BUSCA GERAL EM VIZINHANÇA VARIÁVEL COM RECONEXÃO POR CAMINHOS PARA O PLANEJAMENTO OPERACIONAL DE LAVRA

Vitor Nazário Coelho[1], Marcone Jamilson Freitas Souza[1], Igor Machado Coelho[2], Sabir Ribas[2]

[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

[2]Instituto de Computação – Universidade Federal Fluminense (UFF)
Rua Passo da Pátria, 154 – Bloco E, 3º andar – CEP 24.210-240 – Niterói (RJ), Brasil

vncoelho@gmail.com, marcone@iceb.ufop.br, imcoelho@iceb.ufop.br, sabir@iceb.ufop.br

Abstract. *This paper improves a hybrid heuristic algorithm based on the GRASP and General Variable Neighborhood Search (GVNS) metaheuristics. The improvements consist in adding a mathematical programming module to solve, to optimality, small subproblems of the main problem and a post-optimization module, based on Path Relinking approach, to find attraction basins with better optimality potential. The proposed algorithm 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 the CPLEX optimizer and those generated by the same algorithm without these improvements. Computational results show the effectiveness of the proposal.*

KEYWORDS*: Open-pit mining, Variable Neighborhood Search, Path Relinking.*

Resumo. *Este trabalho apresenta um aperfeiçoamento de um algoritmo heurístico híbrido baseado nas metaheurísticas GRASP e General Variable Neighborhood Search (GVNS). O aperfeiçoamento consiste em introduzir um módulo de programação matemática para resolver, na otimalidade, partes pequenas do problema; bem como um módulo de Reconexão por Caminhos ao final do algoritmo para buscar bacias de atração com melhores potenciais de otimalidade. Esse algoritmo é aplicado a um problema que requer decisões rápidas, o problema de planejamento operacional de lavra em minas a céu aberto com alocação dinâmica de caminhões (POLAD). Para validá-lo, seus resultados são comparados com aqueles produzidos pelo otimizador CPLEX, bem como com os gerados pelo mesmo algoritmo em uma versão sem tais aperfeiçoamentos. Experimentos computacionais mostram a efetividade da proposta.*

PALAVRAS-CHAVE*: Planejamento Operacional de Lavra, Busca em Vizinhança Variável, Reconexão por Caminhos.*

1 Introdução

Este trabalho tem seu foco no planejamento operacional de lavra com alocação dinâmica de caminhões (POLAD). Nesse problema, o objetivo é determinar o ritmo de lavra nas frentes de lavra (minério ou estéril), associando a elas carregadeiras e caminhões de forma que sejam atendidas as metas de produção e qualidade requeridas para a mistura desejada e minimizando o número de caminhões necessários ao processo de produção.

Considera-se a alocação dinâmica de caminhões, isto é, a cada viagem realizada, o caminhão pode se direcionar a uma frente diferente. Essa forma 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 e Resende, 1995) e VNS (Hansen e 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 a um modelo de programação matemática desenvolvida. 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, VND 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) desenvolveram uma versão aprimorada do algoritmo desenvolvido em (Coelho et al., 2008), denominado GGVNS. Esse algoritmo combina os procedimentos GRASP (Feo e Resende, 1995; Resende e Ribeiro, 2008) e *General Variable Neighborhood Search* – GVNS (Hansen et al., 2008b,a; Hansen e Mladenovic, 2001; Mladenovic e Hansen, 1997) para resolução do POLAD. Adicionalmente foi validado um novo modelo de programação matemática utilizando o software comercial CPLEX 11. Resultados computacionais mostraram que o algoritmo proposto foi capaz de encontrar soluções competitivas com o otimizador CPLEX em oito problemas-teste, encontrando soluções perto da otimalidade (a menos de 1%) na maioria das instâncias, e demandando um pequeno tempo computacional.

O presente trabalho aprimora o algoritmo heurístico GGVNS, de Souza et al. (2010), adicionando ao mesmo um módulo de Reconexão por Caminhos (Glover, 1996), assim como um módulo de programação matemática para atuar como busca local e resolver partes menores do problema. O algoritmo proposto combina a flexibilidade das meta-

heurísticas com o poderio de procedimentos baseados em programação matemática. 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 (método VND) para lidar com diferentes vizinhanças. O otimizador GLPK, chamado periodicamente, foi utilizado para resolver uma parte pequena do problema, sendo quase sempre possível resolvê-la na otimalidade. A fase de pós-otimização, baseada no procedimento de Reconexão por Caminhos, é utilizada na tentativa de encontrar no caminho entre soluções encontradas previamente, alguma solução possivelmente melhor. Esse algoritmo representa uma melhoria em relação ao de (Coelho et al., 2009b), além de ter sido testado em quatro problemas-teste a mais do que nesse último trabalho.

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. (2009a). Nesta formulação, considera-se a função de avaliação mono-objetivo dada pela Eq. (2):

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

Na Eq. (2) 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 vale 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

Uma solução é representada por uma matriz $R = [Y|N]$, sendo Y a matriz $|F| \times 1$ e N a matriz $|F| \times |V|$. Cada célula y_i da matriz $Y_{|F|\times 1}$ representa a carregadeira k alocada à frente i. O 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 $l \in V$ a uma frente $i \in F$. O 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.

2.2.2 Geração de uma solução inicial

Uma solução inicial para o problema é feita em duas etapas. Na primeira, realizamos a alocação das carregadeiras e a distribuição das viagens às frentes estéril, e na segunda, às frentes de minério. Esta estratégia é adotada tendo em vista que nas frentes de estéril o importante é atender à produção e não é necessário observar a qualidade.

Na primeira etapa utilizamos uma heurística gulosa, cujo pseudocódigo está descrito no Algoritmo 1.

Algoritmo 1: ConstróiSoluçãoEstéril()

Entrada: T, S, W
Saída: Solução de estéril S_w
$T \leftarrow$ Conjunto de caminhões ordenados por suas capacidades (o primeiro é o de maior capacidade).
$S \leftarrow$ Conjunto de carregadeiras ordenadas pelas produtividades (a primeira é a de maior produtividade).
$W \leftarrow$ Conjunto de frentes de estéril ordenadas pelas massas (a primeira é a de maior massa).
enquanto *a produção de estéril for menor que a produção recomendada* ***e*** *existirem frentes de estéril não utilizadas* **faça**
  Selecione a primeira frente de estéril i do vetor W;
  se *não há carregadeira alocada à frente* i **então**
    se *Todas as carregadeiras estão alocadas* **então** Remova a frente i de W
    senão
      Atualize S_w alocando a maior carregadeira disponível à frente i ;
    fim
  fim
  se *A frente* i *não foi removida de* W **então**
    Encontre um caminhão $l \in T$ tal que: a) Seja compatível com a carregadeira alocada à frente i; b) Seja possível realizar mais uma viagem; c) Sua capacidade não viole a produção máxima da carregadeira;
    se *O caminhão* l *existe* **então** Atualize S_w, alocando a maior quantidade possível de viagens do caminhão l à frente de estéril i ;
    senão
      Remova a frente i de W ;
    fim
  fim
fim
Retorne S_w;

Na segunda etapa, utilizamos uma heurística que aplica *GRASPmax* vezes a fase de construção do procedimento *GRASP* e retorna a melhor das soluções construídas, desta feita incluindo-se as frentes de minério. A justificativa para esse procedimento é que a busca local de nosso algoritmo é muito custosa computacionalmente. Assim, a mesma requer uma boa solução inicial, o que, de acordo com Lourenço et al. (2003), aceleraria a convergência para um ótimo local.

Para cada construção, utilizamos uma função guia g que relaciona os valores de desvio de qualidade em relação à meta. De acordo com esta função, é mais indicado selecionar uma frente que minimize os desvio de qualidade dos parâmetros de controle.

Inicialmente, todas as frentes i candidatas são ordenadas de acordo com os valores de g_i e inseridas em uma lista de candidatos LC. De LC é extraída uma lista restrita de candidatos LRC contendo as frentes de minério mais bem qualificadas de acordo com a função

guia. A cardinalidade desta lista, isto é, $\lceil \gamma \times |LC| \rceil$ é definida pelo parâmetro $\gamma \in [0, 1]$. A estratégia utilizada para escolher uma frente i consiste em atribuir, primeiramente, uma classificação probabilística para cada frente candidata da LRC. A função $bias(r) = 1/(r)$ é associada à frente que está na r-ésima posição na classificação. Cada frente candidata é, então, escolhida com probabilidade $p(r) = bias(r)/\sum_{i=1,\cdots,|LRC|} bias(i)$. Em seguida, o algoritmo escolhe aleatoriamente uma frente de minério i de LRC, adicionando-a à solução parcial. O Algoritmo 2 descreve este procedimento de construção.

Algoritmo 2: ConstróiSoluçãoMinério()

Entrada: S_w, γ, g, T, S
Saída: Solução S_0
$S_0 \leftarrow S_w$
$T \leftarrow$ Conjunto de caminhões ordenados pelas suas capacidades (o primeiro é o de menor capacidade).
$S \leftarrow$ Conjunto de carregadeiras ordenadas por suas produtividades (a primeira é a que de maior produtividade).
enquanto *a produção de minério for menor que a produção recomendada* ***e*** *existirem frentes de minério não utilizadas* **faça**
  $LC \leftarrow$ Conjunto de frentes de minério ordenadas de acordo com a função g ;
  $|LRC| = \lceil \gamma \times |LC| \rceil$;
  Selecione uma frente $i \in LRC$ de acordo com a função *bias*;
  se *não há carregadeira alocada à frente* i **então**
    se *Todas as carregadeiras estão alocadas* **então** Remova a frente i de LC
    senão
      Atualize S_0 alocando a carregadeira de maior capacidade à frente i ;
    fim
  fim
  se *A frente* i *não foi removida de* LC **então**
    Encontre um caminhão $l \in T$ tal que: a) Seja compatível com a carregadeira alocada à frente i; b) Seja possível realizar mais uma viagem; c) Sua capacidade não viole a produção máxima da carregadeira;
    se *O caminhão* l *existe* **então** Atualize S_0, alocando a maior quantidade possível de viagens do caminhão l à frente de estéril i ;
    senão
      Remova a frente i de W ;
    fim
  fim
fim
Retorne S_0;

2.2.3 Estruturas de Vizinhança

Para explorar o espaço de soluções do problema foram usados oito movimentos, apresentados a seguir, sendo os seis primeiros os de Costa (2005).

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 a uma frente passa para outro caminhão a 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.

2.2.4 Avaliação de uma solução

Como os movimentos desenvolvidos podem gerar soluções inviáveis, uma solução s é avaliada por uma função f baseada em penalidades, dada pela Eq. (2). Nesta equação, a componente f_s^M é a função objetivo propriamente dita (mesma do modelo matemático - Equação (2)), e avalia s quanto ao atendimento às metas de produção e qualidade, bem como o número de caminhões utilizados. As demais componentes penalizam a ocorrência de inviabilidade na solução. Assim, $f^p(s)$ avalia s quanto ao desrespeito aos limites de produção estabelecidos para a quantidade de minério e estéril; $f_j^q(s)$ avalia s quanto à inviabilidade em relação ao j-ésimo parâmetro de controle; $f_l^u(s)$ avalia s quanto ao desrespeito ao atendimento da taxa de utilização máxima estabelecida para o l-ésimo caminhão e $f_k^c(s)$ avalia s quanto ao desrespeito aos limites de produtividade da k-ésima carregadeira.

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

2.3 Algoritmo proposto

O algoritmo proposto, denominado *GGVNSMIP-PR*, consiste em adicionar um módulo de programação matemática e uma fase de pós-otimização ao algoritmo heurístico GGVNS de Souza et al. (2010). Ele consiste na combinação de procedimentos heurísticos GRASP (Feo e Resende, 1995; Resende e Ribeiro, 2008), GVNS (Hansen et al., 2008b,a; Hansen e Mladenovic, 2001; Mladenovic e Hansen, 1997), Reconexão por Caminhos (Glover, 1996)) e o otimizador de programação matemática GLPK.

O pseudocódigo do *GGVNSMIP-PR* está esquematizado no Algoritmo 3. Nele, γ indica o valor do parâmetro que define o tamanho da lista restrita de candidatos da fase de construção GRASP, *GRASPmax* representa a quantidade de iterações em que a fase de construção GRASP é aplicada e *IterMax* indica o número máximo de iterações realizadas em um dado nível de perturbação.

Algoritmo 3: *GGVNSMIP-PR*

Entrada: Solução s, γ, *GRASPmax*, *IterMax*, Função $f(.)$
Saída: Solução s^* de qualidade possivelmente superior à s de acordo com a função f

```
1  s_w ← ConstróiSoluçãoEstéril()
2  s_0 ← Melhor Solução em GRASPmax iterações do procedimento ConstróiSoluçãoMinério(s_w, γ)
3  s* ← VND(s_0, f)
4  p ← 0
5  enquanto critério de parada não satisfeito faça
6      iter ← 0
7      enquanto iter < IterMax e critério de parada não satisfeito faça
8          s' ← s*
9          se p >= 10 então
10             s'' ← GLPK(s', τ)
11         fim
12         senão
13             s'' ← Refinamento(s', p, f)
14         fim
15         se s'' for melhor que s* de acordo com a função f então
16             s* ← s''; p ← 0; iter ← 0
17         fim
18         senão
19             iter ← iter + 1
20         fim
21     fim
22     p ← p + 1
23 fim
24 s* = Min(PR(S_GGVNSMIP → S_GRASP), PR(S_GRASP → S_GGVNSMIP))
25 retorna s
```

A solução inicial (linha 2 do Algoritmo 3) é gerada aplicando-se a fase de construção GRASP, conforme descrito na Seção 2.2.2. A busca local é feita pelo procedimento VND (Algoritmo 5) usando-se um grupo restrito dos movimentos descritos na seção 2.2.3, no caso, apenas nas vizinhanças: N^{CG}, N^{NV}, N^{VC} e N^{VF}. Já para a perturbação, são aplicados movimentos aleatórios de acordo com o procedimento *SelecionaVizinhança* (linha 2 do Algoritmo 4). Entre as vizinhanças desenvolvidas uma delas é escolhida de acordo com os seguintes percentuais: 30% para N^{NV}, 20% para N^{VT}, 20% para N^{CG}, 10% para N^{CT}, 10% para N^{OF} e 10% para N^{OC}. Essa probabilidade diferenciada deveu-se à influência mais significativa de determinados movimentos em relação a outros.

Algoritmo 4: *Refinamento*

Entrada: r vizinhanças: N^{NV}, N^{VT}, N^{CG}, N^{CT}, N^{OF} e N^{OC}
Entrada: Solução Inicial s, Nível p e Funcão de Avaliação f
Saída: Solução s

1 **para** $i \leftarrow 1$ **até** $p + 2$ **faça**
2 | $k \leftarrow$ SelecionaVizinhança(r)
3 | $s' \leftarrow$ Perturbação(s, k)
4 **fim**
5 $s \leftarrow$ VND(s', f)
6 **retorna** s

Algoritmo 5: VND

Entrada: r vizinhanças na ordem aleatória: N^{VF}, N^{VC}, N^{NV} e N^{CG}
Entrada: Solução Inicial s e Funcão de Avaliação f
Saída: Solução s

1 $k \leftarrow 1$
2 **enquanto** $k \leq r$ **faça**
3 | Encontre o melhor vizinho $s' \in N^{(k)}(s)$
4 | **se** $f(s') < f(s)$ **então**
5 | | $s \leftarrow s'$; $k \leftarrow 1$
6 | **fim**
7 | **senão**
8 | | $k \leftarrow k + 1$
9 | **fim**
10 **fim**
11 **retorna** s

No Algoritmo 3, a partir de um determinado nível de perturbação (linha 11 do Algoritmo 3), a busca local não é feita de forma heurística e, sim, de forma exata. Para tanto, periodicamente, é acionado o *solver* GLPK para resolver uma parte pequena do problema, obtida a partir da fixação de um grande conjunto de variáveis. Com a redução da dimensão do problema, é quase sempre possível resolver essa parte pequena na otimalidade. Assim, essa busca local pode retornar uma solução ótima desse subespaço de soluções.

A parte "pequena" do POLAD a ser submetida ao otimizador GLPK consiste em selecionar aleatoriamente 10% das frentes e fixar no modelo exato (formulação de programação matemática de Souza et al. (2010), todas as configurações das outras frentes não selecionadas, ou seja, fixar apenas os valores contidos nas células das matrizes Y, de carregadeiras e N, de viagens, que não pertencem a esses 10% selecionados. O procedimento de solução via o otimizador é interrompido em duas situações: (*i*) após encontrar a solução ótima ou (*ii*) após decorridos τ segundos de processamento.

Na fase de pós-otimização, aplicamos o procedimento de Reconexão por Caminhos Bidirecional (linha 24 do Algoritmo 3), envolvendo a solução inicial do problema (apresentada na Seção 2.2.2), e a solução final obtida pelo módulo de programação matemática.

O atributo considerado na nossa abordagem para a Reconexão por Caminhos foi a localização da carregadeira em uma solução. A cada passo o procedimento escolhe, aleatoriamente, uma frente de lavra na solução guia e move a carregadeira k alocada a essa frente, bem como as viagens dos caminhões a essa frente, para a mesma frente da solução base. Dessa forma, são mantidos os critérios de compatibilidade entre as carregadeiras e os caminhões. A seguir, uma busca local baseada no VND é feita, mas agora com os atributos

fixados durante o processo de busca (isto é, não é permitido realocar a carregadeira k, assim como alterar o número de viagens que cada caminhão faz a essa frente de lavra). Após a busca é atualizada a melhor solução. Esse procedimento se repete até que todos os atributos sejam fixados, ou seja, a solução base tenha se tornado a solução guia. O procedimento retorna a melhor solução encontrada no caminho entre as soluções base e guia.

3 Resultados Computacionais

O algoritmo proposto *GGVNSMIP-PR* foi implementado em C++ usando o framework de otimização OptFrame, disponível em *https://sourceforge.net/projects/optframe*, e compilado pelo g++ 4.0. O modelo exato foi resolvido pelo CPLEX 11.01 e pelo GLPK 4.9 na versão híbrida do algoritmo. Os testes foram executados em um microcomputador Pentium Core 2 Quad(Q6600), 2.4 GHZ e 8 GB de RAM. Para testá-lo, foi usado um conjunto de 8 problemas-teste da literatura, disponível em http://www.iceb.ufop.br/decom/prof/marcone/projects/mining.html.

Todos os experimentos consideraram os seguintes parâmetros: $\mathit{GRASPmax} = 5000$, $\mathit{IterMax} = 50$ e $\gamma = 0.3$. A função objetivo utilizada foi a mesma de Souza et al. (2010).

A Tabela 1 compara os resultados encontrados pelo algoritmo *GGVNSMIP-PR* e o Otimizador Matemático *CPLEX 11.0* aplicado à formulação de programação matemática de Souza et al. (2010). A comparação é feita considerando-se 2 e 15 minutos de processamento. A coluna "Instância" indica o problema-teste utilizado, a coluna "Melhor Conhecido" representa o melhor valor conhecido na literatura para o respectivo problema-teste, considerando a função de avaliação dada pela Equação (2), à página 6. Na coluna "Otm." indicamos por "$\surd$" instâncias nas quais o CPLEX obteve o valor ótimo da função, a coluna "Dev. Padrão" indica o desvio-padrão da amostra, "Desvio" mostra o desvio dos valores médios do algoritmo *GGVNSMIP-PR* (ou o valor encontrado pelo CPLEX) em relação ao melhor valor conhecido na literatura para cada problema-teste. O melhor valor, bem como os valores médios encontrados pelo algoritmo se referem a 30 execuções do mesmo.

Na Tabela 2 são mostrados os resultados da comparação entre os algoritmos *GGVNS*, de Souza et al. (2010), e o *GGVNSMIP-PR*, proposto neste trabalho. Nessa tabela, a coluna "IMPbest" indica o percentual de melhora proporcionado pelo algoritmo *GGVNSMIP-PR* em relação ao valor da melhor solução, enquanto "IMPdesv" indica a melhora em relação ao valor médio.

Como podemos observar na Tabela 1, o algoritmo *GGVNSMIP-PR* foi capaz de gerar soluções de boa qualidade e baixa variabilidade, com desvios de no máximo 0,71% em 2 minutos de processamento, o qual se reduz para 0,49% em 15 minutos de processamento. Quando comparado ao otimizador CPLEX, percebe-se que o algoritmo foi capaz de encontrar soluções competitivas em todas as instâncias, gerando soluções próximas ou melhores que as melhores conhecidas e com baixos desvios. No caso dos problemas-teste PAD02 e PAD05, o algoritmo proposto conseguiu gerar soluções muito melhores que o CPLEX, tanto em 2 ou 15 minutos de processamento. O algoritmo proposto também foi capaz de gerar, em apenas 2 minutos de processamento, uma solução melhor que a conhecida pela literatura para o problema-teste PAD05. Esse fato revela que, para este tempo computacional, a fase de refinamento do algoritmo termina em diferentes bacias de atração, ou seja, muitas vezes a solução que segue para o módulo de Reconexão por Caminhos está

Tabela 1. Comparação de resultados: CPLEX × *GGVNSMIP-PR*

Instância	Tempo (min)	Melhor Conhecido		CPLEX 11		GGVNSMIP-PR			
		Valor	Otm.†	Valor	Desvio (%)	Melhor	Média	Dev. Padrão	Desvio (%)
PADC01	2	227.12		230.65	1.55	228.12	228.72	0.48	0,71
PADC02	2	256.37		4858.39	> 100.00	256.37	256.47	0.67	0,04
PADC03	2	164,027.15	√	164,042.60	0.01	164,033.22	164,050.95	13.12	0.01
PADC04	2	164,056.68	√	164,061.80	0.00	164,066.24	164,099.30	19.87	0,03
PADC05	2	227.04		7,229.07	> 100.00	226.11	227.40	0.55	0,16
PADC06	2	236.58		236.58	0.00	236.58	237.50	2.21	0,39
PADC07	2	164,017.46	√	164,017.46	0.00	164,019.22	164,020.67	0.66	0.00
PADC08	2	164,018.65	√	164,018.65	0.00	164,020.84	164,099.30	0.78	0.00
PADC01	15	227.12		227.12	0.00	228.12	228.23	0.33	0,49
PADC02	15	256.37		262.06	2.22	256.37	256.45	0.59	0,03
PADC03	15	164,027.15	√	164032.29	0.00	164,033.02	164,038.54	5.35	0,01
PADC04	15	164,056.68	√	164056.68	0.00	164,074.69	164,083.39	7.34	0.02
PADC05	15	227.04		7,229.07	> 100.00	227.04	227.04	0.00	0.00
PADC06	15	236.58		236.58	0.00	236.58	236.58	0.00	0.00
PADC07	15	164,017.46	√	164,017.46	0.00	164,018.27	164,019.34	0.68	0.00
PADC08	15	164,018.65	√	164,018.65	0.00	164,020.22	164,021.18	0.52	0.00

† Considerando a tolerância do CPLEX mipgap $\leq 1 \times 10^{-5}$,
exceto no PADC03, no qual, considara-se a tolerância do CPLEX mipgap $\leq 1 \times 10^{-4}$

Tabela 2. Comparação de resultados: *GGVNS* × *GGVNSMIP-PR*

Instância	Tempo (min)	GGVNS		GGVNSMIP-PR		IMPbest (%)	IMPdesv (%)
		Média	Melhor	Média	Melhor		
PADC01	2	230,12	230,12	228,72	228,12	0,87	0,61
PADC02	2	256,56	256,37	256,46	256,37	0,00	0,04
PADC03	2	164064,68	164039,12	164050,95	164033,22	0,00	0,01
PADC04	2	164153,92	164099,66	164099,30	164066,24	0,02	0,03
PADC05	2	228,09	228,09	227,40	226,11	0,87	0,30
PADC06	2	237,97	236,58	237,50	236,58	0,00	0,20
PADC07	2	164021,89	164021,38	164020,67	164019,22	0,00	0,00
PADC08	2	164027,29	164023,73	164022,38	164020,84	0,00	0,00

em um mínimo local não conhecido antes. Logo, a pós-otimização consegue encontrar soluções com melhor potencial de otimalidade quando caminha pelo espaço de busca.

Pela Tabela 2 percebemos claramente a influência dos módulos de Reconexão por Caminhos e do otimizador de programação matemática sobre o algoritmo *GGVNS*. Como se observa, foi possível melhorar a qualidade das soluções finais em até 0,87%, e reduzir a variabilidade dessas soluções em até 0,61%.

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 híbrido, denominado *GGVNSMIP-PR*, que combina a flexibilidade das metaheurísticas com o poderio da programação matemática. O algoritmo ainda utiliza um módulo de pós-otimização baseado na estratégia de Reconexão por Caminhos.

Usando problemas-teste da literatura, o algoritmo foi, inicialmente, comparado com o otimizador CPLEX 11.0 aplicado exclusivamente a um modelo de programação matemática. Verificou-se que o algoritmo proposto é competitivo com esse otimizador na maioria dos casos, pois foi capaz de encontrar soluções de boa qualidade rapidamente e com baixa variabilidade das soluções finais. Em dois dos problemas-teste, o algoritmo proposto ainda produziu soluções muito melhores que o CPLEX.

Posteriormente, o algoritmo proposto foi comparado com o algoritmo *GGVNS*. Os resultados mostraram que a introdução dos dois módulos propostos contribuiram para melhorar a qualidade das soluções finais, bem como para a redução da variabilidade dessas.

Para trabalhos futuros é proposta a concessão de novos tipos de perturbações. Sugere-se, também, substituir o otimizador GLPK por outro mais robusto, como, por exemplo, o CPLEX. Além disso, propõe-se melhorar o mecanismo de pós-otimização, bem como incorporá-lo ao algoritmo como uma estratégia de intensificação a cada ótimo local encontrado, visto que a estratégia atual já se mostrou eficiente. Propõe-se, também, a implementação de uma versão paralela do algoritmo *GGVNSMIP-PR* visando tirar proveito da tecnologia *multi-core*, já presente nas máquinas atuais.

Agradecimentos

Os autores agradecem à FAPEMIG (processos CEX-PPM 357/09 e CEX-APQ 01201-09) pelo apoio recebido.

Referências

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

Coelho, I. M.; Ribas, S.; Souza, M. J. F. e Coelho, V. N. (2009)a. Um algoritmo heurístico híbrido para o planejamento operacional de lavra. *Anais do IX Congresso Brasileiro de Redes Neurais (CBRN)*, p. 1–7, Ouro Preto, MG.

Coelho, I. M.; Ribas, S.; Souza, M. J. F.; Coelho, V. N. e Ochi, L. S. (2009)b. A hybrid heuristic algorithm based on grasp, vnd, ils and path relinking for the open-pit-mining operational planning problem. *Anais do XXX Congresso Ibero-Latino-Americano de Métodos Computacionais em Engenharia (CILAMCE)*, p. 1–14, Búzios, RJ.

Costa, F. P. (2005). Aplicações de técnicas de otimização a problemas de planejamento operacional de lavras em mina a céu aberto. Dissertação, Programa de Pó-Graduação em Engenharia Mineral, Escola de Minas, UFOP, Ouro Preto.

Feo, T. A. e Resende, M. G. C. (1995). Greedy randomized adaptive search procedures. *Journal of Global Optimization*, v. 6, p. 109–133.

Glover, F. (1996). Tabu Search and adaptive memory programming - advances, applications and challenges. Barr, R.; Helgason, R. e Kennington, J., editors, *Interfaces in Computer Sciences and Operations Research*, p. 1–75. Kluwer Academic Publishers.

Guimarães, I. F.; Pantuza, G. e Souza, M. J. F. (2007). 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. *Anais do XIV Simpósio de Engenharia de Produção (SIMPEP)*, p. 11, Bauru, CD-ROM.

Hansen, P. e Mladenovic, N. (2001). Variable neighborhood search: Principles and applications. *European Journal of Operational Research*, v. 130, p. 449–467.

Hansen, P.; Mladenovic, N. e Pérez, J. A. M. (2008)a. Variable neighborhood search. *European Journal of Operational Research*, v. 191, p. 593–595.

Hansen, P.; Mladenovic, N. e Pérez, J. A. M. (2008)b. Variable neighborhood search: methods and applications. *4OR: Quarterly journal of the Belgian, French and Italian operations research societies*, v. 6, p. 319–360.

Lourenço, H. R.; Martin, O. C. e Stützle, T. (2003). Iterated local search. Glover, F. e Kochenberger, G., editors, *Handbook of Metaheuristics*. Kluwer Academic Publishers, Boston.

Mladenovic, N. e Hansen, P. (1997). A variable neighborhood search. *Computers and Operations Research*, v. 24, p. 1097–1100.

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

Souza, M. J. F.; Coelho, I. M.; Ribas, S.; Santos, H. G. e Merschmann, L. H. C. (2010). A hybrid heuristic algorithm for the open-pit-mining operational planning problem. *European Journal of Operational Research, EJOR*, v. , p. 1–21. Aceito para publicação.