Abstract: This paper deals with the Open-Pit-Mining Operational Planning problem concerning dynamic truck allocation. The objective is to meet production goals and quality requirements and to minimize the number of mining trucks used in the mineral extraction from open-pit mines. Due to the combinatorial complexity and the conflicting goals, a multi-objective heuristic strategy is adopted. We present two hybrid genetic algorithms, NSGA and MGHA, that combine characteristics of multi-objective heuristics and a local search heuristic. For each algorithm, two variants in relation to the fitness function are proposed. Both variants consider production and quality goals. The difference between the variants is that one includes number of truck minimization (NT) and the other includes truck usage rate maximization (TU). The proposed algorithms were tested using modified data from literature. The computational experiments show that the NSGAII-NT algorithm is the best one.
Hybrid multi-objective heuristics for the open-pit-mining operational planning problem

G. Pantuza Jr; M. J. F. Souza; I. E. Cabral

This paper deals with the Open-Pit-Mining Operational Planning problem with dynamic truck allocation. The objective is to meet the production goals, the quality requirement and minimize the number of mining trucks used in the mineral extraction in the open-pit mines.

Due to the combinatorial complexity and the conflicting goals, we present two multi-objective hybrid genetic algorithms for the proposed problem, NSGA II and AGHM. This two hybrid genetic algorithms combine characteristics of multi-objective heuristics and a local search heuristic.

For each algorithm it was proposed two variants in relation to the fitness function. Both variants consider production and quality goals. The difference between the variants is that one includes the minimization of the number of trucks (TN) and the other includes the maximization of the truck usage rate (TU). Unlike previous works that treat the problem as a mono-objective optimization, these multi-objective strategies provide more alternatives for the decision maker, who can choose the most appropriate solution. The proposed algorithms were tested using modified data from literature.
Hybrid multi-objective heuristics for the open-pit-mining operational planning problem

G. PANTUZA Jr., M. J. F. SOUZA, I. E. CABRAL

Universidade Federal de Ouro Preto, Campus Universitário, Morro do Cruzeiro, Ouro Preto – MG, Brazil

Abstract

This paper deals with the Open-Pit-Mining Operational Planning problem concerning dynamic truck allocation. The objective is to meet production goals and quality requirements and to minimize the number of mining trucks used in the mineral extraction from open-pit mines. Due to the combinatorial complexity and the conflicting goals, a multi-objective heuristic strategy is adopted. We present two hybrid genetic algorithms, NSGA and MGHA, that combine characteristics of multi-objective heuristics and a local search heuristic. For each algorithm, two variants in relation to the fitness function are proposed. Both variants consider production and quality goals. The difference between the variants is that one includes number of truck minimization (NT) and the other includes truck usage rate maximization (TU). The proposed algorithms were tested using modified data from literature. The computational experiments show that the NSGAII-NT algorithm is the best one.

Keywords: Open-Pit Mining Planning, Multi-Objective Optimization, Multi-Objective Heuristics Algorithm

*Corresponding author

Email addresses: guido.junior@ifmg.edu.br (G. PANTUZA Jr), marcone@iceb.ufop.br (M. J. F. SOUZA), cabralmg@uai.com.br (I. E. CABRAL)

1 Escola de Minas – Departamento de Engenharia de Minas
2 Instituto de Ciências Exatas e Biológicas – Departamento de Computação
3 Escola de Minas – Departamento de Engenharia de Minas

1. Introduction

This paper deals with an Open-Pit-Mining Operational Planning problem (OPMOP). In this problem, determining the hourly production rate at the mine is desirable, in order to minimize deviations from the production and quality goals of the produced ore, as well as to minimize the number of trucks involved in the process.

The hourly production rate is determined by the summation of the number of trips that each truck visits each pit in one hour. Pit is the location from which the material is withdrawn, i.e., ore (material with economic value) or waste (material without economic value). For the removal of material from a pit, it is necessary to allocate a loader (shovel) for it. Each pit has different percentages of control parameters (e.g., %Fe, %SiO2) and different amounts of material available.

According to Costa et al. [1], this problem is NP-hard, i.e., exact methods are not usually able to solve the problem in adequate computational time. In this case, it is recommended to resolve the problem by heuristic methods. These methods do not provide the optimality of the solution, but they are able to generate good solutions in a computational time smaller than required by the exact method.

On the other hand, the objectives of the problem are conflicting. In fact, as the number of trucks is reduced, the difficulty of achieving the ore production and quality targets increases. As such, heuristic algorithms based on multi-objective optimization have been developed.

In problems that involve multi-objective optimization, there is no single solution that simultaneously optimizes all objectives. Therefore, this approach involves finding a set of efficient solutions, which have different characteristics from the goals.

Herein, are presented two multi-objective hybrid genetic algorithms for the proposed problem. Previous works treat the problem as a mono-objective optimization. However, the generation of multiple solutions serving multiple goals provides more alternatives for the decision maker, who can choose for the most appropriate solution.

The remainder of this paper is organized as follows. Section 2 presents a brief theoretical background and some work related to the topic under study. Section 3 describes the problem under study. The methodology used to deal with the problem is presented in section 4. In Section 5, the used scenarios and results are presented and in section 6, conclusions are drawn.
2. Literature review

2.1. Multi-objective Optimization

The concept of Pareto-optimal is the origin of multi-objective optimization. According to Pareto [2], by definition, a set of solutions $I$ is Pareto-optimal, if there is another set of feasible solutions $I^*$ that can improve any objective without worsening at least one other objective. In other words, a solution $S$ belongs to the set of Pareto-optimal solutions, if there is no solution $S^*$ that dominates $S$.

Considering a minimization problem, we have:

- $S$ dominates $S^*$ if, and only if, $S_j \leq S^*_j \ \forall j$ and $S_j < s^*_j$ for some $j$;
- $S$ and $S^*$ are indifferent or have the same degree of dominance if, and only if, $S$ does not dominate $S^*$ and $S^*$ does not dominate $S$.

Among the numerous works related to multi-objective heuristic approaches, we highlight the genetic algorithms (GAs). GAs are based on the theory of evolution and are among the most used because of their success in solving various problems of combinatorial nature (Arroyo [3]).

In multi-objective genetic algorithms, each generation, or iteration has a set of individuals, or parents solutions, to which genetic operators are applied (e.g., crossover and mutation) for generating a new population of solutions formed by the parent- and offspring-solutions. Among the numerous papers on multi-objective genetic algorithm, we point out: Schaffer [4], Fonseca and Fleming [5], Horn et al. [6], Srivivas and Deb [7] and Zitzler [8].

Schaffer [4] proposes an algorithm called the Vector Evaluated Genetic Algorithm (VEGA). In each generation, a group of individuals that surpasses the other in accordance with one of the $ob$ goals is selected, until $ob$ groups are formed. Next, these groups are mixed and genetic operators are applied to construct the next generation.

In the Multi-objective Genetic Algorithm (MOGA) proposed in Fonseca and Fleming [5], each individual $S$ is classified at a level according to the number of individuals that $S$ dominates. The fitness of each individual is assigned in accordance with an interpolation between the best and the worst level. The final fitness assigned to all individuals of the same level is the same and it is equal to the average fitness of this level. Thus, all individuals of the same level are indifferent to each other.
With the Niche Pareto Genetic Algorithm (NPGA), Horn et al. [6], use a tournament based on the concept of Pareto dominance for selecting the individuals. Two individuals are selected and compared with a subset of the population. The one that is not dominated is selected for the next generation.

Srivivas and Deb [7] propose the Non-dominated Sorting Genetic Algorithm (NSGA), in which individuals are classified into levels according to their degree of dominance $Front_k$. A fitness value is assigned to each individual according to its level and its distance from other solutions at the same level, the call crowding distance. The selection is made through tournaments using the fitness value until all the vacancies for the next generation are fulfilled.

In the Strength Pareto Evolutionary Algorithm (SPEA) proposed by Zitzler [8] a selection based on the dominance relationship is used to evaluate and select solutions. In order to evaluate this dominance relationship and to rank individuals as to their level of dominance, the SPEA uses an additional population set, where the non-dominated individuals of the population of the previous generation determine the fitness of individuals of the current population.

Besides these, several other multi-objective genetic algorithms have been proposed and explored, such as those listed below: Knowles [9], Deb et al. [10], C. A. Coello Coello and Lamont [11], Arroyo [3], Arroyo and Armentano [12], Tan et al. [13] and Deb and Tiwari [14].

2.2. Mine Planning

Because of the importance and complexity of mine planning, according to Crawford and Hustrulid [15], the challenge to optimize this process has become a major problem faced by companies in the mining sector. As a result, many papers have been produced dealing with this problem.

The work of White et al. [16] presents a linear programming model to minimize the number of trucks necessary to attend the restrictions for the continuous flow of the material through the loading and unloading points, and the productive capacity of the these trucks.

White and Olson [17] present a linear programming model, which can be divided into two parts: the first, carries out an ore blending optimization problem in which the objective is cost minimization as related to the quality, transportation and storage of the material; and the second, seeks to minimize the number of trucks using as decision variable the volume of transported
material. The authors consider the production rate, the feed rate to the processing plant and the blend quality.

In Chanda and Dagdelen [18] a linear goal programming model is presented. The objective is to minimize deviations from the established production and quality goals. According to the authors, this technique is best suited to the reality of mining, because its goal is to get the solution as close as possible to the required production and quality goals.

Merschmann and Pinto [19] seek to maximize the production rate by using two different models for the equipment allocation. One of these considers the static allocation of trucks; i.e., a particular truck only travels to a single pit. The other adopts the dynamic allocation model of trucks; i.e., a truck can make trips to different pits after each unload.

Merschmann [20] developed a computational system optimization and simulation for the operational planning problem for mines called OTISIMIN. The simulation system uses the result of the optimization model. The linear programming model used in the optimization neither considers the quality goals, nor the minimization of the number of trucks.

Costa et al. [1] and Costa et al. [21] generalized the model of Merschmann and Pinto [19], by including more operational restrictions, and treating various operational requirements together. The first paper adopts the dynamic allocation of trucks, while the second one, the static allocation. In both studies, the linear goal programming was used taking into consideration a mono-objective function represented by the weighted sum of the deviations from the required production and quality goals.

Since the rate of production of each pit also depends on logistics management, i.e., of the characteristics of the shovels and trucks designed to operate in that pit, Guimaraes et al. [22] developed a goal programming model involving dynamic allocation. This model added to the Costa et al. [1] model, restrictions on the usage rate of transport vehicles. Moreover, a third goal was proposed, that is, the minimization of the number of trucks. The results of the optimization model were also validated by a computer simulation model.

Pantuza Jr. and Souza [23] proposed a multi-objective linear programming model by adopting the classical $\varepsilon$-restricted formulation. The authors considered two points of discharge, one for ore and the other one for waste. Furthermore, it was considered that cycle time depends on the truck type, the transported material, the pit of origin, and the destination of the trip.

Souza et al. [24] seek to optimize the production and quality of ore, as
well as minimizing the number of trucks, adopting the dynamic allocation method of trucks. The paper proposes a hybrid heuristic algorithm based on the Variable Neighborhood Search General method, and compares the results obtained by this algorithm with those obtained by the CPLEX optimizer applied to a mixed integer programming model.

3. Problem Under Study

The problem under study deals the open-pit-mining operational planning problem (OPMOP), considering the dynamic allocation of trucks. As shown in Figure 1, taken from Souza et al. [24], OPMOP involves a set of pits, shovels and trucks. It consists in selecting the pits to be used, allocating a shovel to them and determining the number of trips of each truck to these pits. The trucks, which are loaded by shovels, transport the material removed from the pit to their point of discharge.

For selecting the pits that will be explored for the mixture composition mixture, it is necessary to take into account the specifications of the existent ore; such as, the percentage of certain chemical elements, %Fe and %SiO. It is important to note that these specifications vary with each mining pit.
Thus, according to the specifications of the desired product, one pit should be given priority over the other.

Moreover, ore removal from the pits is performed by shovels and each shovel is associated with a minimum and maximum production. The maximum production is determined by the productive capacity of the equipment, while the minimum production is established in order to determine if the shovels use is economically feasible.

To determine the number of trips of each truck to each pit, it is necessary to consider which shovel is allocated to the pit, the rate of usage of the truck and the production goals of the ore and waste. The shovel allocated must be operationally compatible with the truck.

Moreover, the maximum usage rate of the truck must be respected in order to limit the number of trips each truck performs within an hour. The ore production must respect a fixed ratio in relation to the amount of waste (relation waste/ore), as well as the production goals. Production goals and its lower and upper limits are defined by the processing capacity of the plant that will receive the produced ore.

Herein, it was considered that there exists a heterogeneous fleet of trucks, i.e. they have different load capacities. In addition, the method of dynamic allocation is adopted, which means that one truck can be allocated to different pits after each discharge of material. This technique reduces the length of the queue and increases the usage of trucks.

Unlike most previous studies in literature, e.g., Costa et al. [1], Guimaraes et al. [22], Merschmann [20] and Souza et al. [24], two points of discharge for the mined material is considered: the primary crusher, to discharge the ore; and the waste pile, to dispose of the waste. This is due to the fact that the trajectories of the trucks to the waste pile are increasingly distant due to environmental protection regulations, with a significant difference between the trip times from the pit to the crusher and from the pits to waste pile.

We also consider the trucks cycle time as a variable that depends on the pit, the type of equipment, and the type of transported material. This is due to the fact that each type of truck demands a different time to travel the same path and this time is also influenced by the type of material (ore or waste).
4. Methodology

This paper proposes two different evolutionary algorithms adapted to the problem under study, a Multi-objective Genetic Hybrid Algorithm (MGHA) and a hybrid algorithm based on Fast Multi-objective Non-dominated Sorting Genetic Algorithm, NSGA II (Deb et al. [10]). Both will be used in conjunction with the local search algorithm VND of Mladenovic and Hansen [25].

4.1. MGHA

The proposed MGHA begins its execution starting from a randomly generated initial population $I_0$, with the maximum number of the individuals of the population being $(N_{max})$, defined by empirical tests.

After this step, individuals are classified into various levels $Front_1$, $Front_2$, \ldots, $Front_k$, according to their degree of dominance.

Then the crowding distance is calculated for such individuals. After that, it forms a new generation of $Q$ individuals by applying genetic operators. The best individuals are selected for each objective to integrate the next generation. This process is repeated until it reaches the maximum number of generations $t_{max}$, which was also defined by empirical tests.

Algorithm 1 presents the pseudocode of the proposed MGHA.

\begin{algorithm}
\caption{MGHA}
\begin{algorithmic}
\State $Q_0 \leftarrow \emptyset$; 
\Comment{Create a parent population $I_0$;}
\For{$t = 0$ \textbf{to} $t_{max}$}
\State $Front \leftarrow$ fast nondominated sort;
\State $F \leftarrow$ Calculate crowding distance;
\State $Q_t \leftarrow$ genetic operators on $I_t$;
\State $R \leftarrow I_t \cup Q_t$;
\State $Front \leftarrow$ fast nondominated sort;
\State $Front \leftarrow$ Calculate crowding distance;
\State Select next generation;
\EndFor
\State \textbf{Return} $I_{t_{max}}$;
\end{algorithmic}
\end{algorithm}
4.2. NSGA II

The algorithm NSGA II proposed by Deb et al. [10], illustrated in Algorithm 2, begins with an initial population randomly generated $I_0$, on which a selection is made through a binary tournament for the parents choice. This choice is based on the fitness of each individual.

The fitness is calculated according to the level of dominance $Front$ ($Front_1$: best solutions) and its distance from other solutions to the same level, the so-called crowding distance. The new individuals, $Q_t$, are created using genetic operators applied to parents $I_t$.

Then the entire population $R_t$, (of parents and offspring) and the fitness is again calculated. The fittest individuals, i.e. those with lower levels are maintained for the next generation, until all vacancies for the next generation are fulfilled. In the event of a tie, the individual with the greatest value of the crowding distance is chosen. This procedure is repeated until the algorithm reaches the maximum number of generations, given by $t_{max}$. 
Algorithm 2 NSGA II

$Q_0 \leftarrow \emptyset$;
Make the random parent population $I_0$;
for $t = 0$ to $t_{\text{max}}$ do
    $R \leftarrow I_t \cup Q_t$ {combine parent and offspring population};
    $\text{Front} \leftarrow \text{fast-nondominated-sort } (I_g)$ {Front = (Front$_1$, Front$_2$, ...),
    all nondominated fronts of $R_t$};
    {while the parent population is not filled}
    while $|I_{t+1}| + |\text{Front}_k| < N_{\text{max}}$ do
        crowding-distance-assignment(\text{Front}_k) {calculate crowding-distance
        in \text{Front}_k};
        $I_{t+1} \leftarrow I_{t+1} \cup \text{Front}_k$ {include $i$-th nondominated front in the parent
        pop};
        $k \leftarrow k + 1$ {check the next front for inclusion};
        sort(\text{Front}_k) {sort in descending order using non-domination rank
        and crowding distance};
        $I_{t+1} \leftarrow I_{t+1} \cup \text{Front}_k[1:(N_{\text{max}} - |I_{t+1}|)]$ {choose the first $(N_{\text{max}} - |
        I_{t+1}|)$ elements of \text{Front}_k};
        $Q_{t+1} \leftarrow \text{make-new-pop}(I_{t+1})$ {use genetic operators to create a new
        population $Q_{t+1}$};
        $t \leftarrow t + 1$ {increment the generation counter};
    end while
end for
Return $I_{t_{\text{max}}}$;
4.2.1. Fast Non Dominated Sorting

Algorithm 3 Fast Non Dominated Sorting(\(I\))

1: for \(S = 0\) to \(S < |I|\) do
2: \(Id_S \leftarrow \emptyset\);
3: \(Nd_S \leftarrow 0\);
4: for \(S' = 0\) e \(S' \neq S\) to \(S' < |I|\) do
5: if \(S\) dominates \(S'\) then
6: \(Id_S \leftarrow Id_S \cup \{p\}\) \{Add \(S'\) to the set of solutions dominated by \(S\);\}
7: end if
8: if \(p\) dominates \(S\) then
9: \(Nd_S \leftarrow Nd_S + 1\) \{Increment the domination counter of \(S\);\}
10: end if
11: end for
12: if \(Nd_S = 0\) then
13: \(Front_1 \leftarrow Front_1 \cup \{S\}\);
14: end if
15: end for
16: \(k \leftarrow 1\);
17: while \(Front_k \neq \emptyset\) do
18: \(aux \leftarrow \emptyset\);
19: for \(S = 0\) e \(S \in Front_k\) to \(S < |I|\) do
20: for \(S' = 0\) e \(S' \in Id_S\) to \(p < |I|\) do
21: \(Nd_p \leftarrow Nd_p + 1\);
22: if \(Nd_p = 0\) then
23: \(aux \leftarrow aux \cup \{p\}\);
24: end if
25: end for
26: end for
27: \(k \leftarrow k + 1\);
28: \(Front_k \leftarrow aux\);
29: end while
30: Return \(Front\);  

The Algorithm 3 illustrates the fast non dominated sorting procedure, which calculates the fitness of each individual. To each individual \(S\), of the
set of solutions $I$, two values are associated: $Nd_S$ and $Id_S$.

- $Id_S$ is the set of individuals that are dominated by the individual $S$;
- $Nd_S$ is the number of individuals that dominate the individual $S$.

Initially, lines 2-16 of Algorithm 3 calculate such values. It can be noticed that individuals with $Nd_S = 0$ (not dominated) are contained in the $Front_1$ level. Next, the lines in the 17 to 30 range, the number of individuals dominated $Id_S$ for each individual $S$ of $Front_k$.

The counter $Nd_S$ of each individual $S$ in $Id_S$ is decreased by one unit. If $Nd_S = 0$, then the solution $S$ belongs to the current front. The algorithm is repeated until all individuals are classified at the level $Front_k$.

Once the sorting is complete the crowding distance is calculated. It involves the average distance of two individuals adjacent to each individual in the population for all of the objectives.

Thus, individuals are classified according to their distribution in the whole solution, and the more scattered individuals are prioritized. This is done to ensure that all visited individuals are closer to the Pareto-optimal set.

4.2.2. Crowding distance

Algorithm 4 shows how to calculate the crowding distance. In this algorithm, $dist_S$ is the crowding distance of the individual $S$ of the set $I$, $NS$ is the number of solutions contained in $I$, $f_{ob}(S)$ is the value of $ob$-th objective function of the solution $S$ and $OB$ the number of objectives.

<table>
<thead>
<tr>
<th>Algorithm 4 Crowding Distance</th>
</tr>
</thead>
<tbody>
<tr>
<td>1: $dist_S \leftarrow \emptyset$;</td>
</tr>
<tr>
<td>2: for $obj = 1$ to $OB$ do</td>
</tr>
<tr>
<td>3: sort ($obj; f_{obj}$) {sort using each objective value};</td>
</tr>
<tr>
<td>4: $dist_0 \leftarrow dist_{NS-1} \leftarrow \infty$;</td>
</tr>
<tr>
<td>5: for $S = 1$ to $N - 1$ do</td>
</tr>
<tr>
<td>6: $dist_S \leftarrow dist_S + f_{obj}(S + 1) - f_{obj}(S - 1)$;</td>
</tr>
<tr>
<td>7: end for</td>
</tr>
<tr>
<td>8: end for</td>
</tr>
<tr>
<td>9: Return $dist_S$;</td>
</tr>
</tbody>
</table>

The calculation of the crowding distance allows that better-spread individuals will occupy the last places available for the next generation, ensuring a diversity of solutions.
4.3. Representation of an individual

For each individual $S$, there are used $|T|$ integer arrays of size $|No| \times (|No| + 1)$, where:

- $T$ is the set of transport equipment;
- $No$ is the set of nodes, formed by the set of pits $F$, points of discharge (primary crusher – ore, and waste pile – waste) at origin node (begin) and the destination node (end).

The matrix $S$ can be decomposed into two sub-matrices $Y$ and $V$, i.e.:

$$S_{|N|\times(|N|+1)} = Y_{|N|\times1} \cup V_{|No|\times|No|}.$$  

The sub-matrix $Y_{|N|\times1}$ represents the allocation of shovels to a set $No$ of nodes and the respective status of these devices, which can be active or not. The sub-matrix $V_{|No|\times|No|}$ represents the number of times each truck $l$ uses the arc $(b, e)$, where $b$ is the origin node and $e$ is the destination node, with $b, e \in No$.

Each matrix $S^l$ belonging to the individual represents the route to be used by truck $l$. While the column “Shovel”, which is the same for all trucks $l$ of the same individual, represents the allocation of the shovel to the pits, the lines represent the origin nodes $b$ and the other columns, the destination nodes $e$ of the arc $(b, e)$.

<table>
<thead>
<tr>
<th>Shovel</th>
<th>Begin</th>
<th>$F_1$</th>
<th>$F_2$</th>
<th>$F_3$</th>
<th>...</th>
<th>$F_{13}$</th>
<th>ore</th>
<th>waste</th>
<th>End</th>
</tr>
</thead>
<tbody>
<tr>
<td>$Begin$</td>
<td>•</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>...</td>
<td>0</td>
<td>3</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>$F_1$</td>
<td>(Shovel 2, 1)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>...</td>
<td>0</td>
<td>3</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$F_2$</td>
<td>(Shovel 8, 0)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>...</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$F_3$</td>
<td>(Shovel 3, 1)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>...</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>$F_4$</td>
<td>(Shovel 4, 1)</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>...</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>$F_{13}$</td>
<td>(Empty, 0)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>...</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$ore$</td>
<td>•</td>
<td>0</td>
<td>3</td>
<td>0</td>
<td>1</td>
<td>...</td>
<td>0</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>$waste$</td>
<td>•</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>...</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>$End$</td>
<td>•</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>...</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 1: Individual representation
In this representation, the truck always leaves the origin and goes on to the primary crusher (ore): first, making the trips to the ore pits allocated to it; and afterwards, traveling to the waste pits. Finally, after completion of all trips, it goes on to the destination node.

Table 1 illustrates an example of a solution for the problem. The value \((\text{Shovel}_2, 1)\) in line \(F_1\) column “Shovel” can be observed in this table. This value indicates that the shovel \(\text{Shovel}_2\) is allocated ahead to the pit \(F_1\) and it is active.

In the “Shovel” column, line \(F_2\), the value \((\text{Shovel}_8, 0)\) indicates that shovel \(\text{Shovel}_8\) is allocated ahead \(F_2\), but it is inactive. Also in this column, row \(F_f\), value \((\text{Empty}, 0)\) indicates that this pit is available because there is no shovel allocated to it. The letter “X” in the line \(F_4\) means the incompatibility between the truck and the \(\text{Shovel}_4\). The symbol “•” in column “Shovel” means no loader can be allocated to the corresponding pit.

In Table 1, the value in the “Begin” line, “ore” column, represents the total number of trips that the truck \(l\) does to the ore pits and the column “waste” of this line represents the total number of trips that the truck \(l\) makes to the waste pits. The remaining values represent the route of the truck \(l\). For example, the value 3 of the cell \(\text{V}_{\text{ore},F_1}\) indicates that the truck \(l\) will leave the primary crusher and it will make a trip to the ore pit \(F_1\). Next, it will return to the primary crusher and it will repeat three times this route. After making all trips to \(F_1\), it will leave the primary crusher in direction of another ore or waste pit.

4.4. Initial Population

The process of determining the initial population can be divided into two stages, each one repeated for each individual until the maximum number of the individuals of the population \((N_{\text{max}})\) is reached.

In the first stage, for each available shovel, a pit is randomly chosen in the sub-matrix \(Y\). We observe that each pit receives at most a single shovel.

The second stage is repeated for all pits that have a shovel allocated. For each truck that is operationally compatible with the shovel, a number of trips is assigned to the truck for this pit. This value is random and less than the maximum number of trips that the truck \(l\) can make in one hour. In this calculus, the cycle time of the trucks as well as their rate of usage is considered.

This value is assigned to cell \(\text{V}_{F_i;\text{ore}}\) if it is an ore pit or to cell \(\text{V}_{F_i;\text{waste}}\), if it is a waste pit. This assignment is made for the shovel until their maximum
productivity is achieved or until no more trucks are available. A truck is only
used in pit $F_i$ if its maximum rate of use has not been reached.

4.5. Evaluation of an individual

Individuals are evaluated according to the Pareto dominance relationship. Each one is rated at a level $Front_k$ according to their degree of dominance over other individuals using Algorithms 3 and 4.

To calculate the dominance relationship of the individual $S$, two different methodologies are considered. The first adopts three objectives: namely, the deviations from production and quality goals, and number of trucks. The versions of genetic algorithms that use this methodology evaluation of the objective function are denoted by NSGAII-NT and MGHA-NT, respectively.

The second one also uses three goals: in this case, deviations from production and quality goals and usage of trucks, trying to minimize the number of trucks indirectly. The versions of these algorithms are denoted by NSGAII-TU and MGHA-TU, respectively.

The ore production related to an individual of the population $S$ is measured according to Eq. (1).

$$f^p(S) = \theta \times |P - Pr|$$

in which:

- $P$ : ore production (t);
- $Pr$ : goal for ore production (t);
- $\theta$ : weight associated with the evaluation of production.

The value of ore production $P$ can be obtained by the sum of the loads of all trips made by the trucks to the ore pits, multiplied by their respective capacities.

The quality deviation of the final product depends on how much ore from each pit is used in the blend and the values of the $j$-th control parameters. This deviation is calculated according to Eq. (2).

$$f^q(S) = \sum_{j \in Q} \beta_j \times |tr_j - tc_j|$$

being:
\( Q \) : set of quality parameters \( j \) (\%);
\( tc_j \) : percentage of the control parameter \( j \) in the final blend;
\( tr_j \) : percentage of the goal of control parameter \( j \) in the final blend;
\( \beta_j \) : weight associated with the evaluation of the quality parameter \( j \).

Each pit \( i \) has a value \( t_{ij} \) for the quality parameter \( j \), then \( tc_j \) can be obtained by the weighted average of \( t_{ij} \) and the production of each front ore \( x_i \).

The number of trucks used is calculated by Eq. (3):

\[
f^n(S) = \alpha \times NT
\]  

in which:

\( NT \) : number of trucks used;
\( \alpha \) : weight associated with the evaluation of the number of trucks.

The utilization rate of the truck \( l \) is expressed by Eq. (4). If a truck is not used, the penalty for its utilization rate will be zero.

\[
f^u(S) = \delta \times \sum_{t \in T} \frac{\sum_{b \in \mathcal{N}_o} \sum_{e \in \mathcal{N}_o} V_{l,b,e} T_{m,l,b,e}}{Ph}
\]  

in which:

\( V_{l,b,e} \) : number of trips of the truck \( l \) on the arc \((b,e)\);
\( T_{m,l,b,e} \) : time of the trip to the truck \( l \) on the arc \((b,e)\);
\( Ph \) : planning horizon;
\( \delta \) : weight associated with the evaluation of the truck usage rate.

Other desirable objectives, such as the ratio waste/ore Eq. (5) and the shovel production Eq. (6) are also evaluated, but are not used to calculate the degree of dominance. They are used only to ensure the viability of the solutions and in the mono-objective function eqrefeq.favnd of the VND algorithm.
The ratio waste/ore measures the proportion of waste that is recorded in relation to the mined ore, expressed as tons of waste per ton of ore. It is calculated by dividing the value found for the production of waste by the value found for the production of ore. The ratio waste/ore is evaluated according to Eq. (5).

\[ f_{\text{ratio}}(S) = \varphi \times |RWO - RWO_{Rec}| \]  

being:

- \( RWO \): ratio waste/ore of the current individual;
- \( RWO_{Rec} \): recommended ratio waste/ore;
- \( \varphi \): weight associated with the evaluation of the ratio waste/ore.

The shovel must operate in a range of production to ensure its operational viability. Production of the shovel is evaluated according to the production rate of the pit in which it is allocated according to Eq. (6).

\[ f_{\text{shovel}}(S) = \sigma \times \sum_{i \in F} P_{\text{shovel}}(S) \]  

4.6. Recombination

The selection of parents is done through a binary tournament. Two individuals are selected and the one who has a lower \( \text{Front}_k \) is selected. In the case of a tie, the crowding distance \( \text{dist}_S \) is calculated, and the one that has the largest value of \( \text{dist}_S \) is selected. The crowding distance is calculated according to Algorithm 4.

The crowding distance calculation allows that more-spread individuals occupy the last places available for next population \( I_{t+1} \), in order to guarantee a diversity of solutions (Deb et al. [26]).

After the selection of individuals, only the sub-matrix \( Y \) containing the shovels allocated to each pit is maintained. The sub-matrix \( V \) is reconstructed by the procedure described in subsection 4.4.

4.7. Local Search

After the creation of the offspring, these are refined by the Variable Neighborhood Descent (VND) method. The VND is a method that explores the solution space by a systematic change of neighborhoods.
Initially, there is a set of \( r \) distinct neighborhoods, each defined by a type of movement, previously ordered.

Next, the VND selects an individual, called the current individual, and examines all individuals who are in its first neighborhood, moving to one that represents an improvement according to the evaluation function 7.

\[
f^{VND}(s) = f^p(s) + f^q(s) + f^n(s) + f^u(s) + f^{ruo}(s) + f^{pshovel}(s) \quad (7)
\]

This procedure is repeated until the individual is not an improvement. In this case, it starts looking for the best individual in the second neighborhood. If there is improvement, it returns to the first neighborhood, otherwise it goes to the next neighborhood.

The method terminates when an individual is found that has no better neighbor than him in any of the considered neighborhoods. Its pseudocode is shown in Algorithm (5).

**Algorithm 5 VND**

\[
\begin{align*}
& r \leftarrow \text{number of neighborhoods;} \\
& k \leftarrow 1; \{\text{current neighborhood}\} \\
& \text{while } k \leq r \text{ do} \\
& \quad \text{find-the-best-neighbor } S' \in N^{(k)}(S); \\
& \quad \text{if } f(S') < f(S) \text{ then} \\
& \quad \quad S \leftarrow S'; \\
& \quad \quad k \leftarrow 1; \\
& \quad \text{else} \\
& \quad \quad k \leftarrow k + 1; \\
& \text{end if} \\
& \text{end while} \\
& \text{Return } S;
\end{align*}
\]

4.8. Neighborhood structures

To explore the solution space of the problem the six different movements used in Costa [27] were applied. The six different neighborhoods generated by these movements are described below:

**Movement Load - \( N^L(S) \):** It consists in exchanging shovel operating in \( i \) and \( k \) pits, where the two pits have allocated shovel. If only one of
the pits has a shovel and the other one is available, this movement will consist of relocating the shovel to the available pit. To maintain compatibility between shovels and trucks, the number of trips are relocated along with the selected pits.

**Movement number of trips** - $N^{NT}(S)$: This move is to increase or decrease the number of trips from a truck $l$ in a pit $i$, where a compatible shovel is operating.

**Movement relocate trip from a truck** - $N^{TT}(S)$: consists of selecting two active pits $i$ and $k$, a truck $l$ operationally compatible with these pits and passes pit $i$ to the front of pit $k$. Thus, a truck $l$ does not perform a trip to pit $i$, so that it can perform it in pit $k$.

**Movement relocate trip from a pit** - $N^{TP}(S)$: Here, pit $i$ and two trucks $l$ and $k$ are selected, and a trip of this truck $l$ to pit $i$ is transferred to truck $k$. Compatibility restrictions between equipment are respected in this movement, with relocation of the trip assigned only when there is compatibility between them.

**Movement swap trips** - $N^{ST}(S)$: Two cells of the matrix $V$ are selected and a trip is reallocated from one to another. Such movement can occur between any two cells of the matrix $V$, respecting the constraints of compatibility between devices.

**Movement swap shovels** - $N^{SS}(S)$: It involves exchanging the loaders that operate for the pits $i$ and $k$. Similarly, for the movement $N^{L}$, in a movement $N^{SS}$, the shovel for the pits is replaced, but the trips made to the pits are not changed. To maintain compatibility between shovels and trucks, the incompatible trips are removed.

### 4.9. Survivor population

The algorithms proposed in this paper differ by the method of choice of the surviving population.

To set the surviving population in the algorithm, the MGHA rule of elitism is applied for each goal. That is, the fittest individuals survive for each objective, i.e. those with lower value of the corresponding objective. To define the number of individuals for each objective, a random choice of the objective is made by ensuring that each objective is chosen at least once.
For the NSGA II, the fittest individuals, i.e. those with lower \( Front_k \) are maintained for the next generation, until all vacancies for the next generation are met. In the event of a tie, the individual with the greater crowding distance is chosen.

As much for NSGA II as for MGHA, the generation of clones in the surviving population is not acceptable, i.e. each individual is selected once.

5. Results

The algorithms were developed in C, using the C++ Builder 5.0 Borland. It was tested on a PC Pentium Core 2 Duo, 2.4 GHz and 4 GB of RAM on Windows Vista platform.

To test it four virtual mines were considered. Such instance-tests can be found at www.iceb.ufop.br/decom/prof/marcone/projects/mining.html and were adapted from Souza et al. [24].

The adaptation of such scenarios was to consider the truck’s cycle time depending on the material being transported (if waste, the route of the truck is from the waste pit to waste pile; while, if it is ore, the truck goes from the ore pit to the primary crusher); all depending on the truck capacity, which depends on the loaded material and type of truck.

Table 2 presents some characteristics of the instances. In this table, columns \(|F|\) and \(|P|\) represent respectively the number of pits and the number of control parameters of the instances. Column \(|S|\) shows the total number of shovels and the last column, \(|T|\), shows the number of trucks available.

<table>
<thead>
<tr>
<th>Instance</th>
<th>(F)</th>
<th>(P)</th>
<th>(S)</th>
<th>(T)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mine 1</td>
<td>17</td>
<td>10</td>
<td>8</td>
<td>30</td>
</tr>
<tr>
<td>Mine 2</td>
<td>17</td>
<td>05</td>
<td>8</td>
<td>30</td>
</tr>
<tr>
<td>Mine 3</td>
<td>17</td>
<td>05</td>
<td>7</td>
<td>30</td>
</tr>
<tr>
<td>Mine 4</td>
<td>17</td>
<td>10</td>
<td>7</td>
<td>30</td>
</tr>
</tbody>
</table>

Initially, the algorithm was subjected to a battery of tests to calibrate the various existing parameters. For the maximum number of generations and the total population size 150 generations and 6 individuals were used, respectively. After defining these parameters, the results were obtained after a battery of 10 runs.
Tables 3, 4, 5 and 6 show the results for the deviations from the goals of production and quality, the number of trucks used and the computational time (in seconds), respectively. For the production goal, the deviation percentage is presented and for the deviation of quality, the average percentage of deviation from the quality parameters.

The line “Lowest” shows the lowest values found for each algorithm, after 10 runs. The row “Average” shows the average values while the line “Highest” shows the highest values.

For each objective and for each of the instances are presented, too, the Box Plot graphics. For each instance, Figures 2, 3, 4 and 5 respectively show the results of the production, quality, number of trucks and computational time.

Observing the graphics Box Plot 2, 3, 4 and 5, and Tables 3, 4, 5 and 6,
Table 5: Results for the number of trucks

<table>
<thead>
<tr>
<th></th>
<th>Mine 1</th>
<th>Mine 2</th>
<th>Mine 3</th>
<th>Mine 4</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Lowest</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NSGAII-T</td>
<td>21</td>
<td>20</td>
<td>21</td>
<td>20</td>
</tr>
<tr>
<td>NSGAII-TU</td>
<td>19</td>
<td>23</td>
<td>21</td>
<td>20</td>
</tr>
<tr>
<td>AGHM-NT</td>
<td>20</td>
<td>23</td>
<td>22</td>
<td>22</td>
</tr>
<tr>
<td>AGHM-TU</td>
<td>22</td>
<td>23</td>
<td>23</td>
<td>22</td>
</tr>
<tr>
<td><strong>Average</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NSGAII-NT</td>
<td>26</td>
<td>26</td>
<td>26</td>
<td>25</td>
</tr>
<tr>
<td>NSGAII-TU</td>
<td>26</td>
<td>28</td>
<td>26</td>
<td>26</td>
</tr>
<tr>
<td>AGHM-NT</td>
<td>26</td>
<td>28</td>
<td>27</td>
<td>27</td>
</tr>
<tr>
<td>AGHM-TU</td>
<td>27</td>
<td>27</td>
<td>27</td>
<td>27</td>
</tr>
<tr>
<td><strong>Highest</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NSGAII-NT</td>
<td>30</td>
<td>30</td>
<td>30</td>
<td>30</td>
</tr>
<tr>
<td>NSGAII-TU</td>
<td>30</td>
<td>30</td>
<td>30</td>
<td>30</td>
</tr>
<tr>
<td>AGHM-NT</td>
<td>30</td>
<td>30</td>
<td>30</td>
<td>30</td>
</tr>
<tr>
<td>AGHM-TU</td>
<td>30</td>
<td>30</td>
<td>30</td>
<td>29</td>
</tr>
</tbody>
</table>

Table 6: Computational Time

<table>
<thead>
<tr>
<th></th>
<th>Mine 1</th>
<th>Mine 2</th>
<th>Mine 3</th>
<th>Mine 4</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Lowest</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NSGAII-NT</td>
<td>53.1</td>
<td>17.3</td>
<td>15.8</td>
<td>15.9</td>
</tr>
<tr>
<td>NSGAII-TU</td>
<td>76.2</td>
<td>17.6</td>
<td>15.6</td>
<td>15.8</td>
</tr>
<tr>
<td>AGHM-NT</td>
<td>80.5</td>
<td>17.6</td>
<td>15.7</td>
<td>16.2</td>
</tr>
<tr>
<td>AGHM-TU</td>
<td>86.5</td>
<td>17.3</td>
<td>15.7</td>
<td>16.9</td>
</tr>
<tr>
<td><strong>Average</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NSGAII-NT</td>
<td>89.7</td>
<td>18.1</td>
<td>17.5</td>
<td>18.5</td>
</tr>
<tr>
<td>NSGAII-TU</td>
<td>135.7</td>
<td>18.7</td>
<td>16.9</td>
<td>17.8</td>
</tr>
<tr>
<td>AGHM-NT</td>
<td>144.4</td>
<td>18.4</td>
<td>17.3</td>
<td>17.6</td>
</tr>
<tr>
<td>AGHM-TU</td>
<td>171.8</td>
<td>19.5</td>
<td>17.0</td>
<td>18.8</td>
</tr>
<tr>
<td><strong>Highest</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NSGAII-NT</td>
<td>125.0</td>
<td>19.6</td>
<td>21.4</td>
<td>23.9</td>
</tr>
<tr>
<td>NSGAII-TU</td>
<td>252.9</td>
<td>19.6</td>
<td>19.8</td>
<td>21.3</td>
</tr>
<tr>
<td>AGHM-NT</td>
<td>251.3</td>
<td>19.4</td>
<td>18.8</td>
<td>22.1</td>
</tr>
<tr>
<td>AGHM-TU</td>
<td>239.5</td>
<td>21.8</td>
<td>20.7</td>
<td>22.5</td>
</tr>
</tbody>
</table>
it can be seen that the algorithm NSGAII-NT in most cases, had a better performance than the others. This is due to the fact that in most cases, it produced the lowest median, and produced most of the solutions with values close to the best known values.

In addition, NSGAII-NT also found for instances “Mine 1”, “Mine 2” and “Mine 3”, at least for one goal, the best value in 100% of the runs. However, for Mine 4, the same occurs in only 70.0% of the executions. On the other hand, the other algorithms failed to find the best known values, for at least one goal in more than 90% of the runs.

6. Conclusion

This paper deals with the Open-Pit-Mining Operational Planning problem (OPMOP) involving the dynamic allocation of trucks. Due to the combinatorial complexity of OPMOP, we proposed two multi-objective genetic
Figure 3: Box Plot for the quality goal
Figure 4: Box Plot for the number of trucks goal
Figure 5: Box Plot for the computational time
algorithms, NSGAII and MGHA. Both use a local-search algorithm, based on the Variable Neighborhood Descent method for solving the problem. For each algorithm, two variants were proposed with respect to the calculation of the fitness function.

Both variants are intended to minimize the deviations of production and quality. However, the first variant (NT) also includes minimizing the number of trucks, while the second (TU), maximizing the truck usage rate.

As this work explores multi-objective optimization, at the end of the algorithm executions, a set of efficient solutions is presented. These solutions either prioritize some goal or they are balanced between the various values of goals. The decision-maker becomes responsible for choosing which alternative is best suited to the operational reality of the mine at that moment.

When being compared, the variant NSGAII-NT was the best, because it generated solutions closer to best known values in shorter processing times.

Acknowledgements

The authors acknowledge CAPES, FAPEMIG and CNPq for the financial support of this research.

References


[22] Guimaraes I, Pantuza Jr G, Souza MJF. A computer simulation model to validate the results of dynamic allocation of trucks to service quality goals and production in open pit mines (in portuguese). In: Proceedings of XIV Simpsoio de Engenharia de Produo – SIMPEP. Bauru, Brazil; 2007.,


