PARALLEL ITERATED LOCAL SEARCH APLICADO AO PLANEJAMENTO OPERACIONAL DE LAVRA

Sabir Ribas, Igor Machado Coelho
Marcone Jamilson Freitas Souza, David Menotti

Universidade Federal de Ouro Preto
Departamento de Ciência da Computação
Instituto de Ciências Exatas e Biológicas
35400-000, Ouro Preto, MG
{sabir, igor, marcone, menotti}@iceb.ufop.br

RESUMO

Este trabalho apresenta um algoritmo paralelo baseado na metaheurística *Iterated Local Search.* Para testá-lo, ele é aplicado a um problema que necessita de agilidade na geração da resposta, no caso, o planejamento operacional de lavra em minas a céu aberto. Dada a complexidade combinatória deste problema, foi desenvolvido um algoritmo heurístico híbrido, que combina a paralelização do *Iterated Local Search* com os procedimentos GRASP e *Variable Neighborhood Descent*. O algoritmo paralelo foi implementado em uma arquitetura *multi-core.* Para validá-lo, os resultados obtidos foram comparados com aqueles produzidos pelo otimizador CPLEX e uma versão sequencial do mesmo, todos limitados ao mesmo tempo de processamento. Observou-se melhoria na qualidade das suas soluções finais quando comparadas com as obtidas pela sua versão sequencial. Houve também melhora em relação às soluções encontradas pelo CPLEX.

PALAVRAS CHAVE. Algoritmos Paralelos. *Multi-core*. Metaheurísticas. *Iterated Local Search.* Planejamento operacional de lavra.

ABSTRACT

This work presents a parallel algorithm based on Iterated Local Search metaheuristic. In order to test this algorithm, it is applied to a problem which requires fast answers, *i.e*., the open-pit mining problem. Given its combinatorial complexity, we develop a hybrid heuristic algorithm, which combines an Iterated Local Search parallel version with GRASP and Variable Neighborhood Descent procedures. The parallel algorithm was implemented on a multi-core architecture. In order to validate the proposed parallel algorithm, its results were compared with the results of both CPLEX and its sequential version, all in the same processing time. Results show improvement in the quality of its final solutions in relation to those obtained by its sequential version. In addition, the proposed algorithm overcomes the solutions found by CPLEX.

KEYWORDS. Parallel Algorithms. Multi-core. Metaheuristics. Iterated Local Search. Open-pit mining.

1. Introdução

O crescimento da demanda por sistemas computacionais de auxílio à tomada de decisão acompanha o aumento da complexidade das cadeias produtivas. Tais sistemas normalmente estão relacionados a proporcionar uma visualização objetiva e clara da situação de uma linha de produção, à resolução de um gargalo nos processos de um sistema produtivo ou à melhoria de qualidade de algum produto.

Muitos problemas encontrados em linhas produtivas são de natureza combinatória e, como tal, o desafio é encontrar a melhor entre as soluções existentes. Para resolvê-los destacam-se duas abordagens: a heurística e a exata. A vantagem de se adotar metodologias exatas é a garantia da obtenção das soluções ótimas para o problema, o que não acontece no caso de heurísticas. Porém o uso de heurísticas é muito atraente quando a prioridade é o tempo de resposta. Heurísticas são métodos de busca geralmente dotados de alguma inteligência e apresentam boas soluções em um tempo relativamente baixo se comparados com os métodos exatos. Assim, métodos heurísticos passaram a ser vastamente empregados na resolução de problemas reais.

Para melhorar o desempenho das heurísticas vêm surgindo na literatura propostas de paralelização desses procedimentos, como os de Resende e Ribeiro (2005), Muhammad (2006) e Campos *et al*. (2006). Isso se deve ao crescimento das facilidades em se montar um *cluster* e também pelo crescente investimento em multiprocessadores por parte dos fabricantes. O tempo de resolução de problemas de otimização poderia ser reduzido se os algoritmos utilizassem os recursos *multi-core* desses processadores assim como versões de sistemas operacionais especializados em ambientes distribuídos e construção rápida de *clusters*, como é o caso das distribuições Linux PelicanHPC e Parallel Knoppix.

De forma a explorar a capacidade dos processadores e das arquiteturas *multi-core* atuais, desenvolve-se neste trabalho uma proposta de paralelização da metaheurística *Iterated Local Search*. Para testá-la, esta é aplicada a um problema que necessita de agilidade na geração da resposta, no caso, o planejamento operacional de lavra em minas a céu aberto com alocação dinâmica de caminhões. A tomada de decisão nesse problema tem que ser rápida já que os custos de produção envolvidos são bastante elevados (Rodrigues, 2006).

Para resolvê-lo, as abordagens variam entre métodos exatos e heurísticos. Entre essas, apontamos os trabalhos de White e Olson (1986), Chanda e Dagdelen (1995), Alvarenga (1997), Merschmann (2002), Costa *et al*. (2004), Rodrigues (2006), Guimarães *et al*. (2007) e Coelho *et al*. (2008). Neste último trabalho é proposto um algoritmo heurístico híbrido que representa uma evolução em relação ao de Costa (2005), por incluir mais restrições e outros movimentos para explorar o espaço de soluções. O algoritmo proposto combina os procedimentos heurísticos GRASP (Feo e Resende, 1995), *Variable Neighborhood Descent* (Mladenovic e Hansen, 1997) e *Iterated Local Search* (Lourenço *et al*., 2003) para resolver o problema em pauta. Usando quatro problemas-teste da literatura, o algoritmo heurístico foi comparado com o otimizador CPLEX 9.1 aplicado a um modelo de programação matemática. Foram realizados testes envolvendo dois minutos de processamento, considerado adequado para a tomada de decisão. 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 0,12% piores, na média.

O restante deste trabalho está organizado como segue. A Seção 2 apresenta as abordagens para a resolução do problema de planejamento de lavra. Na Seção 3 são apresentados e discutidos os resultados obtidos. Finalmente, na Seção 4, são apresentadas as conclusões e propostas para trabalhos futuros.

2. Abordagens

Nesta seção são apresentadas duas abordagens encontradas na literatura para o planejamento operacional de lavra, sendo uma exata e uma heurística sequencial, bem como uma proposta de algoritmo heurístico paralelo.

2.1 Abordagem de Programação Matemática

O modelo matemático de alocação dinâmica de caminhões é o mesmo descrito em Coelho *et*

al. (2008), o qual, por sua vez, representa um aperfeiçoamento daquele proposto em Costa *et al.* (2004). Para sua apresentação, sejam os seguintes parâmetros:

M Conjunto de frentes de minério;
E Conjunto de frentes de estéril;
F Conjunto de frentes formado por $M \cup E$;
T Conjunto de parâmetros de controle analisados no minério;
C Conjunto de equipamentos de carga;
V Conjunto de equipamentos de transporte;
M_r Ritmo de lavra recomendado relativo a minério (t/h);
M_l Ritmo de lavra mínimo relativo a minério (t/h);
M_u Ritmo de lavra máximo relativo a minério (t/h);
E_r Ritmo de lavra recomendado relativo a estéril (t/h);
E_l Ritmo de lavra mínimo relativo a estéril (t/h);
E_u Ritmo de lavra máximo relativo a estéril (t/h);
α^- Penalidade por desvio negativo da produção de minério;
α^+ Penalidade por desvio positivo da produção de minério;
β^- Penalidade por desvio negativo da produção de estéril;
β^+ Penalidade por desvio positivo da produção de estéril;
t_{ij} Valor do parâmetro j na frente i (%);
tr_j Valor recomendado para o parâmetro de controle j na mistura (%);
tl_j Valor mínimo admissível para o parâmetro de controle j na mistura (%);
tu_j Valor máximo admissível para o parâmetro de controle j na mistura (%);
λ_j^- Penalidade por desvio negativo para o parâmetro de controle j na mistura;
λ_j^+ Penalidade por desvio positivo para o parâmetro de controle j na mistura;
ω_l Penalidade por uso do l-ésimo caminhão;
Qu_i Ritmo de lavra máximo para a frente i (t/h);
Tx_l Taxa máxima de utilização do caminhão l;
Cl_k Produção mínima do equipamento de carga k (t/h);
Cu_k Produção máxima do equipamento de carga k (t/h);
cap_l Capacidade do caminhão l (t);
T_{il} Tempo total de ciclo do caminhão l na frente i (min);
g_{lk} 1, se o caminhão l é compatível com o equipamento de carga k; e 0, caso contrário.

e as seguintes variáveis de decisão:

x_i Ritmo de lavra da frente i (t/h);
y_{ik} 1, se o equipamento de carga k opera na frente i; e 0, caso contrário.
n_{il} Número de viagens que um caminhão l realiza à frente i;
P_m^- Desvio negativo de produção de minério em relação ao recomendado (t/h);
P_m^+ Desvio positivo de produção de minério em relação ao recomendado (t/h);
P_e^- Desvio negativo de produção de estéril em relação ao recomendado (t/h);
P_e^+ Desvio positivo de produção de estéril em relação ao recomendado (t/h);
d_j^- Desvio negativo do parâmetro de controle j na mistura (t/h);
d_j^+ Desvio positivo do parâmetro de controle j na mistura (t/h);
U_l 1, se o veículo l é usado; e 0, caso contrário.

A seguir, pelas Equações (1)-(26), é apresentada uma formulação de programação linear inteira mista relativa à alocação dinâmica de uma frota heterogênea de caminhões e equipamentos de carga, tendo-se como objetivo alcançar as metas de produção e qualidade de minério e reduzir o número de caminhões necessários.

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

$$\sum_{i \in M} (t_{ij} - tu_j) x_i \leq 0 \quad \forall j \in T \quad (2)$$

$$\sum_{i \in M} (t_{ij} - tl_j) x_i \geq 0 \quad \forall j \in T \quad (3)$$

$$\sum_{i \in M} (t_{ij} - tr_j) x_i + d_j^- - d_j^+ = 0 \quad \forall j \in T \quad (4)$$

$$\sum_{i \in M} x_i \leq M_u \quad (5)$$

$$\sum_{i \in M} x_i \geq M_l \quad (6)$$

$$\sum_{i \in M} x_i + P_m^- - P_m^+ = M_r \quad (7)$$

$$\sum_{i \in E} x_i \leq E_u \quad (8)$$

$$\sum_{i \in E} x_i \geq E_l \quad (9)$$

$$\sum_{i \in E} x_i + P_e^- - P_e^+ = E_r \quad (10)$$

$$x_i \leq Qu_i \quad \forall i \in F \quad (11)$$

$$x_i \geq 0 \quad \forall i \in F \quad (12)$$

$$d_j^+, \; d_j^- \geq 0 \quad \forall j \in T \quad (13)$$

$$P_m^+, \; P_m^- \geq 0 \quad (14)$$

$$P_e^+, \; P_e^- \geq 0 \quad (15)$$

$$\sum_{k \in C} y_{ik} \leq 1 \quad \forall i \in F \quad (16)$$

$$\sum_{i \in F} y_{ik} \leq 1 \quad \forall k \in C \quad (17)$$

$$y_{ik} \in \{0,1\} \quad \forall i \in F, \; \forall k \in C \quad (18)$$

$$x_i - \sum_{k \in C} Cu_k y_{ik} \leq 0 \quad \forall i \in F \quad (19)$$

$$x_i - \sum_{k \in C} Cl_k y_{ik} \leq 0 \quad \forall i \in F \quad (20)$$

$$n_{il} T_{il} - 60 \sum_{k \in C, \, g_{lk} \neq 0} y_{ik} \leq 0 \quad \forall i \in F, \; \forall l \in V \quad (21)$$

$$x_i - \sum_{l \in V} n_{il} cap_l = 0 \quad \forall i \in F \quad (22)$$

$$\frac{1}{60} \sum_{i \in F} n_{il} T_{il} \leq T x_l \quad \forall l \in V \quad (23)$$

$$U_l - \frac{1}{60} \sum_{i \in F} n_{il} T_{il} \geq 0 \quad \forall l \in V \quad (24)$$

$$n_{il} \in \mathbf{Z}^+ \quad \forall i \in F, \; \forall l \in V \quad (25)$$

$$U_l \in \{0,1\} \quad \forall l \in V \quad (26)$$

Podemos distinguir nesta formulação três grupos de equações. O primeiro diz respeito ao problema de mistura de minérios com metas, sendo formado pelas Equações (2)-(15). O segundo (Equações (16) a (20)) modela a alocação das carregadeiras e a faixa de produtividade que torna viável a utilização desses equipamentos; enquanto o último grupo (Equações (21) a (26)) está relacionado ao transporte de material na mina e a alocação e utilização dos caminhões.

No primeiro grupo, as Equações (7) e (10) modelam o atendimento às metas de produção de minério e estéril, respectivamente; enquanto as Equações (4) visam atender as metas de qualidade estabelecidas para os diversos parâmetros de controle. Para cada uma dessas equações envolvendo metas, há variáveis de desvio que são penalizadas na função objetivo (1). As Equações (2) e (3) impedem que soluções infactíveis com respeito aos limites de especificação dos parâmetros de controle sejam aceitas. As Equações (5) e (6) asseguram que a produção de minério está dentro dos limites $[M_l, M_u]$, enquanto que as Equações (8) e (9) garantem que a produção de estéril esteja na faixa $[E_l, E_r]$. As Equações (11) asseguram que o ritmo de lavra em cada frente não supera aquele estabelecido pelo usuário. As demais equações desse grupo definem que as variáveis envolvidas são não-negativas.

Em relação ao segundo grupo, as Equações (16) definem que em cada frente pode ser alocado, no máximo, um único equipamento de carga, enquanto que as Equações (17) asseguram que cada equipamento de carga pode operar, no máximo, em uma única frente. As Equações (18)

caracterizam as variáveis y_{ik} como binárias. As Equações (19) e (20) limitam, respectivamente, o ritmo de lavra máximo e mínimo, definidos pela carregadeira alocada à frente. Já as Equações (11) limitam o ritmo de lavra máximo definido pelo usuário.

No terceiro grupo, cada Equação (21) faz com que um caminhão somente realize viagens a uma frente onde esteja alocado um equipamento de carga compatível. As Equações (22) fazem com que o ritmo de lavra de uma frente seja igual à produção realizada pelos caminhões alocados à frente. As Equações (23) definem que cada caminhão opere no máximo *Tx*% em uma hora. As Equações (24), juntamente com a função objetivo, forçam com que os caminhões usados sejam penalizados. As Equações (25) forçam que seja inteiro positivo o número de viagens que um caminhão faz a uma frente. As Equações (26) indicam que as variáveis U_l são binárias.

2.2 Abordagem Heurística

2.2.1 Representação de uma Solução

Uma solução é representada por uma matriz $R = [Y \mid N]$, onde Y é a matriz $|F| \times 1$ e N é 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 (Equação (18) do modelo matemático).

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.

A partir de Y, N e os tempos de ciclo da matriz $Tc_{|F|\times|V|}$ são determinados $X_{|F|\times 1}$ e $Tcv_{1\times|V|}$, os quais representam, respectivamente, a quantidade de massa lavrada em cada frente e o somatório dos tempos de ciclo de cada caminhão.

2.2.2 Avaliação de uma Solução

Uma solução s é avaliada pela função f da Equação (27). Esta função, que deve ser minimizada, tem como primeira componente a função objetivo propriamente dita, que corresponde à Equação (1) do modelo de programação matemática. As demais componentes penalizam a ocorrência de inviabilidades da solução s em vista do fato de que os movimentos utilizados para explorar o espaço de soluções (vide Seção 2.2.4) podem gerar soluções infactíveis.

$$f(s) = f^{PM}(s) + f^{p}(s) + \sum_{j\in T} f_j^{q}(s) + \sum_{l\in V} f_l^{u}(s) + \sum_{k\in C} f_k^{c}(s) \qquad (27)$$

Na Equação (27), $f^{PM}(s)$ é uma função que avalia s quanto ao atendimento às metas de produção e qualidade, bem como número de caminhões utilizados; $f^{p}(s)$ 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 do atendimento da taxa de utilização máxima do l-ésimo caminhão e $f_k^c(s)$, que avalia s quanto ao desrespeito aos limites de produtividade da carregadeira k.

2.2.3 Construção de uma Solução inicial

Uma solução inicial para o problema é obtida por um procedimento construtivo guloso. A construção é feita em duas etapas. As alocações das carregadeiras e a distribuição das viagens às frentes são feitas, na primeira etapa, às frentes de 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.

Em seguida, pela Figura 1, são apresentados os procedimentos de construção mencionados. Na classificação dos elementos candidatos à inserção na solução, considera-se que para as frentes de estéril, a melhor frente é a que possui a maior massa, a melhor carregadeira é a que oferece a maior produção e o melhor caminhão é o de maior capacidade. Já para as frentes de minério considera-se que a melhor frente é a que possui o menor desvio dos teores em relação às metas, a melhor carregadeira é a de maior produção e o melhor caminhão é o de menor capacidade.

```
Procedimento Constrói_Solução_Estéril
1  Seja s uma solução vazia
2  enquanto (a produção de estéril for menor que a produção recomendada) e
                (existirem frentes de estéril não utilizadas) faça
3    frente_atual ← melhor frente de estéril ainda não utilizada
4    se não há carregadeira na frente frente_atual então
5      Aloque a melhor carregadeira ainda não alocada
6    se todas as carregadeiras já foram alocadas então Retorne s
7    para cada caminhão l faça
8      se o caminhão l for compatível com a carregadeira da frente frente_atual então
9      enquanto (a produção de estéril for menor que a produção recomendada) e
                  (o caminhão l pode fazer mais uma viagem) faça
10       Aloque ao caminhão l uma viagem para a frente frente_atual
11 Retorne s;
```

```
Procedimento Constrói_Solução_Minério
1  enquanto (a produção de minério for menor que a produção recomendada) e
                (existirem frentes de minério não utilizadas) faça
     Ordene as frentes pelos desvios de meta (ordem crescente)
2    frente_atual ← Escolha a melhor frente de minério que pode ser utilizada
3    se não há carregadeira na frente f_atual então
4      Aloque a melhor carregadeira ainda não alocada
5    se todas as carregadeiras já foram alocadas então Retorne s
6    para cada caminhão l faça
7      se o caminhão for compatível com a carregadeira da frente frente_atual então
8      enquanto (a produção de minério for menor que a produção recomendada) e
                  (o caminhão l pode fazer mais uma viagem) faça
9        Aloque ao caminhão l uma viagem para a frente frente_atual
10 Retorne s;
```

Figura 1: Procedimentos para construção de uma solução inicial

2.2.4 Estruturas de vizinhança

Para explorar o espaço de soluções do problema foram utilizados 8 movimentos, apresentados a seguir, sendo os seis primeiros propostos em Costa (2005) e os dois últimos propostos em Coelho *et al*. (2008).

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. No caso de apenas uma das frentes possuir equipamento de carga e a outra estiver disponível, 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 juntamente 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}. Desta forma, 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 neste movimento, 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}. Portanto, 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 neste movimento, havendo realocação de viagens apenas quando houver compatibilidade entre eles.

Movimento Operação Frente - $N^{OF}(s)$: Consiste em retirar de operação a carregadeira que esteja em operação na frente *i*. Todas as viagens de caminhões feitas a essa frente são, também, eliminadas. A carregadeira retorna à operação assim que uma nova viagem é associada a ele.

Movimento Operação Caminhão - $N^{OC}(s)$: Consiste em selecionar uma célula $n_{il} \in N$ 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 viagem é realocada de uma para outra. Tal movimento pode ocorrer entre quaisquer células da matriz *N*, respeitando-se as restrições de compatibilidade entre equipamentos.

Movimento Troca de Carregadeiras - $N^{CT}(s)$: Consiste em trocar duas células distintas y_i e y_k da matriz *Y*, ou seja, em trocar as carregadeiras que operam nas frentes *i* e *k*. Analogamente ao movimento *CG*, em um movimento *CT* os equipamentos de carga das frentes 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.

2.2.5 Algoritmo heurístico sequencial GVILS

O algoritmo heurístico sequencial GVILS, proposto em Coelho *et al*. (2008), combina os procedimentos *Greed Randomized Adaptative Search Procedure* – GRASP (Feo e Resende, 1995), *Variable Neighborhood Descent* – VND (Mladenovic e Hansen, 1997) e *Iterated Local Search* – ILS (Lourenço *et al*., 2003). Seu pseudocódigo é esquematizado na Figura 2.

```
Algoritmo GVILS (s)
1    s ← ConstróiSoluçãoEstéril()
2    s ← ConstróiSoluçãoMinério()
3    s ← VND(v')
4    enquanto (Critério de parada não satisfeito) faça
5       s' ← Perturbação(s, nível)
6       s'' ← VND(s')
7       se f(s'') < f(s) então
8          s ← s''
9       fim-se
10   fim-enquanto
11   s ← s*
12   retorne s
```

Figura 2: Algoritmo heurístico sequencial GVILS

A construção de uma solução inicial (linhas 1 e 2 do algoritmo GVILS) é feita pelos procedimentos descritos na Seção 2.2.4. A busca local é feita pelo procedimento VND usando-se os movimentos descritos na seção anterior e opera nas vizinhanças em uma ordem pré-definida, começando das que exigem menor esforço computacional para aquelas que exigem maior. Assim, o VND segue a seguinte ordem de exploração: N^{CG}, N^{NV}, N^{VC}, N^{VF}. Os movimentos relativos às vizinhanças N^{CT}, N^{OC}, N^{OF} e N^{VT} não foram utilizados na busca local, mas apenas como perturbação. A justificativa para esta estratégia é o fato de que estes últimos quatro movimentos requerem um elevado tempo computacional durante a busca local, sem um benefício proporcional ao esforço despendido.

Uma das principais características de algoritmos baseados em ILS é o conceito de perturbação. Seu objetivo é diversificar a busca, gerando uma solução diferente e cada vez mais "distante" da região atual de exploração no espaço de busca. Para cumprir esta missão, comumente são estabelecidos vários níveis de perturbação. No trabalho em questão, para cada nível *n*, são aplicados à solução corrente *n+2* movimentos, escolhidos aleatoriamente dentre os 8 descritos na Seção 2.2.4. A essa solução perturbada é aplicada busca local, baseada no procedimento VND (linha 6 do algoritmo GVILS). Após *IterMax* iterações sem melhora em um dado nível, este é aumentado. Encontrando uma solução melhor, a perturbação volta ao seu nível mais baixo.

2.2.6 *Parallel Iterated Local Search* (PILS)

Neste trabalho, propomos uma paralelização da metaheurística *Iterated Local Search*, denominada *Parallel Iterated Local Search* (PILS). Seu pseudocódigo é mostrado na Figura 3.

Procedimento *Parallel Iterated Local Search*

```
1   s0 ← GeraSoluçãoInicial
2   s* ← BuscaLocal (s0)
3   repita
4      A ← |P| cópias de s*
5      B ← ∅
6      para cada s ∈ A faça em paralelo
7         s' ← Perturbação (s*, histórico)
8         s'' ← BuscaLocal (s')
9         Adicione s'' ao conjunto B
10     fim
11     s* ← CritérioAceitação (s*, B, histórico)
12  até critério de parada satisfeito
13  retorne s*
```

Figura 3: *Parallel Iterated Local Search* (PILS)

Nesta figura, s_0 é uma solução inicial; s^* é a melhor solução obtida durante sua execução; P é o conjunto de nodos de processamento; A é um conjunto que armazena $|P|$ cópias da melhor solução até então; B é um conjunto de soluções refinadas a partir do conjunto A; s' é a solução perturbada; e s'', obtido pela aplicação da busca local à solução perturbada, é, potencialmente, um ótimo local melhor que as soluções de A. Como se observa na Figura 3, inicialmente, é gerada uma solução inicial e a ela é aplicada uma busca local. Em seguida vem a parte paralela do algoritmo. Para isso, a cada iteração, é criada uma sequência A de soluções idênticas à melhor solução encontrada até o momento, sendo o tamanho da sequência definida como sendo o número de processos que serão criados. Para se aproveitar ao máximo a arquitetura utilizada, o número de processos deve ser igual ao número de nodos de processamento. O próximo passo é a perturbação e o refinamento das soluções presentes em A, os quais são realizados em um nodo de processamento. As soluções refinadas em cada iteração são armazenadas em um conjunto B. Ao fim da parte paralela do algoritmo, é aplicado um critério de aceitação para se determinar a melhor solução obtida até o momento considerando-se a própria solução s^*, o conjunto B e outras soluções eventualmente armazenadas.

2.2.7 *PILS* Aplicado ao Planejamento Operacional de Lavra

O algoritmo proposto para tratar o Planejamento Operacional de Lavra, chamado PGVILS, é a versão paralela daquele desenvolvido em Coelho *et al*. (2008) usando a estratégia PILS da Figura 3. A fase construtiva de PGVILS é feita pelo procedimento descrito na Seção 2.2.3. A busca local é feita por um procedimento baseado na metaheurística *Variable Neighborhood Descent* (VND) usando-se os movimentos descritos na Seção 2.2.4 e adotando-se a mesma ordem de exploração apresentada na Seção 2.2.5. Além disso, a definição dos níveis de perturbação segue a mesma estratégia estabelecida na Seção 2.2.5.

2.2.8 Sistema Desenvolvido

Para implementar de forma rápida o algoritmo proposto, foi utilizada a biblioteca MapMP[1] que corresponde a uma implementação da abstração *MapReduce*.

MapReduce é uma abstração simples e poderosa, geralmente aplicada ao processamento ou geração de grandes massas de dados. Dean e Ghemawat (2004) apresentam uma visão geral sobre a implementação do *MapReduce* da Google, a qual facilita muito o trabalho de seus programadores. Esse modelo foi feito para processar grandes conjuntos de dados de uma maneira massivamente paralela e é baseado nos seguintes fatores (Lämmel, 2009): (i) iteração sobre a

[1] Disponível em *http://sourceforge.net/projects/mapreducepp*

entrada; (ii) computação sobre cada um dos pares chave/valor da entrada; (iii) agrupamento de todos os valores intermediários por chaves; (iv) iteração sobre os grupos resultantes; (v) redução de cada grupo. O usuário da biblioteca *MapReduce* expressa a computação como duas funções: *map* e *reduce*. *map*, escrita pelo usuário, recebe um par como entrada e produz um conjunto de pares intermediários também na forma chave/valor. A biblioteca *MapReduce* agrupa todos os valores intermediários associados à mesma chave intermediária e os passa à função *reduce*. A função *reduce*, também escrita pelo usuário, aceita uma chave intermediária e o conjunto de valores relacionados àquela chave. Essa função junta esses valores para formar um conjunto possivelmente menor de valores. Tipicamente, zero ou apenas um valor é produzido pela função *reduce*. Os valores intermediários são passados para o usuário da função *reduce*.

Especificamente para problemas de otimização, onde os itens a serem mapeados não necessitam ser ordenados ou agrupados por chaves, pois o que interessa é a solução e uma forma de se medir sua qualidade, o modelo pode ser simplificado. Usuários especificam as funções *map* e *reduce*. A primeira processa um item de um tipo α e gera um tipo β. A segunda mescla todos os dados intermediários, representados por um conjunto de elementos do tipo β, gerado a partir da aplicação da função *map* a um conjunto de dados do tipo α. Programas escritos nesse estilo funcional podem ser automaticamente paralelizados e executados em grandes *clusters*.

MapMP é uma implementação em C++ da abstração *MapReduce* para multiprocessadores. Tal implementação compõe o projeto MapReduce++ que atualmente conta com recursos de paralelização tanto em *threads* quanto em rede. A paralelização por *threads* permite aos usuários o desenvolvimento de procedimentos que utilizam ao máximo a capacidade dos computadores atuais, os quais possuem recursos de multiprocessamento. Já a paralelização em rede permite aos usuários o desenvolvimento de aplicações distribuídas para computação de alto desempenho em *clusters* sofisticados ou em um conjunto de computadores pessoais. O principal objetivo da biblioteca é oferecer ao usuário todo aparato necessário para o desenvolvimento rápido de aplicações paralelas.

3. Resultados e Análise

A aplicação do algoritmo proposto PGVILS ao planejamento operacional de lavra foi desenvolvida em C++ usando o compilador g++ 4.0. O modelo de programação matemático foi resolvido pelo otimizador CPLEX, versão 11. Tanto o algoritmo heurístico quanto o exato foram testados em um PC Pentium Core 2 Quad (Q6600), com 2,4 GHz e 8GB de RAM. O CPLEX foi executado no Windows Vista, e o algoritmo heurístico paralelo na distribuição Linux Ubuntu 8.04LTS, por questões de compatibilidade com a biblioteca de paralelização OpenMP utilizada.

Para testar o algoritmo proposto foram usados quatro cenários de Coelho *et al.* (2008), os quais se referem a dados do planejamento operacional de empresas mineradoras do quadrilátero ferrífero, situadas na região central do Estado de Minas Gerais. Os parâmetros de controle são os teores químicos (Fe, SiO_2, Mn, P, H_2O, entre outros) e granulometrias especificadas para o minério. Tanto os pesos da função de avaliação quanto os parâmetros do algoritmo heurístico foram os mesmos de Coelho *et al*. (2008).

Os melhores resultados atuais para esses problemas-teste são os seguintes: PADC1: 227,12; PADC2: 252,41; PADC3: 164034,27 e PADC4: 164054,04.

A Tabela 1 mostra os resultados obtidos em 15, 30, 60 e 120 segundos de processamento, usando-se 1, 2, 3 e 4 nodos de processamento. Nesta tabela, “Núcleos” indica a quantidade de nós de processamento, as demais colunas indicam o melhor valor e o valor médio da função de avaliação para cada problema-teste (PADC1, PADC2, PADC3 e PADC4) encontrado em 30 execuções do algoritmo. Observa-se que os resultados obtidos pela aplicação do PGVILS em um único núcleo equivalem à aplicação da versão sequencial GVILS.

TABELA 1: Resultados do algoritmo PGVILS

Tempo (segs)	Núcleos	Melhores valores de custo				Valores médios do custo			
		PADC1	PADC2	PADC3	PADC4	PADC1	PADC2	PADC3	PADC4
15	1	227,12	257,40	164090,31	164140,49	4408,83	4431,14	164107,81	164271,73
15	2	227,12	254,80	164089,20	164132,21	2561,24	4143,19	164104,85	164272,55
15	3	227,12	253,37	164080,36	164140,49	2796,25	2852,68	164105,95	164275,18
15	4	227,12	254,80	164058,33	164140,49	3421,54	2218,68	164103,88	164269,16
30	1	227,12	252,41	164073,84	164137,04	1947,76	1681,23	164105,82	164242,80
30	2	227,12	252,41	164080,36	164155,10	947,70	724,88	164102,00	164247,43
30	3	227,12	253,37	164080,36	164146,27	694,23	1280,13	164100,86	164239,58
30	4	227,12	252,41	164072,84	164136,89	694,31	491,40	164105,40	164232,21
60	1	227,12	252,41	164073,84	164111,47	460,77	258,25	164101,43	164231,00
60	2	227,12	252,41	164073,02	164132,21	461,11	255,97	164093,72	164237,06
60	3	227,12	252,41	164073,72	164126,92	227,38	255,02	164095,43	164227,59
60	4	227,12	252,41	164080,36	164107,97	227,42	256,88	164096,15	164203,98
120	1	227,12	252,41	164048,15	164110,29	460,64	254,29	164097,44	164204,40
120	2	227,12	252,41	164052,47	164092,26	227,29	254,97	164086,35	164185,43
120	3	227,12	252,41	164073,02	164121,09	227,34	255,06	164090,93	164173,15
120	4	227,12	252,41	164045,73	164118,79	227,26	254,46	164083,43	164172,10

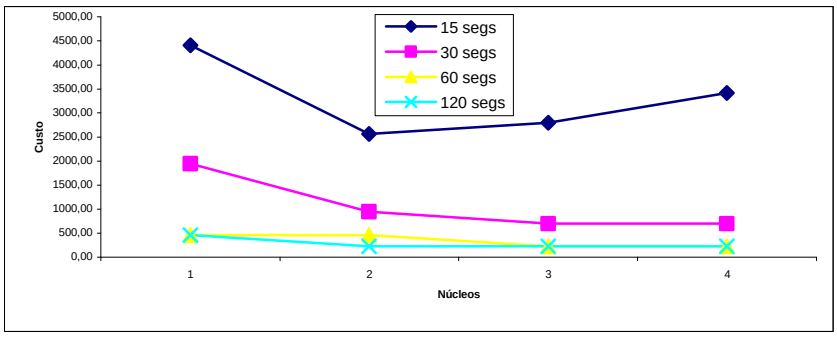

Figura 4: Comportamento médio do PGVILS no PADC1 em relação ao n° de núcleos

Pela Tabela 1 e Figura 4, pode-se observar que à medida que o número de núcleos aumenta há uma tendência de melhoria na qualidade da solução final. Entretanto, em alguns casos, como o relatado na Figura 4, soluções finais obtidas usando-se dois núcleos são melhores que aquelas com três ou quatro núcleos. Isto ocorre devido à aleatoriedade inerente ao método, a qual faz com que a busca caminhe em direções diferentes no espaço de soluções. Assim, não há garantia sempre de que a escolha das regiões de busca feita usando-se três ou quatro núcleos seja mais promissora do que aquela realizada com dois núcleos.

Na Tabela 2 são comparados os resultados obtidos nas abordagens exata (CPLEX), heurística sequencial (GVILS) e heurística paralela (PGVILS) com quatro núcleos. A última coluna apresenta a melhora do valor médio das soluções obtidas pelo PGVILS em relação aos obtidos pelo GVILS, de acordo com $MelhoraMédia = (Média_{GVILS} - Média_{PGVILS}) / Média_{PGVILS}$. O desvio é calculado como segue: $Desvio = (Média - Melhor_{Lit}) / Melhor_{Lit}$, sendo $Melhor_{Lit}$ o melhor valor encontrado na literatura para o problema-teste em questão.

Pela Tabela 2 observa-se que em quinze dos dezesseis casos analisados, o algoritmo paralelo PGVILS obteve soluções finais de maior qualidade, em média, que o GVILS. O PGVILS apresentou melhora significativa nos problemas PADC1 e PADC2. Já nos outros problemas, a melhora foi amena. Nos testes de quinze segundos, a melhora foi de até 99,72%. A maior melhora foi obtida nos testes de 30 segundos: 242,13% em relação à média. Apesar da melhora observada em 15 e 30 segundos, a resolução do problema em tais tempos não é apropriada, principalmente no caso dos problemas PADC1 e PADC2. Nesses casos, o PGVILS ainda apresenta grande variabilidade na qualidade das soluções finais encontradas. Entretanto, percebe-se que 60

segundos é tempo suficiente para a resolução satisfatória do problema. Nesse tempo, o PGVILS é superior ao CPLEX em dois dos quatro problemas. Outro ponto a ser observado é que o PGVILS foi capaz de alcançar a melhor solução da literatura para o problema PADC1 com apenas 15 segundos de processamento e, para o PADC2, a partir de 30 segundos.

TABELA 2 – Comparação do PGVILS com outras abordagens

Probl.	Tempo	CPLEX	GVILS			PGVILS			Melhora
	(segs)		Melhor	Média	Desvio	Melhor	Média	Desvio	Média
PADC1	15	90238,416	227,12	4408,83	1841,19%	227,12	3421,54	1406,49%	28,855%
PADC2	15	261,834	257,40	4431,14	1655,53%	254,80	2218,68	779,00%	99,720%
PADC3	15	164043,55	164090,31	164107,81	0,04%	164058,33	164103,88	0,04%	0,002%
PADC4	15	164065,21	164140,49	164271,73	0,13%	164140,49	164269,16	0,13%	0,002%
PADC1	30	90238,416	227,12	1947,76	757,59%	227,12	694,31	205,70%	180,533%
PADC2	30	261,834	252,41	1681,23	566,07%	252,41	491,40	94,68%	242,129%
PADC3	30	164043,31	164073,84	164105,82	0,04%	164072,84	164105,40	0,04%	0,000%
PADC4	30	164065,21	164137,04	164242,80	0,12%	164136,89	164232,21	0,11%	0,006%
PADC1	60	230,3	227,12	460,77	102,88%	227,12	227,42	0,13%	102,609%
PADC2	60	259,746	252,41	258,25	2,31%	252,41	256,88	1,77%	0,531%
PADC3	60	164015,67	164073,84	164101,43	0,04%	164080,36	164096,15	0,04%	0,003%
PADC4	60	164055,68	164111,47	164231,00	0,11%	164107,97	164203,98	0,09%	0,016%
PADC1	120	230,30	227,12	460,64	102,82%	227,12	227,26	0,06%	102,692%
PADC2	120	254,22	252,41	254,29	0,74%	252,41	254,46	0,81%	-0,069%
PADC3	120	164043,55	164048,15	164097,44	0,04%	164045,73	164083,43	0,03%	0,009%
PADC4	120	164085,84	164110,29	164204,40	0,09%	164118,79	164172,10	0,07%	0,020%

4. Conclusões

Este trabalho apresentou uma proposta de paralelização da metaheurística *Iterated Local Search* (ILS) e desenvolveu um algoritmo heurístico paralelo, denominado PGVILS, aplicado ao problema de planejamento operacional de lavra com alocação dinâmica de caminhões.

O algoritmo desenvolvido combina os procedimentos heurísticos ILS, *Variable Neighborhood Descent* (VND) e a fase de construção GRASP e explora a capacidade de multiprocessamento da geração atual de computadores.

O algoritmo PGVILS foi testado usando-se 1, 2, 3 e 4 núcleos de processamento e interrompido após 15, 30, 60 e 120 segundos de execução. Observou-se, como já era esperada, melhoria na qualidade das soluções finais quando comparado com sua versão sequencial. Comparado com os resultados produzidos pelo otimizador CPLEX, o algoritmo proposto conseguiu superá-lo com relação às melhores soluções encontradas e produzir soluções médias competitivas com este.

Como trabalho futuro propõe-se a implementação de uma versão distribuída do algoritmo PGVILS de forma a executá-lo em um maior número de núcleos.

Agradecimentos

Os autores agradecem ao CNPq, processo 474831/2007-8, e à FAPEMIG, processo CEX 2991-06.1/07, pelo apoio ao desenvolvimento da presente pesquisa.

Referências

ALVARENGA, G. B. *Despacho ótimo de caminhões numa mineração de ferro utilizando algoritmo genético com processamento paralelo*. Dissertação (Mestrado em Engenharia Elétrica), Programa de Pós-Graduação em Engenharia Elétrica, UFMG, Belo Horizonte, 1997.

CAMPOS, G. D.; YOSHIZAKI, H. T. Y.; BELFIORE, P. P. Algoritmo Genético e computação paralela para problemas de roteirização de veículos com janelas de tempo e entregas fracionadas. *Gestão e Produção*, v. 13, p. 271-281, 2006.

CHANDA, E. K. C.; DAGDELEN, K. Optimal blending of mine production using goal programming and interactive graphics systems, *International Journal of Surface Mining, Reclamation and Environment*, v. 9, p. 203-208, 1995.

COELHO, I. M.; RIBAS, S.; SOUZA, M. J. F. Um algoritmo baseado em GRASP, Iterated Local Search para a otimização do planejamento operacional de lavra. In: Anais do *XI Encontro de Modelagem Computacional*, Volta Redonda (RJ), 8 p., 2008.

COSTA, F. P.; SOUZA, M. J. F. e PINTO, L. R. Um modelo de alocação dinâmica de caminhões. *Revista Brasil Mineral*, v. 231, p. 26-31, 2004.

COSTA, F. P. *Aplicações de técnicas de otimização a problemas de planejamento operacional de lavras em mina a céu aberto*. 141 p. Dissertação (Mestrado em Engenharia Mineral) - Programa de Pós-Graduação em Engenharia Mineral, UFOP, Ouro Preto, 2005.

DEAN, J. e GHEMAWAT, S. MapReduce: Simplied Data Processing on Large Clusters, In OSDI'04, 6th Symposium on Operating Systems Design and Implementation, Sponsored by USENIX, in cooperation with ACM SIGOPS, pg 137-150, 2004.

FEO, T. A.; RESENDE, M. G. C. Greedy randomized adaptive search procedures. *Journal of Global Optimization*, v. 6, p. 109–133, 1995.

GUIMARÃES, I. F.; PANTUZA, G. e 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), Bauru, CD-ROM, 11 p., 2007.

LÄMMEL, R. Google's MapReduce Programming Model, Microsoft Corp. In *www.cs.vu.nl/ ~ralf/MapReduce/paper.pdf* Acesso em 03/03/2009.

LOURENÇO, H. R.; MARTIN, O. C.; STUTZLE, T. Iterated Local Search. In: GLOVER, F. e KOCHENBERGER, G. (eds). *Handbook of Metaheuristics*, Kluwer Academic Publishers, 2003.

MERSCHMANN, L.H.C. *Desenvolvimento de um sistema de otimização e simulação para análise de cenários de produção em minas a céu aberto*. Dissertação (Mestrado em Engenharia de Produção) - Programa de Engenharia de Produção/COPPE, UFRJ, Rio de Janeiro, 2002.

MLADENOVIC, N.; HANSEN, P. A Variable Neighborhood Search. *Computers and Operations Research*, v. 24, p. 1097-1100, 1997.

MUHAMMAD, R. B. A Parallel Local Search Algorithm for Euclidean Steiner Tree Problem, In *IEEE Proceedings of Seventh International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD 2006)*, p. 157-164, 2006.

RESENDE, M. G. C e RIBEIRO, C. C. Parallel Greedy Randomized Adaptive Search Procedures. *Parallel Metaheuristics: A New Class of Algorithms* (E. Alba, ed.), 315-346, Wiley, 2005.

RODRIGUES, L. F. *Análise comparativa de metodologias utilizadas no despacho de caminhões em minas a céu aberto*. 86 p. Dissertação (Mestrado em Engenharia de Produção), Programa de Pós-Graduação em Engenharia de Produção, UFMG, Belo Horizonte, 2006.

WHITE, J. W. e OLSON, J. P. Computer-based dispatching in mines with concurrent operating objectives. *Mining Engineering*, v. 38, n. 11, p. 1045–1054, 1986.