UM ALGORITMO BASEADO EM ESTRATÉGIAS EVOLUTIVAS PARA O PROBLEMA DE PLANEJAMENTO OPERACIONAL DE LAVRA

Vitor Nazário Coelho[1], Marcone Jamilson Freitas Souza[1], Igor Machado Coelho[2], Frederico Gadelha Guimarães[3], Bruno Nazário Coelho[1]

[1]Departamento de Ciência da Computação - Universidade Federal de Ouro Preto (UFOP)
Campus Universitário, Morro do Cruzeiro, CEP 35.400-000, Ouro Preto (MG), Brasil
vncoelho@gmail.com, marcone@iceb.ufop.br
brunonazario@gmail.com

[2]Departamento de Engenharia Elétrica - Universidade Federal de Minas Gerais (UFMG)
Campus Pampulha, Av. Antônio Carlos, 6627, CEP 31270-010, Belo Horizonte (MG), Brasil
frederico.g.guimaraes@gmail.com

[3]Instituto de Computação - Universidade Federal Fluminense (UFF)
Campus da Praia Vermelha, CEP 24210-240, Niterói (RJ), Brasil
imcoelho@gmail.com

Resumo

Este artigo apresenta um algoritmo evolutivo inspirado em Estratégias Evolutivas para resolução de um problema de programação inteira mista. O algoritmo proposto usa o procedimento (GRASP) para gerar a população inicial e é aplicado a um problema que requer decisões rápidas, o problema de Planejamento Operacional de Lavra com Alocação Dinâmica de Caminhões (POLAD). Para validá-lo, seus resultados são comparados com os produzidos por um algoritmo da literatura, denominado GGVNS, que não contempla o conceito de população. Resultados computacionais mostram a efetividade do algoritmo proposto.

Palavras-Chaves: Planejamento Operacional de Lavra, Programação Inteira Mista, Estratégias Evolutivas, GRASP, VND.

Abstract

This paper presents an evolutionary algorithm based on evolutionary strategies for solving a mixed integer programming problem. The proposed algorithm uses Greedy Randomized Adaptive Search Procedure (GRASP) to generate the initial population and it is applied to a problem that requires quick decisions, the Open-Pit-Mining Operational Planning problem with dynamic truck allocation (OPMOP). To validate the developed algorithm, its results are compared with those produced by a literature algorithm, called GGVNS, without the concept of population. Computational results show the effectiveness of the proposal.

Keywords: Open-pit mining, Mixed Integer Programming, Evolution strategies, Greedy Randomized Adaptive Search Procedure, Variable Neighborhood Descent.

1. Introdução

Este trabalho tem seu foco no planejamento operacional de lavra com alocação dinâmica de caminhões (POLAD). Esse problema envolve a alocação de carregadeiras às frentes de lavra (que podem ser de minério ou estéril), assim como a determinação do número de viagens que cada caminhão deve fazer a cada frente de forma que sejam atendidas tanto a meta de produção quanto a da composição mineral requerida para o minério. O objetivo é encontrar um ritmo de lavra em cada frente que minimize os desvios das metas de produção e qualidade, assim como o número de caminhões necessários ao processo produtivo.

Considera-se o sistema de alocação dinâmica de caminhões, isto é, a cada viagem realizada o caminhão pode se direcionar a uma frente diferente. Esse sistema de alocação contribui para o aumento da produtividade da frota e, consequentemente, para a redução do número de caminhões necessários ao processo produtivo.

O POLAD é um problema da classe NP-difícil e, como tal, métodos exatos de solução têm aplicabilidade restrita. A abordagem mais comum é por meio de procedimentos heurísticos.

Costa (2005) desenvolveu um algoritmo heurístico baseado em *Greedy Randomized Adaptive Search Procedures* - GRASP (FEO & RESENDE, 1995; RESENDE & RIBEIRO, 2008) e VNS (HANSEN & MLADENOVIC, 2001) para o POLAD usando seis tipos diferentes de movimentos para explorar o espaço de soluções. Foi feita uma comparação entre os resultados obtidos por esse algoritmo heurístico e os encontrados pelo otimizador LINGO, versão 7, aplicado ao modelo de programação matemática de Costa *et al*. (2004). Mostrou-se que o algoritmo heurístico foi capaz de encontrar soluções de melhor qualidade mais rapidamente.

Guimarães *et al*. (2007) apresentaram um modelo de simulação computacional para validar resultados obtidos pela aplicação de um modelo de programação matemática na determinação do ritmo de lavra em minas a céu aberto. Dessa maneira, foi possível validar os resultados da otimização, já que na modelagem de otimização não é possível tratar a variabilidade nos tempos de ciclo e a ocorrência de fila.

Em Coelho *et al*. (2008), o POLAD é resolvido por um algoritmo heurístico, denominado GVILS, que combina os procedimentos heurísticos GRASP, *Variable Neighborhood Descent* – VND (MLADENOVIC & HANSEN, 1997) e ILS (LOURENÇO *et al.*, 2003). O algoritmo GVILS faz uso de oito movimentos para explorar o espaço de soluções. Além dos desvios de produção e qualidade, procurou-se minimizar, também, o número de veículos. Usando quatro problemas-teste da literatura, o GVILS foi comparado com o otimizador CPLEX 9.1 aplicado a um modelo de programação matemática. Foram realizados testes envolvendo 15 minutos de processamento. Em dois dos problemas, o algoritmo proposto mostrou-se bastante superior; enquanto nos dois outros ele foi competitivo com o CPLEX, produzindo soluções médias com valores até 0,08% piores, na média.

Souza *et al*. (2010) propuseram um algoritmo, denominado GGVNS, que combina as metaheurísticas *General Variable Neighborhood Search* (GVNS) e o procedimento *Greedy Randomized Adaptative Search Procedure* (GRASP). Do procedimento GRASP utilizou-se a fase de construção para produzir soluções viáveis e de boa qualidade rapidamente. O GVNS foi escolhido devido a sua simplicidade, eficiência e capacidade natural de sua busca local para lidar com diferentes vizinhanças. Os autores compararam os resultados gerados pelo GGVNS com aqueles alcançados pelo otimizador CPLEX 11.01, utilizando oito problemas-teste. Os experimentos computacionais mostraram que o algoritmo proposto era competitivo com o CPLEX e capaz de encontrar soluções próximas do ótimo (com um *gap* < 1%) na maioria das instâncias, demandando um pequeno tempo computacional.

Por outro lado, a literatura tem mostrado que algoritmos evolutivos têm resolvido com eficiência vários problemas combinatórios (DE JONG *et al.*,1997; FREITAS e GUIMARÃES, 2011).

No presente trabalho investiga-se uma classe desses algoritmos, as chamadas Estratégias Evolutivas, denotadas por ES, do termo em inglês *Evolution Strategies* (BEYER & SCHWEFEL, 2002). Essas técnicas têm sido usadas na resolução de problemas inteiros ou mistos. Rajasekaran (2006) desenvolveu uma estratégia evolutiva combinada com redes neurais para resolução do problema de consistência do Concreto de Alta Performance (HPC). O algoritmo proposto foi comparado com um Algoritmo Genético – AG e com um outro baseado no procedimento *Simulated Annealing* – SA, obtendo, nos problemas-teste em questão, o melhor desempenho. Costa & Oliveira (2001) desenvolveram um algoritmo evolutivo baseado nos conceitos das ES para resolução de problemas não-lineares mistos, sendo que o algoritmo ES também foi comparado com um algoritmo AG clássico e com o procedimento SA, obtendo novamente o melhor desempenho nos problemas-teste analisados.

O algoritmo proposto, denominado GES, gera sua população inicial por meio de um procedimento parcialmente guloso e diversificado, baseado na metaheurística GRASP. Para mutação dos indivíduos, foram utilizadas as vizinhanças propostas em Souza *et al*. (2010), sendo a mutação o principal operador de busca no espaço de soluções. Para intensificar a busca, a cada geração uma pequena parte da população sofre uma busca local baseada no procedimento VND. O algoritmo proposto foi comparado ao GGVNS daqueles autores e se mostrou superior com relação à capacidade de encontrar melhores soluções mais rapidamente.

O restante deste trabalho está organizado como segue. A Seção 2 detalha o algoritmo proposto para resolver o POLAD. A Seção 3 mostra os resultados dos experimentos computacionais e a Seção 4 conclui o trabalho.

2. Metodologia

2.1. Modelo Exato

A formulação de programação matemática usada neste trabalho é a mesma de Coelho *et al.* (2008), em que se considera a função de avaliação mono-objetivo dada pela Eq. (1):

$$\min f^{PM}(s) = \sum_{j \in T} \lambda_j^- d_j^- + \sum_{j \in T} \lambda_j^+ d_j^+ + \alpha^- P_m^- + \alpha^+ P_m^+ + \beta^- P_e^- + \beta^+ P_e^+ + \sum_{l \in V} \omega_l U_l \qquad (1)$$

Na Eq. (1) busca-se minimizar os desvios positivos (d_j^+) e negativos (d_j^-) das metas de cada parâmetro de controle j da mistura, bem como minimizar os desvios positivos e negativos das metas de produção de minério e estéril, representados pelas variáveis de decisão P_m^+, P_m^-, P_e^+ e P_e^-, respectivamente. Nessa função também considera-se a minimização do número de veículos utilizados, representado pela variável binária U_l, que assume valor 1 se o veículo l for utilizado e 0, caso contrário.

As constantes λ_j^-, λ_j^+, α^-, α^+, β^-, β^+, e ω_l são pesos que refletem a importância de cada componente da função objetivo.

2.2. Modelo Heurístico

2.2.1. Representação de uma Solução

Dado um conjunto de frentes de lavra F, um conjunto de caminhões V e um conjunto de carregadeiras K, uma solução para o POLAD é representada por uma matriz $R = [Y \mid N]$, sendo Y a matriz $|F| \times 1$ e N uma matriz $|F| \times |V|$. Cada célula y_i da matriz $Y_{|F| \times 1}$ representa a carregadeira $k \in K$ alocada à frente $i \in F$. Um valor -1 significa que não existe carregadeira alocada. Se não houver viagens feitas a uma frente i, a carregadeira k associada a tal frente é considerada *inativa* e não é penalizada por produção abaixo da mínima para este equipamento de carga.

Na matriz $N_{|F| \times |V|}$, cada célula n_{il} representa o número de viagens do caminhão $l \in V$ à frente $i \in F$. Um valor 0 (zero) significa que não há viagem para aquele caminhão. O valor -1 informa a incompatibilidade entre o caminhão e a carregadeira alocada àquela frente.

Na Tabela 1, tem-se um exemplo de uma possível solução para o POLAD, observa-se que na coluna CARGA, linha F_1, a dupla $\langle Car_1, 1 \rangle$, indicando que o equipamento de carga Car_1 está alocado à frente F_1 e em operação. Na coluna CARGA, linha F_3, a dupla $\langle Car_8, 0 \rangle$ indica que o equipamento de carga Car_8 está alocado à frente F_3, mas não está em operação. Observa-se, ainda, na coluna CARGA, linha F_2, o valor $\langle D, 0 \rangle$ informando que não existe equipamento de carga alocado à frente F_2 e que, portanto, esta frente está disponível. As demais colunas representam o número de viagens a serem realizadas por um caminhão a uma frente, considerando a compatibilidade entre o caminhão e o equipamento de carga alocado à frente. As células com os valores X indicam incompatibilidade entre um caminhão e o respectivo equipamento de carga.

Tabela 1 – Representação de uma solução

	Carga	**Cam_1**	**Cam_2**	**...**	**Cam_V**
F_1	$<Car_1, 1>$	8	X	...	X
F_2	$<D, 0>$	0	0	...	0
F_3	$<Car_8, 0>$	0	0	...	0
...	...	...	...	...	...
F_F	$<Car_5, 1>$	0	9	...	3

2.2.2. Estruturas de Vizinhança

Como forma de explorar o espaço de soluções, foram utilizados os oito movimentos a seguir, os quais possuem uma boa capacidade exploratória, como relatado em Souza *et al*. (2010).

Movimento Número de Viagens - N^{NV}(s): Este movimento consiste em aumentar ou diminuir o número de viagens de um caminhão l em uma frente i onde esteja operando um equipamento de carga compatível. Desta maneira, neste movimento uma célula n_{il} da matriz N tem seu valor acrescido ou decrescido de uma unidade.

Movimento Carga - N^{CG}(s): Consiste em trocar duas células distintas y_i e y_k da matriz Y, ou seja, trocar os equipamentos de carga que operam nas frentes i e k, caso as duas frentes possuam equipamentos de carga alocados. Havendo apenas uma frente com equipamento de carga, esse movimento consistirá em realocar o equipamento de carga à frente disponível. Para manter a compatibilidade entre carregadeiras e caminhões, as viagens feitas às frentes são realocadas junto com as frentes escolhidas.

Movimento Realocar Viagem de um Caminhão - N^{VC}(s): Consiste em selecionar duas células n_{il} e n_{kl} da matriz N e repassar uma unidade de n_{il} para n_{kl}. Assim, um caminhão l deixa de realizar uma viagem em uma frente i para realizá-la em outra frente k. Restrições de compatibilidade entre equipamentos são respeitadas, havendo realocação de viagens apenas quando houver compatibilidade entre eles.

Movimento Realocar Viagem de uma Frente - N^{VF}(s): Duas células n_{il} e n_{ik} da matriz N são selecionadas e uma unidade de n_{il} é realocada para n_{ik}. Isto é, esse movimento consiste em realocar uma viagem de um caminhão l para um caminhão k que esteja operando na frente i. Restrições de compatibilidade entre equipamentos são respeitadas, havendo realocação de viagens apenas quando houver compatibilidade entre eles.

Movimento Operação Frente - N^{OF}(s): Consiste em retirar de operação o equipamento de carga que esteja em operação na frente i. O movimento retira todas as viagens feitas a esta frente, deixando o equipamento *inativo*. O equipamento retorna à operação assim que uma nova viagem é associada a ele.

Movimento Operação Caminhão - N^{OC}(s): Consiste em selecionar uma célula n_{il} da matriz N e zerar seu conteúdo, isto é, retirar de atividade um caminhão l que esteja operando em uma frente i.

Movimento Troca de Viagens - N^{VT}(s): Duas células da matriz N são selecionadas e uma unidade de uma célula passa para a outra, isto é, uma viagem de um caminhão associado a uma frente i passa para outro caminhão associado outra frente.

Movimento Troca de Carregadeiras - N^{CT}(s): Duas células distintas y_i e y_k da matriz Y tem seus valores permutados, ou seja, os equipamentos de carga que operam nas frentes i e k são trocados. Analogamente ao movimento *CG*, os equipamentos de carga são trocados, mas as viagens feitas às frentes não são alteradas. Para manter a compatibilidade entre carregadeiras e caminhões, as viagens feitas a frentes com equipamentos de carga incompatíveis são removidas.

2.2.3 Avaliação de uma Solução

Como os movimentos usados podem gerar soluções inviáveis, uma solução é avaliada por uma função f, a ser minimizada, composta por duas parcelas. A primeira delas é a função objetivo propriamente dita, f^{PM} , dada pela Eq. (1), e a segunda é composta pelas funções que penalizam a ocorrência de inviabilidade na solução corrente. Assim, a função f, definida pela Eq. (2), mensura o desvio dos objetivos considerados e penaliza o não atendimento às restrições do problema.

$$f(s) = f^{PM}(s) + f^{p}(s) + \sum_{j \in T} f_j^q(s) + \sum_{l \in V} f_l^u(s) + \sum_{k \in C} f_k^c(s) \quad (2)$$

em que:

- f^{PM}(s) é uma função que avalia s quanto ao atendimento às metas de produção e qualidade, bem como o número de caminhões utilizados (mesma do modelo de programação matemática, Subseção 2.1);
- f^{p}(s) avalia s quanto ao desrespeito aos limites de produção estabelecidos para a quantidade de minério e estéril;
- f_q^j (s) avalia s quanto à inviabilidade em relação ao j-ésimo parâmetro de controle;
- f_u^l (s) avalia s quanto ao desrespeito do atendimento da taxa de utilização máxima do l-ésimo caminhão;
- f_c^k(s), que avalia s quanto ao desrespeito aos limites de produtividade da carregadeira k.

2.2.4 Representação de um Indivíduo

Um dado indivíduo possui, além da matriz de solução $R = [Y \mid N]$ definida na Subseção 2.2.1, dois vetores de parâmetros de mutação.

O primeiro vetor diz a probabilidade de aplicação de cada um dos movimentos descritos na Subseção 2.2.2; logo, ele é um vetor de números reais, em que cada posição i diz a probabilidade de aplicação de um dado movimento. Já o segundo vetor de parâmetros que compõe a representação de um indivíduo é um vetor de números inteiros e regula a intensidade da perturbação, ou seja, cada posição i deste vetor limita o número de aplicações de um dado movimento, caso ele venha a ser aplicado.

Desta forma, tem-se um vetor de probabilidades P, tal que:

$$P = [p_1, p_2, ..., p_i, ..., p_8] \quad (3)$$

sendo $p_i \in [0, 1]$, $p_i \in \Re$.

Já para o vetor de aplicações, tem-se:

$$A = [a_1, a_2, ..., a_i, ..., a_8] \quad (4)$$

sendo $a_i \in [0, nap_i], a_i \in Z^+$, com nap_i representando o número máximo de aplicações para um dado movimento i.

2.3 Algoritmo Proposto

O algoritmo proposto neste trabalho, denominado *GES*, consiste na combinação dos procedimentos heurísticos *GRASP* e segue os passos de uma Estratégia Evolutiva. Seu pseudocódigo está esquematizado na Figura 1.

Entrada: γ, *IterMax*, Função $f(.)$
Saída: População *Pop*

1 **para** $i \leftarrow 1$ **até** μ **faça**
2 $s_w \leftarrow$ ConstróiSoluçãoEstéril()
3 Gere um número aleatório $\gamma \in [0, 1]$
4 $s_i \leftarrow$ ConstróiSoluçãoMinério(s_w, γ)
5 $ind_i \leftarrow s_i +$ ConstróiVetorMutação()
6 $Pop[i] = ind_i$
7 **fim**
8 **enquanto** *critério de parada não satisfeito* **faça**
9 **para** $i \leftarrow 1$ **até** λ **faça**
10 Gere um número aleatório $j \in [1, \mu]$
11 $ind_i \leftarrow Pop[j]$
12 $ind_i \leftarrow$ MutaParametros(ind_i, σ_{real}, $\sigma_{binomial}$)
13 $ind_i \leftarrow$ AplicaMutação(ind_i)
14 $PopFilhos[i] = ind_i$
15 **fim**
16 **para** $i \leftarrow 1$ **até** κ **faça**
17 Gere um número aleatório $i \in [1, \lambda]$
18 VND($PopFilhos[i]$)
19 **fim**
20 $Pop =$ Seleção(Pop,$PopFilhos$)
21 **fim**
22 **retorna** *Pop*

Figura 1 – Algoritmo *GES*

A população inicial do algoritmo (linhas 1 a 7 da Figura 1) é composta por μ indivíduos e é criada em duas etapas.

Na primeira, realiza-se a construção da matriz de solução $R = [Y \mid N]$ de cada indivíduo da população. Para esta etapa são utilizados os procedimentos *ConstróiSoluçãoEstéril* e *ConstróiSoluçãoMinério* descritos em Souza *et al*. (2010). A obtenção de uma população inicial diversificada é de extrema importância para a convergência do algoritmo; logo, o parâmetro γ que define o tamanho da lista restrita de candidatos varia em cada um dos indivíduos.

Na segunda etapa (linha 5 da Figura 1), é feita a construção dos vetores de mutação de cada um dos indivíduos. Para o vetor P de probabilidades utiliza-se uma Distribuição Normal. Já para o vetor de aplicações A utiliza-se uma Distribuição Binomial.

Na linha 12 da Figura 1 é acionado o procedimento de mutação dos parâmetros para um dado

indivíduo, cujo pseudocódigo está descrito na Figura 2.

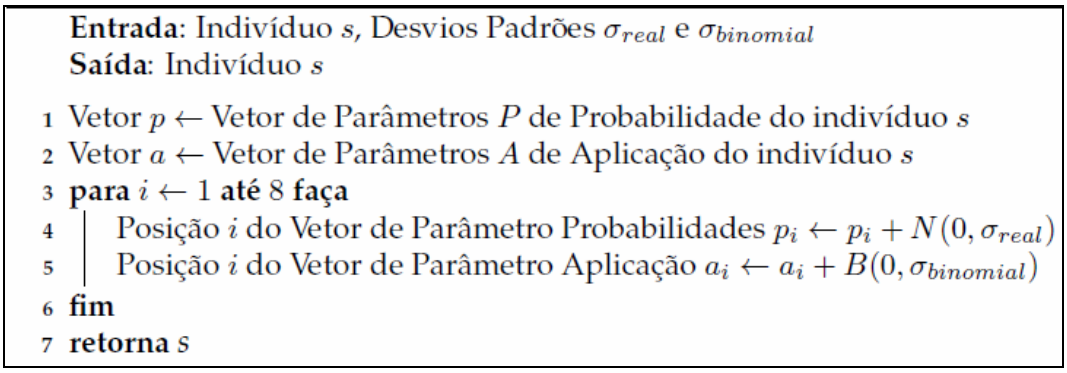

Figura 2 – Algoritmo MutaParâmetros

Para cada posição *i* dos vetores de parâmetros da Figura 2 aplica-se uma Distribuição Normal ou Binomial, ambas centradas com média zero e desvio-padrão σ_{real} e $\sigma_{binomial}$, respectivamente. Este procedimento pode ser visto nas linhas 4 e 5 desta figura. Para a mutação do vetor de probabilidades *P* aplica-se uma perturbação seguindo uma Distribuição Normal com desvio σ_{real}; já para a mutação do vetor aplicações *A* aplica-se uma perturbação seguindo uma Distribuição Binomial com desvio $\sigma_{binomial}$.

O procedimento AplicaMutação, linha 13 da Figura 1, está descrito na Figura 3.

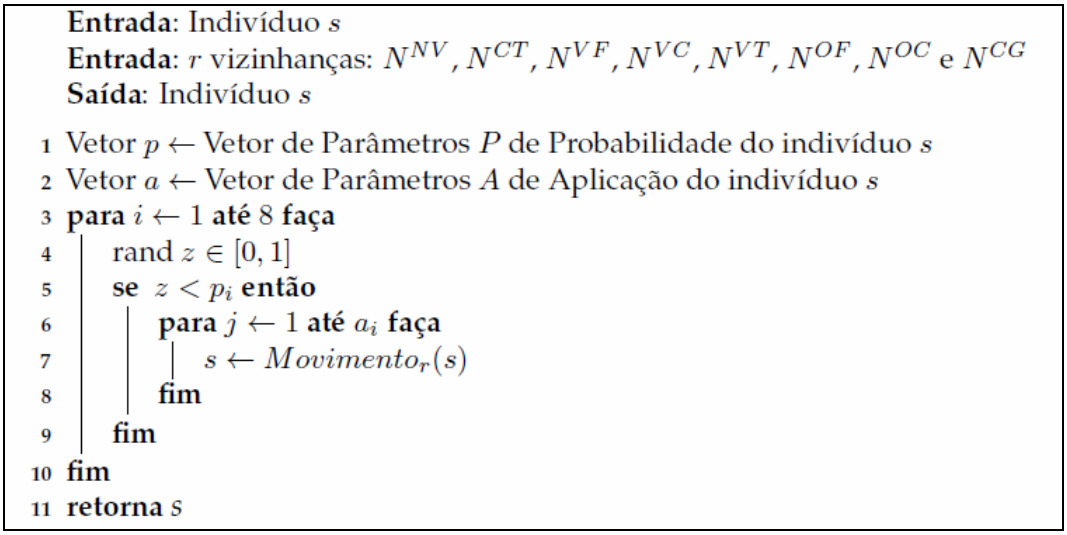

Figura 3 – Algoritmo AplicaMutação

Na linha 4 da Figura 3 é gerado um número aleatório $z \in [0,1]$ e, em seguida, é verificado se este número satisfaz a probabilidade p_i do vetor de probabilidades. Caso afirmativo, aplica-se a_i vezes um dos oito movimentos (seção 2.2.2) referentes à posição *i*. A ordem da vizinhança neste vetor de parâmetros é escolhida aleatoriamente.

O procedimento VND de busca local (linha 18 da Figura 1) é opcional, sendo aplicado apenas em algumas versões do algoritmo proposto. Quando este procedimento é acionado, realiza-se um busca local de acordo com o procedimento VND (descrito pela Figura 4) em κ filhos. Para esta busca local, usa-se, apenas, um grupo restrito dos movimentos descritos na seção 2.2.2, no caso, apenas aqueles que definem as vizinhanças N^{CG}, N^{NV}, N^{VC} e N^{VF}. A justificativa para essa restrição é que a busca local do algoritmo é muito custosa computacionalmente. Estrategicamente, a busca local opera nessas vizinhanças em uma ordem aleatória, definida a cada chamada. Esse resultado foi alcançado após uma bateria de testes preliminares, a qual

mostrou que não havia uma ordem de vizinhança que resultasse, sempre, na geração de soluções melhores. Na seção 3 o resultado da adição deste procedimento é mostrado.

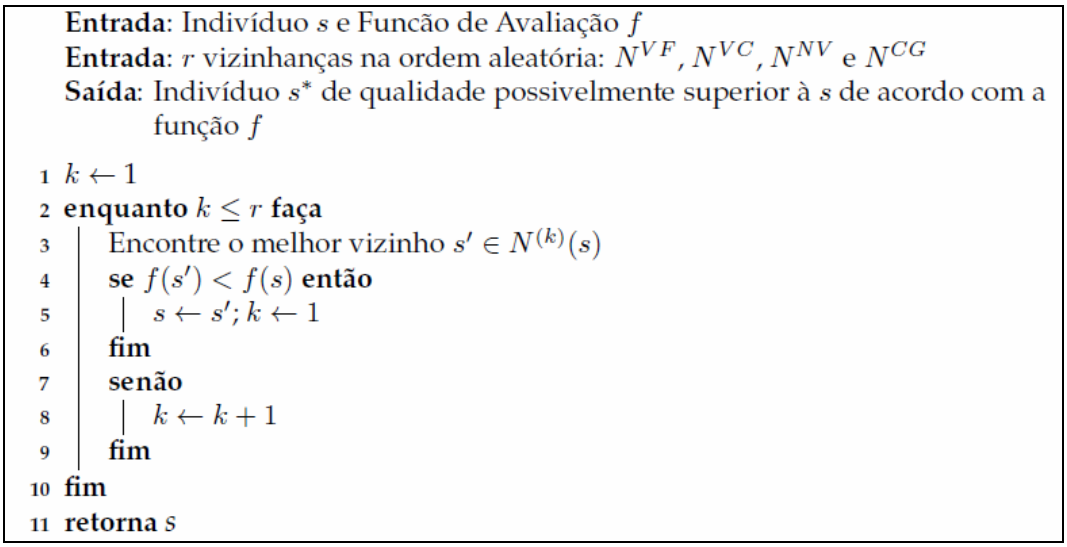

Figura 4 – Algoritmo *VND*

O procedimento de Seleção (linha 20 da Figura 1) pode ser qualquer estratégia de seleção desejada, desde que esta retorne uma população de tamanho μ. Foram utilizadas duas formas básicas de competição, ambas com a mesma notação de Beyer & Schwefel (2002). Na primeira delas, denotada por (μ + λ), ocorre uma competição entre pais e filhos. Nesta estratégia são selecionados os μ melhores indivíduos dentre pais e filhos. Já na segunda estratégia de seleção utilizada, denotada por (μ, λ), os indivíduos que sobrevivem para a próxima geração são os μ melhores filhos. É notório que utilizando a estratégia (μ, λ) como forma de seleção a população que sobrevive para próxima geração sofre uma considerável pressão seletiva; porém, esta pressão torna-se ainda maior quando a estratégia (μ + λ) é utilizada.

3. Experimentos Computacionais e Análises

O algoritmo *GES* proposto foi implementado em C++ usando o *framework* de otimização OptFrame, disponível em https://sourceforge.net/projects/optframe, e compilado pelo g++ 4.13, utilizando a IDE Eclipse 3.1. Os experimentos foram feitos em um microcomputador Pentium Core 2 Quad(Q6600), com 8 GB de RAM, no sistema operacional Ubuntu 10.10. Para testar o algoritmo foi usado um conjunto de 8 problemas-teste da literatura, disponíveis em http://www.iceb.ufop.br/decom/prof/marcone/projects/mining.html. Estes problemas-teste foram os mesmos utilizados em Souza *et al*. (2010) para validar o algoritmo *GGVNS*.

Os melhores resultados da literatura para os problemas-teste analisados são apresentados na Tabela 2. Na coluna "Opt." Indica-se por "√" os problemas-teste nos quais o otimizador matemático CPLEX 11.02 obteve o valor ótimo da função.

Tabela 2 – Melhores valores

Problema-Teste	Melhor da Literatura	Opt.
opm1	227,12	
opm2	256,37	
opm3	164.027,15	√
opm4	164.056,68	√
opm5	227,04	
opm6	236,58	
opm7	164.017,46	√
opm8	164.018,65	√

A Tabela 3 mostra seis variantes do algoritmo *GES*, criadas a partir de diferentes valores para seus parâmetros. As variantes *GES-VND1* e *GES-VND2* incluem o procedimento VND (descrito na Figura 4) como método de busca local. Nestas duas variantes uma parcela de κ=3 indivíduos-filho da população sofre esse procedimento de busca local. O número de indivíduos que sofrem este procedimento de busca local é pequeno em relação à população de filhos visto que, como já citado anteriormente, a busca local utilizada é muito custosa computacionalmente.

Tabela 3 – Variantes do Algoritmo *GES*

Algoritmo	μ	λ	**Seleção**	***VND***
GES1	30	160	$(\mu; \lambda)$	
GES2	30	160	$(\mu + \lambda)$	
GES3	100	600	$(\mu; \lambda)$	
GES4	100	600	$(\mu + \lambda)$	
GES-VND1	30	160	$(\mu; \lambda)$	√
GES-VND2	30	160	$(\mu + \lambda)$	√

Primeiramente, foi realizado um experimento de probabilidade empírica, *Time-to-target* (TTT) *plots*, na forma indicada por Aiex *et al*. (2007), de forma a verificar a eficiência das variantes do algoritmo elencadas na Tabela 2. Este teste mostra, no eixo das ordenadas, a probabilidade de um algoritmo em encontrar uma solução boa em um dado limite de tempo, eixo das abscissas. TTT *plots* já foi utilizado por vários autores, como Hoos & Stutzle (1998), e continua sendo defendido (RIBEIRO & REZENDE, 2011) como uma forma de caracterizar tempo de execução de algoritmos estocásticos aplicados a problemas de otimização combinatória.

Foram realizadas 120 execuções com cada uma das seis variantes do algoritmo desenvolvido. O problema-teste utilizado foi o opm1, tendo como alvo o valor 229,00 e um tempo limite de 120 segundos de processamento. A Figura 5 mostra as curvas obtidas.

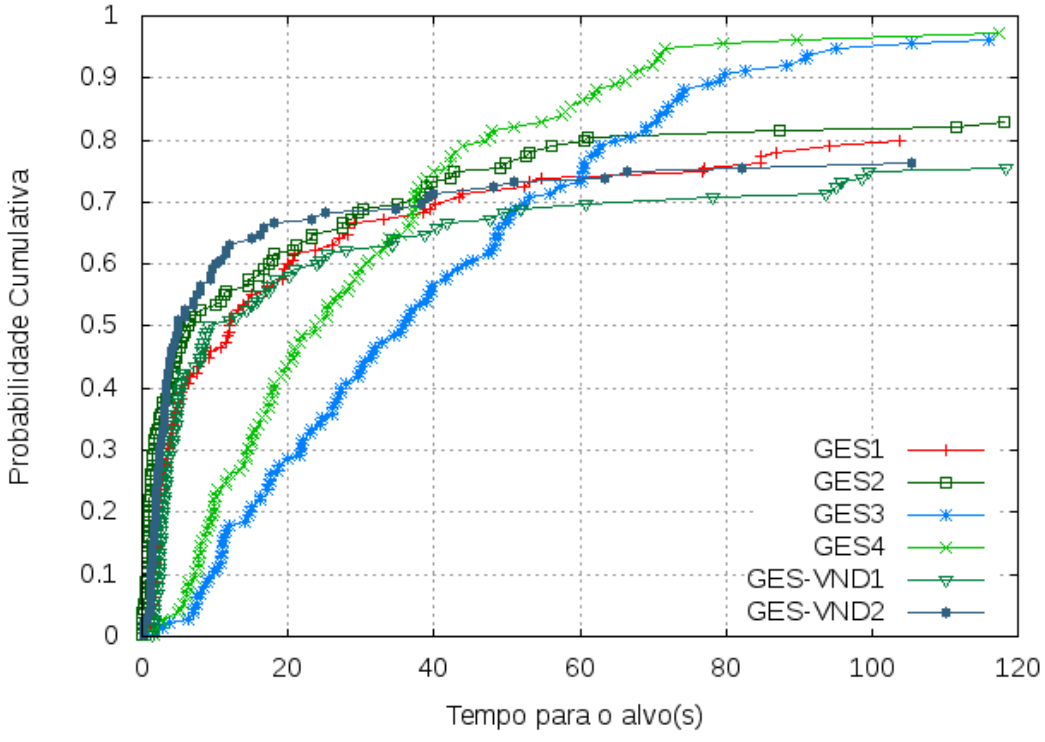

Figura 5 – Curva de Probabilidade Empírica - Instância opm1

Analisando as curvas de probabilidade empírica da Figura 5, percebe-se que as variantes que

utilizaram a estratégia de seleção ($\mu + \lambda$) prevaleceram em relação a suas versões com a estratégia (μ, λ). Este fato mostra que a competição entre pais e filhos fez com que indivíduos com um bom potencial de otimalidade permanecessem por mais gerações.

Nos instantes iniciais da busca, a variante *GES-VND2* foi capaz de gerar melhores soluções do que os outros algoritmos propostos. No entanto, a partir de 40 segundos esta variante perde seu desempenho, sendo superada pela variante *GES4*, a qual continua progredindo sistematicamente, sendo a primeira a alcançar o alvo desejado com uma probabilidade de aproximadamente 100%.

Pela Figura 5, verifica-se, também, que as variantes que utilizam o método de busca local VND, *GES-VND1* e *GES-VND2*, têm ótimo desempenho no início da busca, mas convergem prematuramente. Isto é devido ao fato de que o VND contribui para gerar super-indivíduos que causam estagnação da população depois de alguns instantes de execução do algoritmo. Por outro lado, pode-se comprovar a força do mecanismo de mutação das Estratégias Evolutivas quando é analisada a convergência das variantes que não usam o procedimento de busca local.

Desta forma, tendo em vista a robustez da variante *GES4*, foi feita uma comparação entre os algoritmos *GES*4 e o algoritmo *GGVNS*, de Souza *et al.*, (2010).

A Tabela 3 mostra a comparação entre algoritmo *GES4* e o algoritmo *GGVNS*. Nessa tabela, a coluna "Instância" indica o problema-teste utilizado. A coluna "*IMPdesv*" menciona o ganho relativo do algoritmo *GES*4 em relação ao algoritmo *GGVNS*, ou seja:

$$IMPdesv_i = \frac{\bar{f}_i^{GGVNS} - \bar{f}_i^{GES4}}{\bar{f}_i^{GGVNS}} \tag{5}$$

em que $\bar{f}_i^{GGVNS}$ representa o valor médio obtido pelo algoritmo *GGVNS* em 30 execuções na *i*-ésima instância e $\bar{f}_i^{GES4}$ o valor médio relativo a aplicação do algoritmo *GES4*. Já a coluna "*IMPbest*" indica o percentual de melhora proporcionado pelo algoritmo *GES4* em relação ao valor da melhor solução encontrada pelo algoritmo *GGVNS*, dada por $f_i^{GGVNS^*}$.

$$IMPbest_i = \frac{f_i^{GGNVS^*} - f_i^{GES4^*}}{f_i^{GGNVS^*}} \tag{6}$$

sendo $f_i^{GES4^*}$ o melhor valor encontrado por *GES4* na instância *i*.

Tabela 4 – Comparação de resultados: GGVNS x GES4

Instância	**Tempo (s)**	***GGVNS***		***GES4***		***IMPbest***	***IMPgap***
		MEDIA	**MELHOR**	**MEDIA**	**MELHOR**		
opm1	2	230,12	230,12	228,50	228,12	**0,87%**	**0,70%**
opm2	2	256,56	256,37	256,43	256,37	0,00%	**0,05%**
opm3	2	164064,68	164039,12	164044,68	164031,28	0,00%	**0,01%**
opm4	2	164153,92	164099,66	164097,61	164057,04	**0,03%**	**0,03%**
opm5	2	228,09	228,09	227,21	226,66	**0,63%**	**0,39%**
opm6	2	237,97	236,58	237,07	236,58	0,00%	**0,38%**
opm7	2	164021,89	164021,38	164020,24	164018,22	0,00%	0,00%
opm8	2	164027,29	164023,73	164022,38	164020,26	0,00%	0,00%

Analisando a Tabela 5 podemos verificar que o algoritmo *GES4* proposto foi capaz de gerar soluções de boa qualidade e baixa variabilidade em relação ao algoritmo *GGVNS*. Percebe-se

que ele conseguiu melhorar a qualidade das soluções finais em até 0,87%, e reduzir a variabilidade dessas soluções em até 0,39%. Além disso, tendo em vista a Tabela 2, pode ser observado que para o problema-teste opm5 o *GES4* foi capaz de encontrar uma solução melhor que a da literatura.

4. Conclusões

Este trabalho teve seu foco no problema de planejamento operacional de lavra considerando alocação dinâmica de caminhões (POLAD). Em virtude de sua dificuldade de solução, foi proposto um algoritmo populacional, denominado *GES*, que combina o poderio do GRASP com as Estratégias Evolutivas, percorrendo um espaço de busca de soluções inteiras.

Seis variantes desse algoritmo foram desenvolvidas e comparadas entre si. Dessas, quatro eram algoritmos baseados em Estratégia Evolutiva tendo como principal mecanismo de busca no espaço a mutação e diferiam entre si pelos parâmetros de tamanho da população e estratégia de seleção. As outras duas incluíam um procedimento de busca local, baseado em VND, na exploração do espaço de soluções. Verificou-se a convergência prematura das variantes que usam o procedimento de busca local e optou-se pela variante que mostrou ser mais eficiente em alcançar o valor alvo.

Usando problemas-teste da literatura, essa variante do algoritmo evolutivo, denotada por GES4, foi comparada com o algoritmo *GGVNS* de Souza *et al*. (2010). Os resultados mostraram que o algoritmo evolutivo proposto é competitivo, apresentando boas soluções com uma menor variabilidade em torno da média.

Para trabalhos futuros é proposto usar novas estratégias de perturbação no vetor de parâmetros de cada indivíduo, assim como variar a etapa de seleção durante a execução do algoritmo, de forma que o algoritmo entre em maior harmonia no quesito *Exploration-Exploitation*. Além disso, propõe-se uma melhoria no mecanismo de auto-adaptação, bem como um estudo no sentido de buscar uma melhor calibragem dos parâmetros das distribuições utilizadas. A adição do módulo de busca local apenas em determinadas gerações é outra estratégia que pode ser testada. Finalmente, propõe-se a implementação de uma versão paralela do algoritmo *GES* visando tirar proveito da tecnologia *multi-core*, já presente nas máquinas atuais e de fácil abstração para algoritmos populacionais.

Agradecimentos

Os autores agradecem à FAPEMIG e ao CNPq pelo apoio à execução do presente trabalho de pesquisa.

Referências

Aiex R. M., Resende, M.G.C. & Ribeiro, C.C. TTTPLOTS: a perl program to create time-to-target plots. *Optimization Letters*, Vol. 1, n.4, p.355-366, 2007.

Beyer, H. G. & Schwefel, H. P. Evolution strategies - a comprehensive introduction. *Natural Computing*, Vol. p.3-52, 2002.

Coelho, I. M., Ribas, S., & Souza, M. J. F. Um algoritmo baseado em grasp, vnd e iterated local search para a resolução do planejamento operacional de lavra. In *XV Simpósio de Engenharia de Produção* - SIMPEP, Bauru/SP, 2008.

Coelho, I. M., Ribas, S., Souza, M. J. F., & Coelho, V. N. Um algoritmo heurístico híbrido para o planejamento operacional de lavra. In *Anais do IX Congresso Brasileiro de Redes Neurais* - CBRN, 7 p., Ouro Preto, MG, 2009.

Costa, F. P. *Aplicações de técnicas de otimização a problemas de planejamento operacional de lavra em minas a céu aberto*. Dissertação, Programa de Pós-Graduação em Engenharia Mineral, Escola de Minas, UFOP, Ouro Preto, 2005.

Costa, F. P., Souza, M. J. F., & Pinto, L. R. Um modelo de alocação dinâmica de caminhões. *Revista Brasil Mineral*, Vol. 231, p.26-31, 2004.

Costa, L. & Oliveira, P. Evolutionary algorithms approach to the solution of mixed integer non-linear programming problems. *Computers & Chemical Engineering*, Vol. 25, n.10, p.257-266, 2001.

De Jong, K., David, D., Fogel, B. & Schwefel, H.P. *A history of evolutionary computation.*In: Bäck T, Fogel DB and Michalewicz Z (eds). Handbook of Evolutionary Computation, Vol. A2, n.3, p.1-12. Oxford University Press, New York, and Institute of Physics Publishing,Bristol, 1997.

Feo, T. A. & Resende, M. G. C. Greedy randomized adaptive search procedures. *Journal of Global Optimization*, Vol. 6, p.109-133, 1995.

Freitas, A. R. R.; Guimarães, F. G. Originality and Diversity in the Artificial Evolution of Melodies. In: Genetic and Evolutionary Computation Conference (GECCO-2011), 2011, Dublin. *Proceedings of the 13th annual Conference on Genetic and Evolutionary Computation*. New York: ACM Press, 2011.

Guimarães, I. F., Pantuza, G., & Souza, M. J. F. Modelo de simulação computacional para validação dos resultados de alocação dinâmica de caminhões com atendimento de metas de qualidade e de produção em minas a céu aberto. In *Anais do XIV Simpósio de Engenharia de Produção* - SIMPEP, 11 p., Bauru, CD-ROM, 2007.

Hansen, P. & Mladenovic, N. Variable neighborhood search: Principles and applications. *European Journal of Operational Research*, Vol. 130, p.449-467, 2001.

Hoos, H. H. & Stutzle, T. On the empirical evaluation of Las Vegas algorithms - Position paper. *Technical report*, Computer Science Department, University of British Columbia, 1998.

Lourenço, H. R., Martin, O. C., & Stützle, T. *Iterated local search*. In Glover, F. and Kochenberger, G., editors, Handbook of Metaheuristics. Kluwer Academic Publishers, Boston, 2003.

Mladenovic, N. & Hansen, P. A variable neighborhood search. *Computers and Operations Research*, Vol. 24, p.1097-1100, 1997.

Rajasekaran, S. Optimal mix for high performance concrete by evolution strategies combined with neural networks. *Indian Journal of Engineering and Materials Sciences*, Vol. 13, n.11, p.7-17, 2006.

Resende, M. G. C. & Ribeiro, C. C. *Greedy randomized adaptive search procedures: Advances and applications*. In Gendreau, M. and Potvin, J., editors, Handbook of Metaheuristics. Springer, 2 edition. (to appear). Available at: http://www2.research.att.com/ mgcr/doc/sgrasp2008a.pdf, 2008.

Ribeiro, C.C. & Resende, M. G. C. Path-relinking intensification methods for stochastic local search algorithms. *Journal of Heuristics*, Springer Netherlands, p.1-22, 2011.

Souza, M. J. F., Coelho, I. M., Ribas, S., Santos, H. G., & Merschmann, L. H. C. A hybrid heuristic algorithm for the open-pit-mining operational planning problem. *European Journal of Operational Research*, Vol. 207, n.2, p.1041-1051, 2010.