Support Material

Multiobjective Evolutionary Algorithms for Operational Planning Problems in Open-Pit Mining

Genetic and Evolutionary Computation Conference 2017

Rafael F. Alexandre Felipe Campelo
João A. Vasconcelos

July 17, 2017

1 Problem Definition

This section presents a multiobjective model for the *Open-Pit Mining Operational Planning Problem* (OPMOPP) with emphasis on dispatch trucks. The model is set to perform a dynamic allocation of trucks in an open-pit mine. It enables the definition of the mining pits, each with specific quality characteristics. The shovels have their productivity values preset and are allocated to specific mining pits. Each available truck has a maximum payload and speed, and can only be dispatched to a mining pit with a compatible shovel. The model also allows for multiple crushers, each with predetermined expected qualities, and for more than one waste dump where the waste produced during operation should be sent.

To facilitate the understanding of the mathematical model, the optimization variables are defined first, in section 1.1. After that, the parameters, objective functions and constraints of the mathematical model are presented in section 1.2.

1.1 Optimization Variables

The proposed representation creates a sequence of dispatches for each available truck in the fleet and can be defined as $\widetilde{S} = [\overline{V}|\widetilde{M}]$, where $\overline{V}$ is a column vector of size $|T|$, and $\widetilde{M}$ a $|T| \times j$ matrix, with $|T|$ and j representing the number of trucks and the number of dispatches, respectively. $\overline{V} \in \{0,1\}^{|T|}$ is the vector of optimization variables that represent truck availability, with the tth position of this vector (v_t) indicating whether the truck is in operation ($v_t = 1$) or not. Each cell m_{tj} of dispatch matrix ($\widetilde{M}$) represents the identifier number of the j^{th} destination of the t^{th} truck.

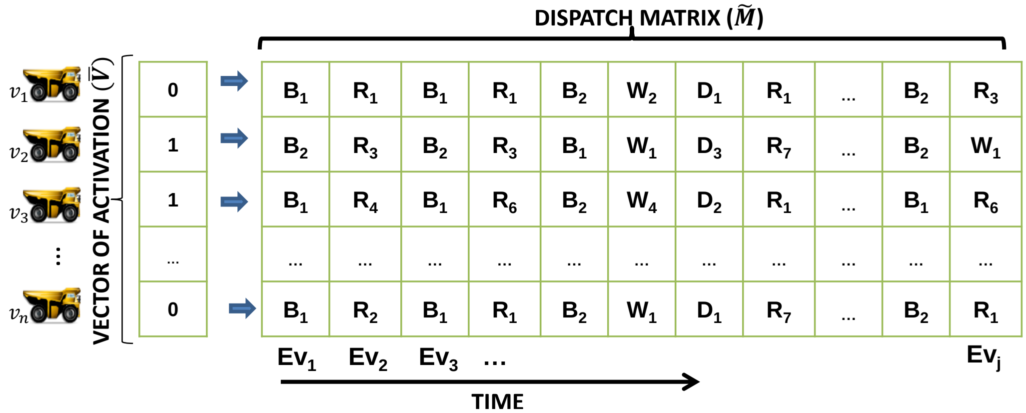

Figure 1: Representation of a candidate solution for the multiobjective OP-MOPP. The activation vector indicates which trucks are in operation, and the matrix of dispatches shows the visiting sequence of each truck.

Figure 1 presents a possible solution for a hypothetical scenario. B_b, R_r, W_w and D_d refer to the identifier number of bth crusher, rth ore pit, wth waste pit and dth waste dump, respectively, and indicate to where the truck v_t should move at each event. For better understanding, consider the row referring to the third truck, which is available for operation ($v_3 = 1$). This truck is initially at site B_1. After receiving the first dispatch, it moves from B_1 to R_4, and so on. The sequence of routes taken by this truck is $\{B_1 \rightarrow R_4,\ R_4 \rightarrow B_1,\ B_1 \rightarrow R_6,\ R_6 \rightarrow B_2, \ldots,\ B_1 \rightarrow R_6\}$.

1.2 Optimization Model

The mathematical formulation for this constrained optimization problem is expressed by equations (1)–(3).

$$\text{Minimize: } \overline{F}(\widetilde{S}) = [-f_1(\widetilde{S}), f_2(\widetilde{S}), -f_3(\widetilde{S})] \tag{1}$$

$$\text{subject to: } g_i(\widetilde{S}) \leq 0;\ i = 1, \ldots, 4 \tag{2}$$

$$h(\widetilde{S}) = 0; \tag{3}$$

where ($\overline{F}(\cdot) : \tilde{S} \mapsto \mathbb{R}^3$) is the vector of objective functions, $g_i(\cdot) : \widetilde{S} \mapsto \mathbb{R}$ is the i^{th} inequality constraint, and $h(\cdot) : \widetilde{S} \mapsto \{0, 1\}$ is the equality constraint. Before presenting the formulas for all functions from equations (1)–(3), let the following quantities be defined:

- B: set of crushers;
- C: set of shovels;
- D: set of waste dumps;
- F: set of pits, $F = R \bigcup W$;
- R: set of ore pits;
- W: set of waste pits;
- Q: set of chemical elements in the ore;
- T: set of trucks;
- x_{rb}: ore received in the b^{th} crusher from the r^{th} ore pit (tonnes);
- x_{wd}: waste received in the d^{th} waste dump from w^{th} waste pit (tonnes);
- x_f: production of the f^{th} pit (tonnes);
- cap_t: payload of the t^{th} truck (tonnes);
- it_c: idle time of the c^{th} shovel (minutes);
- tot: total operation time of the mine (minutes);

- $t_c = tot - it_c$: total time the c^{th} shovel operates (minutes);
- ql_{qb}: lower limit of the amount of concentration (%) of the q^{th} chemical element in the b^{th} crusher;
- qu_{qb}: upper limit of the amount of concentration (%) of the q^{th} chemical element to the b^{th} crusher;
- q_{qr} concentration (%) of the q^{th} chemical element in the r^{th} pits of the ore;
- cl_c: is the lower limit production (tonnes) of the c^{th} shovel;
- cu_c: is the upper limit production (tonnes) of the c^{th} shovel;
- $y_{cf} \in \{0, 1\}$: One if c shovel operates in f pit, zero otherwise;
- $y_{tf} \in \{0, 1\}$: One if t^{th} truck can operate in f^{th} pit, zero otherwise;
- $g_{tc} \in \{0, 1\}$: One if t^{th} truck is compatible with c^{th} shovel, zero otherwise.

The objective functions for the problem in focus are given in eqs. (4)–(6). The first objective is to maximize production at the mine, be it ore or sterile. The second is to minimize the payload capacity of the fleet of trucks in operation, that is, to reduce the number of trucks in operation. The third objective is to maximize the use of the shovels, reducing their idle time during operation. Regarding the second objective, it is important to notice that, as there are trucks with different payload capacities, it is necessary to take their size into consideration, since removing trucks of a certain size will impact on the idle shovels responsible for loading those trucks. In operational terms, the removal of two trucks of x payload capacity should not be more interesting (in terms of the modeling of this objective) than just removing one of $2x$. It is important to highlight at this point that x_{rb}, x_{wd}, x_f and t_c are calculated as a function of $\widetilde{S}$.

$$f_1(\widetilde{S}) = \sum_{\forall r \in R} \sum_{\forall b \in B} x_{rb}(\widetilde{S}) + \sum_{\forall w \in W} \sum_{\forall d \in D} x_{wd}(\widetilde{S}) \tag{4}$$

$$f_2(\widetilde{S}) = \sum_{\forall t \in T} s_{t1} * cap_t \tag{5}$$

$$f_3(\widetilde{S}) = \sum_{\forall c \in C} t_c(\widetilde{S}) \tag{6}$$

These objectives are subject to constraints (7)–(14), which define key aspects of the operating environment of a mine. The constraints represent *limits of chemical quality deviation* (7)–(8); *limits of pit productivity* (9)–(10); *compatibility between shovel and truck* (11); and *non-negativity constraints* (12)–(14). The evaluation of objective functions and constraints depends on a simulator based on discrete events, similar to that found in

Mena *et al.*

[15].

The simulator considers the displacement time of trucks, queue time and loading and discharging times.

$$g_1(\widetilde{S}) = \sum_{\forall r \in R} (ql_{qb} - q_{qr}) * x_{rb}(\widetilde{S}) \leq 0 \ \forall q \in Q; b \in B \tag{7}$$

$$g_2(\widetilde{S}) = \sum_{\forall r \in R} (q_{qr} - qu_{qb}) * x_{rb}(\widetilde{S}) \leq 0 \forall q \in Q; b \in B \tag{8}$$

$$g_3(\widetilde{S}) = \sum_{\forall c \in C} cl_c * y_{cf} - x_f(\widetilde{S}) \leq 0 \ \forall f \in F \tag{9}$$

$$g_4(\widetilde{S}) = x_f(\widetilde{S}) - \sum_{\forall c \in C} cu_c * y_{cf} \leq 0 \ \forall f \in F \tag{10}$$

$$h_1(\widetilde{S}) = y_{cf} + y_{tf} - 2g_{tc} = 0 \tag{11}$$

$$|B|,\ |C|,\ |D|,\ |Q|,\ |R|,\ |T|,\ |W| \ > \ 0 \tag{12}$$

$$cap_t,\ tot,\ ql_{qb},\ qu_{qb} \ > \ 0 \ \forall t \in T; q \in Q; b \in B \tag{13}$$

$$q_{qr},\ cl_c,\ cu_c \ > \ 0 \ \forall q \in Q; r \in R; c \in C \tag{14}$$

$$ql_{qb} > qu_{qb},\ cl_c > cu_c > 0 \ \forall q \in Q; b \in B; c \in C \tag{15}$$

2 Multiobjective Evolutionary Algorithms

Multiobjective Evolutionary Algorithms [6, 18] represent a family of heuristics that perform an adaptive sampling of the search space by means of a population of candidate solutions. The MOEA procedure consists of iteratively updating this population by means of different implementations of variation and selection operators. These methods can be easily adapted to solve constrained problems, as well as to handle a diversity of problem domains, which allows for their straightforward adaptation to the multiobjective OPMOPP.

Coello *et al.* [5] present a brief description of the main MOEAs currently in use, highlighting the NSGA-II [7] and the SPEA2 [19], which have been successfully applied to a variety of engineering problems [3, 11]. These algorithms tackle multiobjective optimization problems directly, without the need of scalarization techniques [6]. A detailed description of both the NSGA-II and the SPEA2 can be found in the references provided, and is not provide here for the sake of brevity. The user-defined parameters of each method are defined in Section 4.1.

Successful MOEA implementations require problem-specific representation systems and variation operators. In the following sections we describe the new variation operators proposed for solutions coded according to representation presented in Section 1.1.

2.1 Constructive Heuristic

To ensure that the initial population contains feasible candidate solutions, a constructive heuristic is employed. Feasibility at this time is guaranteed only with regard to dispatches for valid locations, i.e., according to 11. The set of initial solutions is generated according to Algorithm 1.

Algorithm 1: CONSTRUCTIVE HEURISTIC

```
Input: #desired solutions (nSol); scenario (Mine); #dispatches (J)
Output: P: initial population
P ← ∅
nT ← getNumberTrucks(Mine)                 // number of available trucks
for k=1 to nSol do                            // Generate nSol solutions
    for t = 1 to nT do                                  // For each truck
        s^k_t1 ← randBinary()             // sample random binary value
        cur_Place ← init_Place(Mine, t);         // start place for truck
        for j=2 to (J + 1) do
            s^k_tj ← cur_Place;
            // assign random valid destination
            cur_Place ← nextRandPlace(t, cur_Place, Mine);
    P ← P ⋃ S̃^k
return P
```

In this pseudocode, $getNumberTrucks(Mine)$ returns the number of available trucks. The outer iterative loop creates $nSolutions$ initial solutions. $randBinary()$ returns a random binary value that defines whether the truck will be available for operation or not. Function $init_Place(Mine, t)$ receives the simulation scenario and a truck as input parameters, and returns the starting place of that truck, which is set in the definition of mine scenario. All trucks begin operation in any of the crushers available for the scenario. Function $nextRandPlace(t, cur_Place, Mine)$ receives a truck, its current position, and the simulation scenario, and returns a randomly selected valid destination for that truck.

2.2 Variation Operators: Crossover and Mutation

The crossover operator proposed for this representation is based on recombination operators [6, 8]. The cutoff crossover (1PX) considers two candidate solutions $\widetilde{S}^1$ and $\widetilde{S}^2$ represented by $I_g \times J_g$ matrices. An odd integer $p \in [1, J_g]$ is randomly drawn from a discrete uniform random variable, and used to generate two new solutions $\widetilde{S}^3$ and $\widetilde{S}^4$ from $\widetilde{S}^1$ and $\widetilde{S}^2$. $\widetilde{S}^3$ is generated by combining the first p columns from $\widetilde{S}^1$ and the final $J_g - p$ columns of $\widetilde{S}^2$, and $\widetilde{S}^4$ is generated with the p first columns of $\widetilde{S}^2$ and the last $J_g - p$ columns from $\widetilde{S}^1$. Unlike [1], the binary crossover in $\overline{V}$ was not considered because it did not present significant differences.

Two mutation operators are applied to solutions. The first one is the *Bitwise Mutation* [6], which is applied to the first column of the matrix (availability of trucks). This operator simply flips randomly selected bits from this column. The remaining columns (dispatch of trucks) use *Flip Mutation* [4]. In this procedure, each value selected for mutation receives a new value drawn from an alphabet composed of feasible states for the corresponding truck. This operator is applied, with a certain probability of occurrence p_m, to the candidate solutions generated by the crossover operator. For each line I_g of a given candidate solution, up to k positions can be (randomly) selected for mutation within the interval $[2, J_g]$. For each selected position a new destination, sampled from the set of valid targets for the corresponding truck, is assigned. In this way the operational feasibility (constraint 11) of the resulting candidate solution is ensured, which therefore simplifies the operation of the evolutionary method since the constraints related to the truck compatibility do not need to be explicitly addressed by the algorithm.

3 Multiobjective Iterated Local Search

To provide a comparison baseline for the evolutionary approaches, as well as to evaluate the potential of the specific operators proposed for the multiobjective OPMOPP, we implemented a method based on the Pareto Iterated Local Search (PILS) [9]. PILS is an adaptation of Iterated Local Search (ILS) [12] for multiobjective problems. The motivation behind this concept is the development of simple heuristics for the resolution of complex multiobjective optimization problems. The PILS, as its single-objective counterpart, is based on two principles, namely intensification through variable neighborhoods and diversification through perturbations. The algorithm proposed in this section, which we call Multiobjective Iterated Local Search (MILS), is based on PILS and has five basic steps: (i) generation of an initial population; (ii) selection of a non-dominated candidate solution from the population; (iii) perturbation of the selected solution to generate a single neighboring point; (iv) local search, which returns an improved solution, and; (v) update of the non-dominated front with the improved solution.

The operation of the MILS metaheuristic is illustrated in Algorithm 2. After generating and evaluating an initial population, the set of non-dominated solutions is determined using the ENS-BS [17] method. For $maxIter$ iterations, a non-dominated solution from is selected and the inner loop is executed. In this step, the procedures of perturbation and local search, similar to those of PILS, are applied. The front set is updated with the refined solution obtained after local search. If the solution generated after the procedures of perturbation and local search is non-dominated, it is then inserted into the set $Front$, and the *count* variable is reset. Otherwise, count is incremented by one. The inner loop is exited when $maxCount$ iterations have been executed without generating a new non-dominated solution.

The procedure defined as perturbation (line 9) aims at improving the local search procedure by generating new starting solutions through perturbations applied to $\widetilde{S}'$ solution. For the problem addressed in this work, the neighboring solutions are constructed as follows: two random integers p_1 and p_2 are generated such that $2 \leq p_1 \leq (j - j_p)$ and $p_2 = p_1 + j_p$, where j represents the number of dispatches of each truck and j_d is the number of columns to be changed. All values in the interval $[p_1, p_2]$ of the solution $\widetilde{S}'$ are changed, creating a new candidate solution.

The other procedure used by Algorithm 2 is responsible for performing local search (line 10) with the objective to explore the neighbors of the current solution. A reduced VNS (RVNS) [10] is used in this step. The RVNS is a simplified version of the Variable Neighborhood Search (VNS), where the deterministic local search procedure (the most time-consuming part of VNS) is removed in order to reduce the computational cost. As shown in Algorithm 3, this method uses the mutation operator defined in Section 2.2 on a given candidate solution. If the mutated solution $\widetilde{S}''$ dominates the current one ($\widetilde{S}'$) it replaces the original, and the counter *iter* is reset. The procedure for generating neighboring solutions is performed N times, where N is an input of the algorithm.

4 Experimental Results

In this section we define the test problems and the experimental design of the computational experiments employed to verify the ability of the NSGA-II and SPEA2 heuristics to obtain a good set of trade-off solutions for the multiobjective OPMOPP. The aims of this section are twofold: to verify whether the two algorithms, equipped with the crossover and mutation operators described earlier, will yield significantly different performances over the test scenarios considered; and to evaluate whether any of the algorithms will be able to find feasible, interesting solution trade-offs for the multiobjective OPMOPP instances considered. For this experiment we considered four benchmark instances based on those proposed by Souza *et al.* [14]. The instances, as well as the results and analysis files, are available online [2].

To consider the multi-criteria nature in the evaluation of multiobjective algorithms, regarding the convergence and diversity [16] of the solutions as well as the computational effort demanded by each algorithm, the following quality indicators are used in this work [6]: (i) the *Spread* metric, which quantifies the diversity of solutions found in the final set returned by an algorithm; (ii) the *Inverted Generational Distance* (IGD), which measures both the convergence and the spread of a given solution set; and (iii) the *Runtime* required by each run of the algorithm (in seconds). For all indicators, smaller values indicate better performance.

4.1 Algorithm Setup

All runs of the algorithms used the following parameters: Population size = 200; Maximum number of evaluations = 10,000; Crossover rate = 0.9; and

Mutation rate = 0.4. The dispatch matrices $\widetilde{M}$ have $j = 20$ columns. The initial populations were generated were using Algorithm 1, and all trucks were considered as starting their operation in the crusher. The MILS used $j_p = 4$, $maxIter = 100$, $maxCount = 20$, and $N = 10$ for the RVNS. All runs consider one hour of operation of the mine. All algorithms were coded in Java and compiled with JDK 1.6, and were tested in a PC Intel Core i7, 2.2 GHz, with 8 GB of RAM, running Windows 8.1.

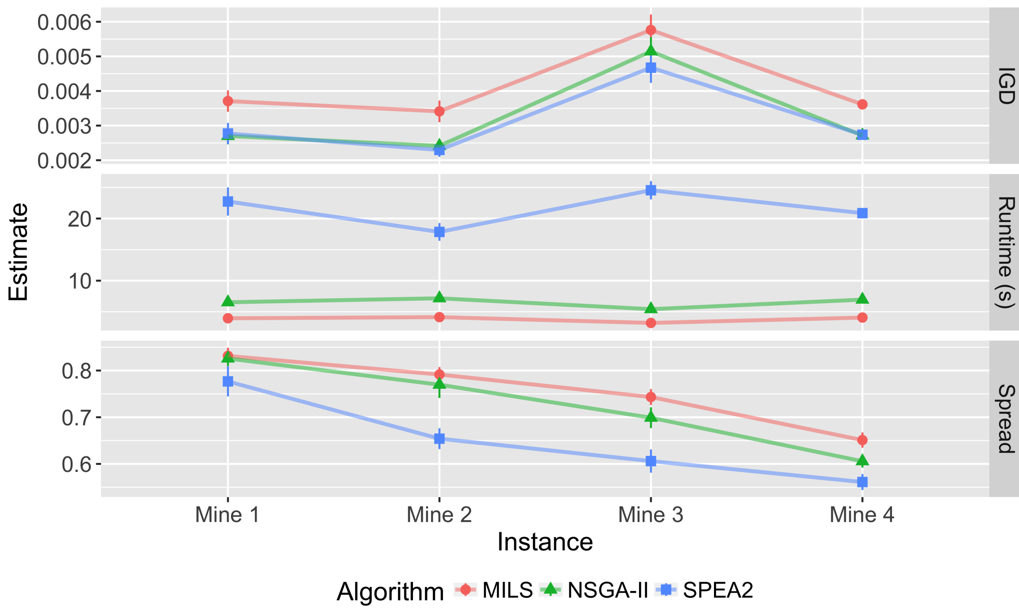

Figure 2: Results for the three methods on each test instance, for each quality metric considered. Vertical bars indicate 95% confidence intervals. Smaller is better for all indicators.

4.2 Experimental Design

The NSGA-II, SPEA2, and MILS were applied for the solution of the four available test instances on 33 independent runs, from which the quality metrics were calculated and 95% confidence intervals on the means were generated. The three algorithms were compared in terms of mean performance across instances by means of paired t-tests (with instance as the pairing variable, which means that each test was performed with three degrees-offreedom) [13] on the mean performance of each method on each instance. For each indicator three comparisons were performed (all-vs-all algorithms), and the familywise confidence level was kept at the 95% level by using Bonferroni-corrected significance levels for each test [13], i.e., each individual test was performed at a significance level of $\alpha' = 0.05/3 = 0.017$. The test assumptions were validated using residual analysis.

4.3 Results

Figure 2 shows the mean performance and confidence intervals of the results obtained in the experiment, and Table 1 displays the results of the statistical tests aimed at uncovering whether the observed differences in performance were statistically significant across instances. These results confirm the significance of the apparent trends observed in Fig. **??**. MILS was outperformed by both MOEA approaches in terms of IGD, but the small magnitude of the differences possibly means little practical significance in terms of this indicator. Regarding Runtime, SPEA2 presented a much larger computational overhead, which can be attributed to its heavy clustering approach, used to preserve diversity. MILS was also slightly better than NSGA-II in this aspect, with an expected Runtime for the family of instances about 3 seconds faster. Finally, SPEA2's clustering approach yielded relatively good gains for this MOEA over MILS in terms of the Spread indicator, but not enough to significantly outperform the NSGA-II.

5 Conclusions

We presented the definition of a multiobjective formulation for the open-pit mining operational planning problem, including considerations both on the productivity of the mine and the operational and chemical constraints involved in a mining operation. Specific data structures for encoding candidate solutions, as well as specialist variation operators for use with evolutionary optimization methods, were proposed. Additionally, a metaheuristic based on the Pareto Iterated Local Search was implemented using a local search heuristic known as Reduced Variable Neighborhood Search.

The proposed optimization approaches were tested on 4 test instances. The results suggest no large differences in convergence, as measured by the IGD indicator. Regarding the diversity of solutions, the SPEA2 presented a clearly superior performance when compared agains MILS, and equivalent with that of the NSGA-II within the detection capabilities of the experiment. This result is coherent with the known features of the two MOEAs and with the absence of explicit mechanisms for diversity preservation in MILS. The comparison of run times shows a large difference between SPEA2 and the two other methods, with the former requiring a much greater computational effort. MILS was slightly faster on average than NSGA-II, which is possibly due to it not having to perform the classification of all solutions in every iteration, unlike the MOEAs. Overall, these results appear to indicate the use of the NSGA-II as a preferred method for the solution of this class of problems.

Future possibilities include the development of tools for handling uncertainties in the mine parameters; further investigations on specialist operators; and the inclusion of preferences into the multiobjective formulation.

Acknowledgment

This work has been partially funded by grants and scholarships from Brazilian agencies FAPEMIG, CNPq, and CAPES. Prof. Felipe Campelo is a member of the Operations Research and Complex Systems Laboratory – ORCS Lab. Prof. João Vasconcelos is the director of the Evolutionary Computation Laboratory, UFMG.

References

[1] R.F. Alexandre, F. Campelo, and J.A. Vasconcelos. Solving multiobjective open-pit mining operational planning problems by using evolutionary algorithms. *International Journal of Modeling, Simulation, and Scientific Computing*. Under review.

[2] R.F. Alexandre, J.A. Vasconcelos, and F. Campelo. Additional electronic files. http://github.com/fcampelo/MOPMOPP/, 2016.

[3] S. Bandyopadhyay and R. Bhattacharya. Solving multi-objective parallel machine scheduling problem by a modified NSGA-II. *Applied Mathematical Modelling*, 37(10-11):6718–6729, 2013.

[4] F. Chicano and E. Alba. Exact computation of the expectation curves of the bit-flip mutation using landscapes theory. In *Genetic and Evolutionary Computation Conference (GECCO)*, pages 2027–2034, Dublin, Ireland, July 2011.

[5] C.A. Coello, G.B. Lamont, and D.A. Veldhuizen. Evolutionary multi-objective optimization: A historical view of the field. *Computational Intelligence Magazine, IEEE*, 1(1):28–36, 2006.

[6] C.A. Coello, G.B. Lamont, and D.A. Veldhuizen. *Evolutionary Algorithms for Solving Multi-Objective Problem*. Springer, 2nd edition, 2007.

[7] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: NSGA-II. *Evolutionary Computation, IEEE*, 6(2):182–187, 2002.

[8] Sen Bong Gee and Kay Chen Tan. Diversity preservation with hybrid recombination for evolutionary multiobjective optimization. In *IEEE Congress on Evolutionary Computation*, pages 1172–1178, 2014.

[9] M.J Geiger. The pils metaheuristic and its application to multi-objective machine scheduling. In Karl-Heinz Küfer, Heinrich Rommelfanger, Christiane Tammer, and Kristin Winkler, editors, *Multicriteria Decision Making and Fuzzy Systems - Theory, Methods and Applications*, pages 43–58. Shaker Verlag, Industriemathematik und Angewandte Mathematik, 2006.

[10] P. Hansen, N. Mladenovic, and J. M. Pérez. Variable neighbourhood search: methods and applications. *4OR*, 6(4):319–360, 2008.

[11] J.S. Lee and S.C. Park. Document clustering with multi-objective genetic algorithms, NSGA-II and SPEA2. *Information (Japan)*, 17(3):1003–1011, 2014.

[12] Helena R. Lourenço, Olivier C. Martin, and Thomas Stützle. Iterated local search: Framework and applications. In M. Gendreau and J.-Y. Potvin, editors, *Handbook of Metaheuristics*, volume 146 of *Intl. Series Oper. Research & Management Science*, pages 363–397. Springer US, 2010.

[13] D.C. Montgomery. *Design and Analysis of Experiments*. Wiley, 7th edition, 2008.

[14] M.J.F. Souza, I.M. Coelho, S. Ribas, H.G. Santos, and L.H.C. Merschmann. A hybrid heuristic algorithm for the open-pit-mining operational planning problem. *European Journal of Operational Research*, 207(2):1041–1051, 2010.

[15] C.H. Ta, A. Ingolfsson, and J. Doucette. A linear model for surface mining haul truck allocation incorporating shovel idle probabilities. *European Journal of Operational Research*, 231(3):770–778, 2013.

[16] G.G. Yen and Zhenan He. Performance metric ensemble for multiobjective evolutionary algorithms. *Evolutionary Computation, IEEE Transactions on*, 18(1):131–144, Feb 2014.

[17] X. Zhang, T. Ye, R. Cheng, and Y. Jin. An efficient approach to nondominated sorting for evolutionary multi-objective optimization. *IEEE Trans. Evol. Comp.*, 19(2):201–213, 2014.

[18] A. Zhou, B.-Y. Qu, H. Li, S.-Z. Zhao, P.N. Suganthan, and Q. Zhangd. Multiobjective evolutionary algorithms: A survey of the state of the art. *Swarm and Evolutionary Computation*, 1(1):32–49, 2011.

[19] E. Zitzler, M. Laumanns, and L. Thiele. SPEA2: Improving the Strength Pareto Evolutionary Algorithm for Multiobjective Optimization. In K.C. Giannakoglou et al., editors, *Evolutionary Methods for Design, Optimisation and Control with Application to Industrial Problems (EUROGEN 2001)*, pages 95–100. International Center for Numerical Methods in Engineering (CIMNE), 2002.

Algorithm 2: Multiobjective Iterated Local Search

Input: *nSolutions*: population size / *Mine*: simulation scenario / j: number of dispatches / j_p: number of perturbation columns / N: number of neighboring solutions $maxIter$ / $maxCount$

Output: $Front$

```
Pop ← IPOP(nSolutions, Mine, j) /*Algorithm 1*/
evaluate(Pop)
Front ← ENS-BS(Pop)
iter ← 1
while iter ≤ maxIter do
    S̃' ← selection(Front)
    count ← 1
    while count ≤ maxCount do
        S̃'' ← perturbation(S̃', j, j_p)
        S̃'' ← RVNS(S̃'', N)
        inserted ← update(Front, S̃'')
        if inserted then
            count ← 1
            S̃' ← S̃''
        else
            count ← cont + 1
    iter ← iter + 1
return Front
```

Algorithm 3: Reduced Variable Neighbourhood Search

Input: Current solution ($\widetilde{S}^{'}$); neighborhood size (N)

Output: Resulting solution ($\widetilde{S}^{'}$)

```
iter ← 1
while iter ≤ N do
    S̃'' ← MakeNeighborhood(S̃')
    evaluate(S̃'')
    if S̃'' ≺ S̃' then
        S̃' ← S̃''
        iter ← 0
    iter ← iter + 1
return S̃'
```

Table 1: Statistical tests, with significant results in boldface

Indic.	Comparison	p-value	Conf. Interval ($\alpha\prime$)
IGD	**MILS v. NSGA-II**	2.3×10^{-3}	$(9 \pm 4) \times 10^{-4}$
	MILS v. SPEA2	4.5×10^{-4}	$(10 \pm 3) \times 10^{-4}$
	NSGA2 v. SPEA2	0.384	$(1 \pm 6) \times 10^{-4}$
Spread	MILS v. NSGA-II	0.055	$(29 \pm 46) \times 10^{-3}$
	MILS v. SPEA2	0.014	$(105 \pm 98) \times 10^{-3}$
	NSGA2 v. SPEA2	0.022	$(75 \pm 84) \times 10^{-3}$
Runtime	**MILS v. NSGA-II**	6.7×10^{-4}	(-2.7 ± 0.9) seg.
	MILS v. SPEA2	0.002	(-17.7 ± 7.8) seg.
	NSGA2 v. SPEA2	0.004	(-15.0 ± 8.7) seg.