pit limit for the copper deposit was calculated using the BK maximum flow algorithm and push-relabel maximum flow; the latter is already being applied in the mining industry [32]. The BK maximum flow algorithm gave a maximum pit value of $652,195,037 within a timeframe of 8.99 s. The push-relabel maximum flow also gave the same maximum pit value but with a longer time of 10.56 s. These two methods concurred with the maximum pit value [40] given in Minelib that used the pseudo-flow maximum flow algorithm. The BK maximum flow algorithm, in this case, was faster than the push-relabel maximum flow. Table 3 gives a summary of the Arizona's copper deposit (KD) and the results of using the two maximum flow methods. The MATLAB output of the maximum pit value obtained using the BK maximum flow algorithm for this deposit is shown in Figure 12.

Table 3. A summary of the Arizona's copper deposit (KD).

	Blocks	Block size (m)	Precedences	Max. pit value ($)	Time (s)
BK	14,153	20 x 20 x15	219,778	652,195,037	8.99
Push-relabel	14,153	20 x 20 x15	219,778	652,195,037	10.56



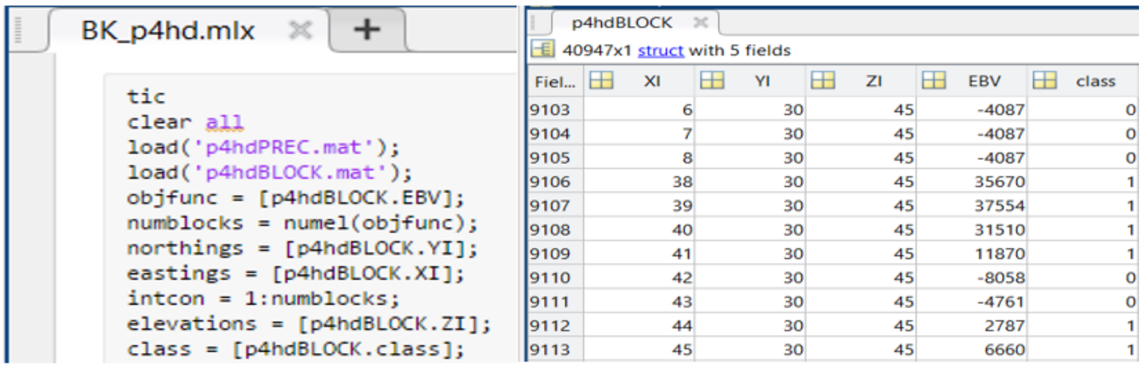
```
BK_MaxPitValue  = sum(objfunc(blk_selection>0));
disp('max closure value: ')
disp(BK_MaxPitValue)
format long
BK_MaxPitValue
toc
```

```
max closure value:
   6.5220e+08
BK_MaxPitValue =
     6.521950369148887e+08
Elapsed time is 8.986110 seconds.
```

Figure 12. The maximum pit value output using the BK maximum flow 'KD' deposit.

5.2. Case 2: Gold/Copper Mine in Nevada, USA (p4hd)

The dataset from a gold and copper mine in Nevada (USA), classified as 'p4hd' in Minelib, was also used in the application of BK maximum flow algorithm in UPLO. The block model contained 40,947 blocks that were 50 x 50 x 20 ft in size. The block indices were also provided so as to know the location of the blocks within the block model. The precedence blocks were also computed at 45 degrees with 8 levels giving a total of 738,609 precedences. EBVs were also provided to help in calculating the ultimate pit value. Figure 13 shows a section of the MATLAB code used to input the data for this deposit and also a section of the data after it was imported in MATLAB.



```
tic
clear all
load('p4hdPREC.mat');
load('p4hdBLOCK.mat');
objfunc = [p4hdBLOCK.EBV];
numblocks = numel(objfunc);
northings = [p4hdBLOCK.YI];
eastings = [p4hdBLOCK.XI];
intcon = 1:numblocks;
elevations = [p4hdBLOCK.ZI];
class = [p4hdBLOCK.class];
```

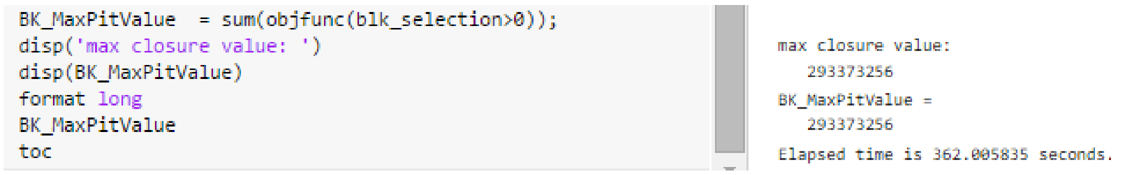
Fiel...	XI	YI	ZI	EBV	class
9103	6	30	45	-4087	0
9104	7	30	45	-4087	0
9105	8	30	45	-4087	0
9106	38	30	45	35670	1
9107	39	30	45	37554	1
9108	40	30	45	31510	1
9109	41	30	45	11870	1
9110	42	30	45	-8058	0
9111	43	30	45	-4761	0
9112	44	30	45	2787	1
9113	45	30	45	6660	1

Figure 13. Sections of the input code and data used for the 'p4hd' deposit.

The ultimate pit limit for this deposit was also calculated using the BK maximum flow algorithm and the push-relabel maximum flow. The BK maximum flow algorithm gave a maximum pit value of $293,373,256 in 362.01 s. The push-relabel maximum flow also gave the same maximum pit value but still with a longer time of 434.61 s. These two methods also concurred with the pseudo-flow maximum pit value [40] given in Minelib. This, therefore, shows that the BK maximum flow algorithm can be applied in the mining industry ultimate pit optimization since it is even faster than the push-relabel maximum flow, which is already being applied. Table 4 gives a summary of the Arizona's copper deposit (KD) and the results of using the two maximum flow methods. The MATLAB output of the maximum pit value obtained using the BK maximum flow algorithm for this deposit is shown in Figure 14.

Table 4. A summary of the gold/copper mine in Nevada, USA (p4hd).

	Block	Block Size (ft)	Precedence	Max. pit value ($)	Time (s)
BK	40,947	50 x 50 x 20	738,609	293,373,256	362.01
Push-relabel	40,947	50 x 50 x 20	738,609	293,373,256	434.61



```
BK_MaxPitValue  = sum(objfunc(blk_selection>0));
disp('max closure value: ')
disp(BK_MaxPitValue)
format long
BK_MaxPitValue
toc
```

```
max closure value:
   293373256
BK_MaxPitValue =
   293373256
Elapsed time is 362.005835 seconds.
```

Figure 14. The maximum pit value output for using BK maximum flow 'p4hd' deposit.

From the two case studies, the difference between the times in the two algorithms is not so big when applied in a block model with few numbers of blocks. From the first case with 14,153 blocks, the BK algorithm reduced the time with 12%, while in the second case with 40,947 blocks, the BK algorithm reduced the time with 16%. The results achieved by the BK maximum flow algorithm, especially in time reduction, can be attributed to two main factors. One of the factors is the use of two search trees instead of one as used in the other algorithms. This helps in speeding-up the process and in-turn saving on time. The second factor is the reuse of trees in the search process instead of starting afresh. This helps in saving time since searching from an already built tree will be quicker, especially from an iterative process. The maximum pit values from the first case and the second case were $652,195,037 and $293,373,256, respectively, using both the BK maximum flow and push-relabel maximum flow algorithms. This also concurs with the results given in Minelib, which provides the best-known solutions for the ultimate pit limit for their datasets [40]. The results of these two case studies show that the BK algorithm can be applied in UPLO and even give better results in terms of time efficiency when applied in the block models with a huge number of blocks.

6. Conclusions

Ultimate pit limit optimization (UPLO), playing a major role in the mining industry and algorithms for solving UPLO, has been developed and improved by various researchers from the 1960s. This work narrowed down to the application of the maximum flow algorithms in UPLO. The maximum flow algorithms such as the push-relabel and pseudo-flow maximum flow algorithms have made their applications in UPLO and have even been demonstrated to perform better than the widely accepted Lerchs–Grossmann algorithm. Production of the optimal results in the shortest time possible plays an important role in UPLO. Since the Boykov-Kolmogorov (BK) algorithm, as a variant of the maximum flow algorithms has been proved to give good practical results in the computer vision problems within a considerable amount of time, it was formulated in MatlabBGL to perform UPLO in two case mining studies. In the first case study, the BK maximum flow algorithm gave a maximum pit value of $652,195,037 in 8.99 s, which was faster compared to push-relabel, which gave the same maximum pit value but in 10.56 s. In the second case study, the BK maximum flow algorithm gave a maximum pit value of $293,373,256 in 362.01 s, which was also faster compared to push-relabel, which gave the same maximum pit value in 434.61 s. This reduction can be attributed to the use of two search trees by the algorithm as well as the reuse of trees in the search process instead of starting afresh. This ascertained that the BK algorithm could as well be applied in the mining industry in pit limit optimization.

Acknowledgments

The authors would like to acknowledge the National Science Foundation of China under Grant 51804235, and the financial support from the CRSRI Open Research Program (Program SN: CKWV201847KY).

Conflicts of interest

The authors declare that they have no competing interests whatsoever.

References

[1]. Kennedy, B. A. (1990). Surface Mining, Second Edition. Society for Mining, Metallurgy, and Exploration.

[2]. Poniewierski, J. (2018). Pseudoflow Explained. Deswik, A discussion of Deswik Pseudoflow Pit Optimization in comparison to Whittle LG Pit Optimization

[3]. Ghebrihiwet, N. (2019). FDI technology spillovers in the mining industry: Lessons from South Africa's mining sector. Resources Policy, vol. 62, 463-471.

[4]. Chatterjee, S., Sethi, M. R., and Asad, M. W. A. (2016). Production phase and ultimate pit limit design under commodity price uncertainty. European Journal of Operational Research, vol. 248, 658-667.

[5]. Zhang, C., Pu, C., Cao, R., Jiang, T., and Huang, G. (2019). The stability and roof-support optimization of roadways passing through unfavorable geological bodies using advanced detection and monitoring methods, among others, in the Sanmenxia Bauxite Mine in China's Henan Province. Bulletin of Engineering Geology and the Environment, 1-13.

[6]. Caccetta, L. and Hill, S. P. (2003). An application of branch and cut to open pit mine scheduling. Journal of global optimization, vol. 27, 349-365.

[7]. Gholamnejad, J. and Mojahedfar, A. (2010). Determination of the largest pit with the non-negative net profit in the open pit mines. Journal of Mining and Environment, vol. 1, 45-52.

[8]. Jamshidi, M. and Osanloo, M. (2018). UPL determination of multi-element deposits with grade uncertainty using a new block economic value

calculation approach. Journal of Mining and Environment, vol. 9, 61-72.

[9]. Shishvan, M. S. and Sattarvand, J. (2012). Modeling of Accurate Variable Slope Angles in Open-Pit Mine Design Using Spline Interpolation/Modelowanie Zmiennego Kąta Nachylenia Stoku W Projektowaniu Kopalni Odkrywkowych Za Pomocą Interpolacji Funkcjami Sklejającymi (Metodą Spline'Ów). Archives of Mining Sciences, vol. 57, 921-932.

[10]. Akbari, A. D., Osanloo, M., and Shirazi, M. A. (2009). Reserve estimation of an open pit mine under price uncertainty by real option approach. Mining Science and Technology (China), vol. 19, 709-717.

[11]. Bai, X., Turczynski, G., Baxter, N., Place, D., and Sinclair-Ross, H. (2017). Pseudoflow Method for Pit Optimization.

[12]. Zhao, Y. (1992). Algorithms for optimum design and planning of open-pit mines. Doctoral Dissertation, University of Arizona.

[13]. Koenigsberg, E. (1982). The optimum contours of an open pit mine: an application of dynamic programming. in 17th Application of Computers and Operations Research in the Mineral Industry.

[14]. Johnson, T. B. (1968). Optimum open pit mine production scheduling. California Univ Berkeley Operations Research Center, California Univ Berkeley Operations Research Center.

[15]. Pana, M. and Davey. (1965). The simulation approach to open-pit design. in Proceeding of 5th International APCOM.

[16]. Korobov, S. (1974). Method for Determining Optimal Ultimate Open Pit Limits. Ecole polytechnique de Montréal.

[17]. Zhao, Y. (1992). A new optimal pit limit design algorithm. in Proc. of the 23rd APCOM, 423-434.

[18]. Khalokakaie, R., Dowd, P., and Fowell, R. (2000). Incorporation of slope design into optimal pit design algorithms. Mining Technology, vol. 109, 70-76.

[19]. Kakaie, R. (2012). A new algorithm for optimum open pit design: Floating cone method III. Journal of Mining and environment, vol. 2, 118-125.

[20]. Wright, A. (1999). Moving Cone II -A simple algorithm for optimum pit limits design. in Proceedings of the 28rd APCOM, 367-374.

[21]. Khalou, K. R. (2007). Optimum Open Pit Design with Modified Moving Cone II Methods. Journal of Faculty of Engineering (University of Tehran), vol. 41, 297-307.

[22]. Dowd, P. and Onur, A. (1992). Optimizing open pit design and sequencing. in Proceedings 23rd Application of Computer in Mineral Industry, 411-422.

[23]. Thorley, U. (2012). Open Pit Mine Planning: Analysis and system modeling of conventional and oil sands applications. Queen's University, Kingston, Ontario, Canada.

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

[25]. Souza, F. R., Melo, M., and Pinto, C. L. L. (2014). A proposal to find the ultimate pit using Ford Fulkerson algorithm. Rem: Revista Escola de Minas, vol. 67, 389-395.

[26]. El-Karmouty, M., El-Wageeh, M., El-Aziz, A. A., and El-Shayeb, Y. (2013). New Technique- "One Three-One Two (13-12)"-In Ultimate Pit Limit Heuristic Algorithms. in The 45th International October Conference on Mining and Metallurgy Bor Lake, Bor (Serbia), 397-400.

[27]. Marcotte, D. and Caron, J. (2013). Ultimate open pit stochastic optimization. Computers & Geosciences, vol. 51, 238-246.

[28]. Khodayari, A. A. (2013). A New Algorithm for Determining Ultimate Pit Limits Based on Network Optimization. Int. Journal of Mining & Geo-Engineering, vol. 47, 129-137.

[29]. Sasaki, K., Dindiwe, C., and Adachi, T. (2001). Optimization of open pit limit designs by newly BPITC approach and initial feasibility study using block grade data set estimated by geostatistical simulation. Journal of the Mining and Materials Processing Institute of Japan(Japan), vol. 117, 62-70.

[30]. Milani, G. (2016). A Genetic Algorithm with Zooming for the Determination of the Optimal Open Pit Mines Layout. The Open Civil Engineering Journal, vol. 10, 301-322.

[31]. Petrov, D., Vasiliev, P., Mikhelev, V., Muromtcev, V., and Batischev, D. (2017). Using parallel computing in modeling and optimization of mineral reserves extraction systems. Journal of Fundamental and Applied Sciences, vol. 9, 939-947.

[32]. Muir, D. (2007). Pseudoflow, New Life for Lerchs-Grossmann pit optimisation. presented at the Orebody Modelling and Strategic Mine Planning, AusIMM Spectrum Series.

[33]. Whittle, D., Brazil, M., Grossman, P. A., Rubinstein, J. H., and Thomas, D. A. (2018). Combined optimisation of an open-pit mine outline and the transition depth to underground mining. European Journal of Operational Research, vol. 268, 624-634.

[34]. Verma, T. and Batra, D. (2012). MaxFlow Revisited: An Empirical Comparison of Maxflow Algorithms for Dense Vision Problems. in BMVC, 1-12.

[35]. Dinic, E. A. (1970). Algorithm for solution of a problem of maximum flow in networks with power estimation. in Soviet Math. Doklady, 1277-1280.

[36]. Boykov, Y. and Kolmogorov, V. (2004). An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. IEEE Transactions on Pattern Analysis & Machine Intelligence, 1124-1137.

[37]. Muir, D. (2007). Pseudoflow, new life for Lerchs-Grossmann pit optimisation. Orebody Modelling and Strategic Mine Planning, AusIMM Spectrum Series, vol. 14,

[38]. Gleich, D. (2006). MatlabBGL. Accessed via https://github.com/dgleich/matlab-bgl,

[39]. Siek, J., Lumsdaine, A., and Lee, L.-Q. (2002). The boost graph library: user guide and reference manual. Addison-Wesley.

[40]. Espinoza, D., Goycoolea, M., Moreno, E., and Newman, A. (2013). MineLib: a library of open pit mining problems. Annals of Operations Research, Accessed via http://mansci-web.uai.cl/minelib/, vol. 206, 93-114.

بهینه سازی حد نهایی پیت معدن با استفاده از الگوریتم حداکثر جریان Boykov-Kolmogorov

آکیسا دیوید موانگی[۱، ۲*]، ژانگ جیان هوآ[۱]، هوآنگ گانگ[۱]، ریچارد موتویی کاسومو[۱] و ماتیدزا مولالو اینسنت[۱]

۱- دانشکده منابع و مهندسی محیط زیست، دانشگاه صنعتی ووهان، ووهان، هوبئی، چین

۲- گروه مهندسی معدن، مواد و نفت، دانشگاه کشاورزی و فناوری جومو کنیاتا، نایروبی، کنیا

ارسال ۲۰۲۰/۱۰/۱۳، پذیرش ۲۰۲۰/۱۲/۱۰

* نویسنده مسئول مکاتبات: davidmwas06@gmail.com

چکیده:

بهینه‌سازی حد نهایی پیت معدن (UPLO) به عنوان گام مهمی در روند برنامه ریزی تولید در معادن است. روش‌های مختلف الگوریتم‌های حداکثر جریان مانند شبه جریان برای بهینه سازی پیت نهایی معدن استفاده شده و نتایج خوبی داده است. از الگوریتم حداکثر جریان Boykov-Kolmogorov (BK) در حل مسائل با دیدگاه رایانه‌ای استفاده شده است و نتایج عملی بسیار خوبی به همراه داشته است؛ اما از این روش هرگز در UPLO استفاده نشده است. در این پژوهش، برای انجام UPLO در دو مطالعه موردی، از الگوریتم حداکثر جریان BK و الگوریتم Push-Relabel با فرموله شدن در نرم افزار MATLAB استفاده شده است. مقایسه نتایج در هر دو مطالعه موردی برای الگوریتم حداکثر جریان BK و الگوریتم حداکثر جریان Push-Relabel نشان داد که، حداکثر اندازه پیت نهایی معدن برای هر دو مورد یکسان است اما الگوریتم حداکثر جریان BK زمان محاسبه پیت نهایی را در حالت اول ۱۲٪ و در حالت دوم ۱۶٪ کاهش می‌دهد. این کاربرد موفقیت آمیز الگوریتم حداکثر جریان BK نشان می‌دهد که می‌توان از آن برای محاسبه UPLO نیز استفاده کرد.

کلمات کلیدی: الگوریتم Boykov-Kolmogorov، حداکثر جریان، Pseudo-flow، حد نهایی پیت.