UMA AMPLIAÇÃO PARA A VIZINHANÇA DO ALGORITMO REDINV-SA APLICADO AO PROBLEMA QUADRÁTICO DE ALOCAÇÃO

Leopoldo Marchiori Rodrigues [1]
Universidade Federal do Espírito Santo - CT - Departamento de Informática
Av. Fernando Ferrari, s/n - Campus de Goiabeiras -Vitória - ES - 29060-970
e-mail: leopoldor@zipmail.com.br

Maria Cristina Rangel
Universidade Federal do Espírito Santo - CT - Departamento de Informática
Av. Fernando Ferrari, s/n - Campus de Goiabeiras -Vitória - ES - 29060-970
e-mail: crangel@inf.ufes.br

Resumo

Uma abordagem algébrica e combinatorial para o Problema Quadrático de Alocação permitem encontrar resultados que podem ser utilizados para informar meta-heurísticas sobre a estrutura do problema. Com base nisso, Abreu et al. [AQB99] apresentaram um *Simulated Annealing*, chamado REDINV-SA. Neste trabalho é apresentada uma expansão da vizinhança do REDINV-SA com o objetivo de torná-lo ainda mais eficiente. Alguns resultados comparativos são apresentados.

Palavras-chaves: Problema Quadrático de Alocação, Meta-heurísticas e *Simulated Annealing*.

Abstract

An algebraic and combinatorial approach of the Quadratic Assignment Problem avoids finding results that can be applied to metaheuristics. Abreu et al. [AQB99] explored this fact and proposed a Simulated Annealing, called REDINV-SA. In this paper, an expansion of neighborhood of REDINV-SA is presented with the objective become it more efficient. Those comparative results are presented.

Keywords: Quadratic Assignment Problem, Metaheuristics and Simulated Annealing.

1- Introdução

Este trabalho aborda o Problema Quadrático de Alocação, PQA, por Matemática Discreta. Na seção **2**, apresenta-se o Problema de Alocação Linear, PAL, como uma relaxação do Problema Quadrático de Alocação. A heurística *Simulated Annealing*, aplicada ao PQA, e o algoritmo REDINV-SA proposto por Abreu et al. [AQB99] são abordados na seção **3**. São descritos o modo de construção de uma vizinhança e a busca local a partir de uma permutação de arestas, visando a redução do número de inversões dessa permutação. Esse trabalho propõe uma ampliação na vizinhança e uma modificação no método de busca local, exposto em **4.1**. O REDINV-SA possui a cardinalidade de sua vizinhança de três ou quatro elementos para qualquer dimensão *n* do PQA. A vizinhança proposta possui *n* elementos. Esse novo procedimento utiliza o REDINV-SA na construção de cada um dos *n* elementos da vizinhança e mantém a aleatoriedade característica da heurística *Simulated Annealing* ao realizar a busca aleatória em alguns casos. Um exemplo detalhado de utilização do algoritmo é apresentado em **4.2**. Na seção **5** são descritos os testes realizados e a conclusão, a partir da

[1] Estudante de Engenharia de Computação – bolsista de Iniciação Científica

implementação do algoritmo apresentado, para várias instâncias do QAPLIB [BKR97], assim como os parâmetros, tais como custo médio, custo mínimo, tempo de processamento, entre outros, usados para a avaliação dos resultados.

2- O Problema Quadrático de Alocação

Considere duas matrizes F e D, com diagonal nula, *n x n*, simétricas, e cujas coordenadas reais são não-negativas. O Problema Quadrático de Alocação, PQA (F,D), é modelado pela função objetivo

$$Z(\varphi^*) = \min_{\varphi \in \Pi_n} \sum_{i<j} f_{ij} d_{\varphi(i)\varphi(j)}, \tag{2.1}$$

onde Π_n é o conjunto de todas as permutações dos elementos de $\Omega^n = \{1,...n\}$. Deseja-se encontrar $\varphi^* \in \Omega^n$ cujo valor seja o menor custo para uma atribuição em que pares de facilidades são alocadas a pares de localidades e reciprocamente. O PQA pode ser tratado como o problema do arranjo físico representado por sobreposição de cliques de *n* vértices, as quais são designadas por K_F, fluxo entre as facilidades e K_D, distâncias entre as localidades. A sobreposição é definida por uma permutação dos vértices $\varphi \in \Pi_n$, cuja solução ótima é denotada por φ^*.

Os elementos das matrizes $F = [f_{ij}]$ e $D = [d_{kl}]$, que definem a instância PQA (F,D), são armazenados em vetores-linha $\boldsymbol{F} = (f_z)$ e $\boldsymbol{D} = (d_z)$, ambos de dimensão $N = C_{n,2}$, cujas coordenadas são dispostas segundo a bijeção

$$\psi(i,j) = ((i-1)n - i(i+1)/2) + j, \tag{2.2}$$

que associa o par (i,j) a um número $z \in \{1,...N\}$.

Considere a matriz $Q = \boldsymbol{F}^t\boldsymbol{D}$, cujos coeficientes correspondem às parcelas da função objetivo (2.1), de modo que para cada φ, os coeficientes estejam na matriz Q. O Problema de Alocação Linear, PAL(Q), definido por

$$Z(\xi^*) = \min_{\xi \in \Pi_N} \sum_{i=1}^{N} f_i d_{\xi(i)} \tag{2.3}$$

é uma relaxação linear do PQA(F,D), considerando-se que o conjunto das soluções lineares contém o das soluções quadráticas.

A partir de cada atribuição dos vértices de K_F sobre K_D, representada por φ em Π_n,

$$\varphi = \begin{pmatrix} 1 & 2 & ... & n \\ \varphi(1) & \varphi(2) & ... & \varphi(n) \end{pmatrix}, \tag{2.4}$$

determina-se, por intermédio de ψ em (2.2), a correspondente sobreposição das arestas das referidas cliques, denotada por $\xi \in \Pi_N$, isto é,

$$\xi = \begin{pmatrix} \psi(1,2) & \psi(1,3) & ... & \psi(n-1,n) \\ \psi(\varphi(1),\varphi(2)) & \psi(\varphi(1),\varphi(3)) & ... & \psi(\varphi(n-1),\varphi(n)) \end{pmatrix}. \tag{2.5}$$

Dada uma solução ξ do PAL(Q), nem sempre existe uma sobreposição de vértices das cliques compatível com a de arestas. Se tal sobreposição φ existir, deve obedecer a seguinte equação:

$$f_p d_{\xi(p)} = f_{\psi^{-1}(p)} d_{\psi^{-1}(\xi(p))} = f_{ij} d_{kl}, \tag{2.6}$$

para $p = 1, ..., N$; $\varphi(i) = k$ e $\varphi(j) = l$; onde i, j, k, l = 1,..., n. Assim, diz-se que ξ é uma **solução viável** para PQA(F,D).

O custo da solução do PAL(Q) associado a $\xi \in \Pi_N$, resulta do produto escalar

$$Z(\xi) = \langle F, \xi(D) \rangle \tag{2.7}$$

onde $\xi(D) = (d_{\xi(1)}, d_{\xi(2)}, ..., d_{\xi(N)})$. O custo mínimo é $Z^* = \langle F^-, D^+ \rangle$, quando $\boldsymbol{F}^-$ e $\boldsymbol{D}^+$ são resultados da ordenação não-crescente e não-decrescente de $\boldsymbol{F}$ e $\boldsymbol{D}$, respectivamente, [HLP52]. O problema

PAL(Q*), onde $Q^* = (\boldsymbol{F}^-)^t \boldsymbol{D}^+$ possui o mesmo conjunto de soluções viáveis de PAL(Q) e, por conseguinte, a mesma solução ótima cujo custo é o traço de Q*, notado por **tr(Q*)**. Tal valor define um limite inferior para o PQA(F,D).

Considere Φ_F e Φ_D as respectivas permutações auxiliares que conduzem $\boldsymbol{F}$ e $\boldsymbol{D}$ aos vetores ordenados $\boldsymbol{F}^-$ e $\boldsymbol{D}^+$. Se $\rho \in \Pi_N$ é uma solução do PAL (Q*), tem-se que a composta

$$\xi = \Phi^{-1}_D \circ \rho \circ \Phi_F \tag{2.8}$$

corresponde a uma solução do PAL(Q).

3- *Simulated Annealing* e Algoritmo REDINV-SA

Simulated Annealing (SA), proposto primeiramente por Kirkpatrick et al. [KGV83], é uma técnica baseada em idéias da mecânica estatística e motivada por uma analogia ao comportamento de sistemas físicos na presença de um banho de calor, proposto por Metrópolis et al. [MRRTT53]. O termo *annealing* refere-se ao processo de cozimento do material, lentamente, até que se atinja um estado de equilíbrio, isto é, a melhor solução da heurística. Dada uma solução atual uma nova solução é obtida pela técnica SA determinando-se a diferença Δ entre o valor objetivo da solução atual e o de uma solução na sua vizinhança. Se a diferença for favorável, a solução vizinha será aceita sem restrições, caso contrário, a solução vizinha poderá ser aceita segundo uma probabilidade $e^{-\Delta/T}$, onde T é um parâmetro de controle chamado *temperatur*a. Claramente à medida que a temperatura decresce, a probabilidade de se aceitar uma solução vizinha de qualidade muito inferior à solução atual decai exponencialmente à zero. Espera-se que a solução final seja ótima quando a temperatura aproximar-se de zero, tendo esta sido decrescida bem lentamente entre iterações bem sucedidas.

Muitos autores que têm reportado o uso de *simulated annealing*, definem uma estrutura de vizinhança para seu problema em particular e escolhem a próxima solução potencial aleatoriamente a partir do conjunto de vizinhos da solução atual. Segundo Conolly, [Co90] esta abordagem poderia ser ineficiente por dois motivos. Primeiramente, melhoramentos potenciais poderão ser perdidos sob baixa temperatura, devido à natureza aleatória da busca. Além disso, tende-se a mover para fora de uma região de ótimo global dado a possibilidade de uma mudança brusca na solução. Conolly sugere um aperfeiçoamento no procedimento SA aplicado ao problema de alocação quadrática. Baseado em testes, ele concluiu que com vizinhanças geradas seqüencialmente o algoritmo SA é mais eficiente.

O algoritmo REDINV-SA, proposto por Abreu et al. [AQB99], avalia a função objetivo

$$z = \sum_{i,j=1}^{n} f_{ij} d_{\varphi(i)\varphi(j)} \tag{3.1}$$

onde $F = [f_{ij}]$ e $D = [d_{kl}]$ são matrizes que definem a instância do Problema Quadrático de Alocação e se caracterizam por serem de dimensão *n x n*, simétricas, terem diagonal nula e coordenadas reais não-negativas.

A vizinhança de $\varphi \in \Pi_n$, denotada por Viz(φ), é constituída por permutações $\varphi' \in \Pi_n$ resultantes da troca, dois a dois, de quatro vértices em $\varphi \in \Pi_n$. Essa troca visa reduzir o número de inversões de ρ, razão do nome REDINV-SA. O objetivo de reduzir o número de inversões segue da aplicação do Teorema das Inversões provado em [Ran00], por Rangel. Esse teorema mostra uma relação entre o número de inversões de permutações e seus custos associados, o que possibilita avaliar as soluções do PQA pelo número de inversões e não somente pelos custos. Reduzindo o número de inversões é possível que o custo da permutação decresça.

As posições das duas arestas a serem trocadas em ρ são denominadas *m* e *j*. Escolhe-se a posição *m* tal que corresponde ao máximo de $| i - \rho(i) |$, e a posição *j* tal que corresponde ao mínimo de $\{| m - \rho(i) | + | i - \rho(m) |\}$, $\forall i = 1, ..., N$. Calcula-se então os vértices a serem trocados em φ utilizando as seguintes expressões $a_1b_1 = \psi^{-1}(\xi_\varphi(\phi_F^{-1}(m)))$ e $a_2b_2 = \psi^{-1}(\xi_\varphi(\phi_F^{-1}(j)))$, onde a_1b_1 e a_2b_2

são os dois pares de vértices. Tomando-se cada elemento de a_1b_1 e combinando-o com os dois de a_2b_2 obtém-se 4 trocas. Caso existam elementos iguais entre a_1b_1 e a_2b_2, só se obtém 3 trocas. Calcula-se o custo de cada permutação de Viz(φ) e se escolhe a que tiver o menor custo.

Exemplo 3.1: Considere a instância Gavett e Plyter [GP66], onde:

$$F = \begin{bmatrix} 0 & 28 & 25 & 13 \\ 28 & 0 & 15 & 4 \\ 25 & 15 & 0 & 23 \\ 13 & 4 & 23 & 0 \end{bmatrix} \text{ e } D = \begin{bmatrix} 0 & 6 & 7 & 2 \\ 6 & 0 & 5 & 6 \\ 7 & 5 & 0 & 1 \\ 2 & 6 & 1 & 0 \end{bmatrix}, \text{ ou } \boldsymbol{F} = (28\ 25\ 13\ 15\ 4\ 23) \text{ e } \boldsymbol{D} = (6\ 7\ 2\ 3\ 6\ 1).$$

Ordenando em ordem não crescente e não decrescente, obtém-se $\boldsymbol{F}^- = (28\ 25\ 23\ 15\ 13\ 4)$ e $\boldsymbol{D}^+ = (1\ 2\ 5\ 6\ 6\ 7)$. As trocas feitas coordenadas são armazenadas nas permutações ϕ_F e ϕ_D. Para o exemplo tem-se $\phi_F = (1\ 2\ 5\ 4\ 6\ 3)$ e $\phi_D = (4\ 6\ 2\ 3\ 5\ 1)$.

Deseja-se construir a vizinhança de $\varphi_0 = (3\ 4\ 1\ 2)$, cujo custo é $z_0 = 460$. Sendo a permutação de arestas associada $\xi = (6\ 2\ 4\ 3\ 5\ 1)$, onde $N = C_{n,2} = 6$, pois $n = 4$. Como $\rho = \phi_D \circ \xi \circ \phi_F^{-1}$, tem-se $\rho = (4\ 6\ 2\ 3\ 5\ 1) \circ (6\ 2\ 4\ 3\ 5\ 1) \circ (1\ 2\ 6\ 4\ 3\ 5) = (1\ 6\ 4\ 2\ 3\ 5)$. Calculando as diferenças $| i - \rho(i) |$, $\forall\ i = 1,...,N$, onde cada i se refere a uma posição em ρ:

$$\begin{array}{llll} i = 1 \Rightarrow & |1 - \rho(1)| = |1 - 1| = 0 & i = 4 \Rightarrow & |4 - \rho(4)| = |4 - 2| = 2 \\ i = 2 \Rightarrow & |2 - \rho(2)| = |2 - 6| = 4 & i = 5 \Rightarrow & |5 - \rho(5)| = |5 - 3| = 2 \\ i = 3 \Rightarrow & |3 - \rho(3)| = |3 - 4| = 1 & i = 6 \Rightarrow & |6 - \rho(6)| = |6 - 5| = 1. \end{array}$$

No caso de haver duas ou mais posições que correspondam ao máximo de $| i - \rho(i) |$ o algoritmo escolhe a menor posição. No exemplo somente a posição i = 2 satisfaz a essa condição, então **m = 2**. Conhecendo *m*,os valores de $\{| m - \rho(i) | + | i - \rho(m) |\}$, $\forall\ i = 1,...,\ N$ são

$$\begin{array}{ll} i = 1 \Rightarrow & |2 - \rho(1)| + |1 - \rho(2)| = |2 - 1| + |1 - 6| = 6 \\ i = 2 \Rightarrow & |2 - \rho(2)| + |2 - \rho(2)| = |2 - 6| + |2 - 6| = 8 \\ i = 3 \Rightarrow & |2 - \rho(3)| + |3 - \rho(2)| = |2 - 4| + |3 - 6| = 5 \\ i = 4 \Rightarrow & |2 - \rho(4)| + |4 - \rho(2)| = |2 - 2| + |4 - 6| = 2 \\ i = 5 \Rightarrow & |2 - \rho(5)| + |5 - \rho(2)| = |2 - 3| + |5 - 6| = 2 \\ i = 6 \Rightarrow & |2 - \rho(6)| + |6 - \rho(2)| = |2 - 5| + |6 - 6| = 3. \end{array}$$

Caso haja duas ou mais posições que correspondam ao mínimo de $\{| m - \rho(i) | + | i - \rho(m) |\}$ escolhe-se a menor posição. No exemplo as posições 4 e 5 satisfazem a essa condição, então se escolhe **j = 4**.
Com os valores de *m* e *j* em mãos, pode-se calcular os vértices a serem trocados em φ_0, como segue:

$$a_1b_1 = \psi^{-1}(\xi(\phi_F^{-1}(m))) = \psi^{-1}(\xi(\phi_F^{-1}(1))) = (1\ 3)$$
$$a_2b_2 = \psi^{-1}(\xi(\phi_F^{-1}(j))) = \psi^{-1}(\xi(\phi_F^{-1}(5))) = (1\ 4).$$

As permutações com as respectivas combinações de vértices a serem trocados são: (3 4 1 2), trocando-se os vértices 1 e 1, que não tem efeito; (3 1 4 2), para os vértices 1 e 4; (1 4 3 2), combinado os vértices 3 e 1; finalmente (4 3 1 2), trocando-se os vértices 3 e 4. Então, a vizinhança construída foi Viz (3 4 1 2) = {(3 1 4 2), (1 4 3 2), (4 3 1 2)}. Calculando-se o custo de cada permutação da vizinhança, tem-se a φ' = (4 3 1 2) com custo z' = 419, que é o menor deles. A partir desse elemento pode-se construir a próxima vizinhança da mesma maneira que se procedeu no exemplo.

4- REDINV-SA-NOVO

A técnica do *Simulated Annealing* é aplicada ao Problema Quadrático de Alocação por alguns pesquisadores, definindo seus parâmetro e funções com base em diferentes aspectos. Neste trabalho, a escala *annealing* para redução de temperatura é a adotada por Kirkpatrick et al. [KGV83]:

$$T(t + 1) = \alpha T(t) \qquad (4.1)$$

onde $t \geq 0$ e $\alpha = 0{,}9$. A cada temperatura, M iterações são feitas até que se atinja o "estado de equilíbrio". A dimensão de M, importante na convergência do algoritmo, é $M = 2n$, fixada por Burkard e Rendl [BR83]. O critério de parada utilizado foi o alcance de um valor pré-definido de temperatura T(t), suficientemente pequeno. A temperatura inicial é $T_0 = LimSup$, onde *LimSup*, limite superior universal, é a soma da diagonal secundária de Q*. O critério de parada é $T(t) < LimInf$, onde $LimInf = tr(Q^*)$ é o limite inferior universal das soluções viáveis.

REDINV-SA-NOVO (instância,n);
Gerar aleatoriamente $\varphi_0 \in \Pi_n$ e calcular z_0;
t = 0;
T(0) = *LimSup*;
Enquanto T (t) > *LimInf* faça
Enquanto $M \leq 2n$ faça
ConstViz-BuscaLocal (φ_0), veja seção **4.1**;
Avaliar $\Delta z = z' - z_0$;
Se $\Delta z < 0$
Então faça
$\varphi_0 \leftarrow \varphi'$;
Caso contrário faça
Se $\Delta z \geq 0$
Se *random* (0, 1) < $\exp^{-\Delta z / T}$
Então faça
$\varphi_0 \leftarrow \varphi'$;
FimSe;
FimSe;
FimSe;
FimEnquanto;
T (t) = α T (t);
t = t + 1;
FimEnquanto;
Fim REDINV-SA-NOVO;

4.1- Vizinhança proposta e Busca Local

Deseja-se construir uma vizinhança de uma permutação de vértices $\varphi_0 \in \Pi_n$, armazenada no conjunto VizNova = $\{(\varphi_1', z_1'), (\varphi_2', z_2'), ..., (\varphi_n', z_n')\}$, onde as $\varphi_k' \in \Pi_n$, k = 1,..., n, são resultantes da troca de dois elementos em φ_0 e z_k' é o custo calculado por (3.1) de uma permutação φ_k'.

A partir de uma permutação de arestas $\rho \in \Pi_N$ correspondente à φ_0 calculam-se as diferenças $| i - \rho(i) |$, $\forall$ i = 1, ..., N. De acordo com o algoritmo REDINV-SA, escolhe-se a posição *m* tal que corresponde ao máximo de $| i - \rho(i) |$, e a posição *j* tal que corresponde ao mínimo de $\{| m - \rho(i) | + | i - \rho(m) |\}$, $\forall$ i = 1, ..., N. Observando o exemplo **3.1** constata-se que a vizinhança gerada pelo REDINV-SA tem apenas três ou quatro permutações.

A modificação proposta neste trabalho consiste na ampliação da vizinhança, de maneira que se tenham *n* permutações. Escolhem-se *n* valores de *i*, i = 1,...,N, tais que forneçam as maiores diferenças $| i - \rho(i) |$. Veja o exemplo:

Exemplo 4.1.1: Considere a mesma permutação do exemplo **3.1**. Neste caso são escolhidas as posições $m_1 = 2$, $m_2 = 4$, $m_3 = 5$ e $m_4 = 3$, que são consideradas as *n* melhores posições *m*. Em seguida, para cada m_k, k = 1,..., n, é calculado seu respectivo j_k, obtendo-se $j_1 = 4$, $j_2 = 3$, $j_3 = 3$, $j_4 = 2$. Veja este exemplo detalhado na seção **4.2**.

Para cada *k* (k = 1,..., n) o algoritmo constrói o par (φ_k', z_k'), onde φ_k' é a permutação, dentre as três ou quatro fornecidas por m_k e j_k, cujo valor do custo z_k' é mínimo. Todos os pares (φ_k', z_k') são armazenados no conjunto VizNova.

A VizNova é então ordenada em ordem não-decrescente pelo custo de cada par (φ_k', z_k'), $1 \leq k \leq n$. Se $\Delta z = z_1' - z_0 < 0$ para o primeiro elemento da VizNova, já ordenada, esse elemento é escolhido. Caso contrário, escolhe-se aleatoriamente um dentre os *n/2* primeiros elementos desse conjunto. A aleatoriedade nesse caso demonstra a utilização de uma característica importante da heurística SA.

A seguir, é apresentado o algoritmo, em pseudo-código, que descreve os passos da construção da vizinhança e da busca local propostos:

ConstViz-BuscaLocal (φ_0);
Determinar ξ através da expressão 2.5;
Determinar $\rho = \phi_D \circ \xi \circ \phi_F$ De211.3201.2Ei

Denomina-se cada posição *i* como m_i', para i = 1, ..., N. Ordenam-se então todos os m_i' de acordo com a ordem não-crescente de | i -ρ(i) |. Tomando-se os n primeiros m_i', obtém-se lista_m = (m_k), k=1,...,n. De acordo com o exemplo, lista_m = (m_1, m_2, m_3, m_4) = (m_2', m_4', m_5', m_3') = (2, 4, 5, 3). Observe que em caso de empate no valor de | i -ρ(i) |, toma-se o m_i' que aparece primeiro. Por exemplo, houve empate entre m_4' e m_5' e entre m_3' e m_6', dando-se preferência a m_4' e m_3', respectivamente. Cada solução da vizinhança que será construída, e seu respectivo custo, é guardado em VizNova = [(φ_1', z_1'), (φ_2', z_2'), ..., (φ_n', z_n')]. Cada elemento de VizNova é obtido a partir de um m_k e um j_k, k = 1, ..., n. Segue a construção da VizNova.

Para $\mathbf{m_1=2}$, calculam-se os valores de {| m_1 - ρ(i) | + | i - ρ(m_1) |}, ∀ i = 1,..., N, analogamente como na seção **3**, temos $\mathbf{j_1 = 4}$. Calculando os vértices a serem trocados em φ_0, tem-se a_1b_1 = (1 3) e a_2b_2 = (1 4). As permutações e as combinações de vértices a serem trocados entre os pares (1,3) e (1,4) gerando a Viz(3 4 1 2) = {(3 1 4 2), (1 4 3 2), (4 3 1 2)}. Calculando-se o custo de cada permutação e escolhendo o menor deles temos o primeiro elemento (φ_1', z_1') da vizinhança proposta. Portanto, VizNova = [((4 3 1 2), 419)].

Para $\mathbf{m_2 = 4,}$ tem-se $\mathbf{j_2 = 3}$ associado, de modo que min {| m_2 - ρ(i) | + | i -ρ(m_2) |}, para i = 1,..., N, de maneira análoga ao cálculo de $\mathbf{j_1}$. Tem-se o segundo elemento (φ_2', z_2') da VizNova = [((4 3 1 2), 419) , ((3 1 4 2), 478)]. Seguindo este procedimento para $\mathbf{m_3}$ e $\mathbf{m_4}$, chega-se à vizinhança proposta VizNova = [((4 3 1 2), 419) , ((3 1 4 2), 478) , ((1 4 3 2), 463), ((1 4 3 2), 463)]. A VizNova é então ordenada em ordem não-decrescente pelo custo de cada par (φ_i', z_i'), $1 \le i \le n$. Escolhe-se adequadamente, conforme já descrito na seção **4.1**, a solução da vizinhança (VizNova). Portanto,

VizNova = [((4 3 1 2), 419), ((1 4 3 2), 463), ((1 4 3 2), 463), ((3 1 4 2), 478)]

$\Delta z = z_1' - z_0 = 419 - 502 = -83 \Rightarrow \Delta z < 0$

Logo, a permutação escolhida na vizinhança gerada pela busca é: φ' = (4 3 1 2).

5- Testes realizados e conclusões

Todos os testes foram realizados utilizando um processador PENTIUM MMX 233MHz. O algoritmo apresentado foi implementado em C (Padrão ANSI).

Foram testadas 43 instâncias simétricas da QAPLIB, das quais 24 são de ordem baixa ($12 \le n \le 18$) e 19 são de ordem alta ($n \ge 20$). Em cada uma delas foram feitos 16 testes, de modo que se obtivessem os melhores resultados possíveis, tanto para as instâncias de ordem baixa quanto para as de ordem alta. Para avaliar o quão distante estão as soluções encontradas em relação aos respectivos valores ótimos, retirados da QAPLIB, foram escolhidos os parâmetros listados a seguir.

Para cada teste de cada instância, foram obtidos os seguintes resultados:

Custo Médio ($\mathbf{Z_{med}}$) – Média dos custos obtidos durante a execução do algoritmo. A cada iteração, a busca local encontra um custo melhor na vizinhança. O custo médio é a média desses custos.

Custo Mínimo ($\mathbf{Z_{min}}$) – Mínimo dos custos calculados pelas iterações do algoritmo. É o menor de todos os melhores custos encontrados nas buscas locais.

Variação Média Percentual (ρ_{med}) – Variação percentual do Custo Médio em relação ao custo ótimo ($\mathbf{Z_{otimo}}$) da instância.

Onde $\rho_{med} = ((\mathbf{Z_{med}} - \mathbf{Z_{otimo}}) / \mathbf{Z_{otimo}}) * 100$.

Variação Mínima Percentual (ρ_{min}) – Variação percentual do Custo Mínimo em relação ao custo ótimo ($\mathbf{Z_{otimo}}$) da instância.

Onde $\rho_{min} = ((\mathbf{Z_{min}} - \mathbf{Z_{otimo}}) / \mathbf{Z_{otimo}}) *100$.
Obs.: Caso o ótimo seja encontrado, $\rho_{min} = 0$, pois $\mathbf{Z_{min}} = \mathbf{Z_{otimo}}$.

Tempo de processamento (t) – Tempo, em segundos, gasto pelo algoritmo para executar todas as iterações do algoritmo.

Após todos os testes, para avaliar o comportamento do algoritmo em cada instância, foram obtidos os seguintes resultados:

Custo Médio Total ($\mathbf{Z_{medT}}$) – média de todos $\mathbf{Z_{med}}$;

Custo Mínimo Total ($\mathbf{Z_{minT}}$) – média de todos $\mathbf{Z_{min}}$;

Variação Média Percentual Total (ρ_{medT}) – média de todos ρ_{medT} ;

Variação Mínima Percentual Total (ρ_{minT}) – média de todos ρ_{min};
Obs.: Caso o ótimo seja encontrado em algum dos testes, considera-se $\rho_{minT} = 0$;

Tempo de processamento – há duas possibilidades:
Tempo Médio (t_m) – Média de todos os tempos (**t**).
Tempo Ótimo (t_o) – Média dos tempos (**t**) dos testes em que o ótimo foi encontrado.

Obs.: Caso o ótimo seja encontrado em um ou mais testes da instância, somente o Tempo Ótimo é exibido.

As tabelas apresentam oito colunas. A primeira e a segunda contêm o nome e o valor do ótimo de cada instância (retirados da QAPLIB). Em seguida são exibidos, para todas as instâncias, o Custo Médio ($\mathbf{Z_{medT}}$), o Custo Mínimo ($\mathbf{Z_{medT}}$), a Variação Média Percentual (ρ_{medT}). Quando o ótimo é encontrado, apresenta-se o Tempo Ótimo ($\mathbf{t_o}$), caso contrário, são fornecidos a Variação Mínima Percentual (ρ_{minT}) e o Tempo Médio ($\mathbf{t_m}$).

5.1- Tabelas de Resultados

Instâncias	Ótimo	Z_{medT}	Z_{minT}	ρ_{medT}	ρ_{minT}	t_m	t_o
rou12	117764	124958,66	118070,75	6,26	-	-	0,85
had12	826	840,36	826,93	1,68	-	-	0,44
nug12	289	308,46	289,87	6,46	-	-	0,99
tai12a	112208	122391,00	112208,00	9,12	-	-	0,88
scr12	15705	16963,49	15705,00	7,55	-	-	1,39
chr12b	4871	7343,15	5129,37	43,49	-	-	2,74
chr12a	4776	7271,20	5146,12	50,47	-	-	2,81
had14	1362	1384,08	1363,87	1,78	-	-	1,00

Tabela 5.1: Resultados para instâncias de dimensão $12 \leq n \leq 14$.

Instâncias	Ótimo	Z_{medT}	Z_{minT}	ρ_{medT}	ρ_{minT}	t_m	t_o
tai15a	194107	204471,57	194936,93	5,36	-	-	1,85
nug15	575	601,71	575,56	4,64	-	-	2,40
rou15	177105	189189,07	179821,98	6,52	-	-	2,17
scr15	25570	28094,43	25570,00	12,08	-	-	3,73
chr15a	4948	8482,15	5272,74	79,58	6,94	8,39	-
chr15b	3995	7649,80	4759,12	94,05	21,79	8,36	-
chr15c	4752	9410,43	5772,37	96,81	22,18	8,35	-
esc16a	34	35,21	34,00	2,91	-	-	9,77
esc16b	146	146,08	146,00	0,03	-	-	3,34
nug16a	805	844,07	810,56	5,42	-	-	3,10
esc16c	80	80,72	80,00	1,83	-	-	6,42
had16	1860	1878,15	1863,00	1,31	0,08	1,78	-
had18	2679	2718,66	2681,62	1,17	-	-	2,90
nug18	965	1011,66	972,06	5,11	0,36	5,17	-
chr18a	5549	11300,02	6289,37	102,52	9,32	18,21	-
chr18b	767	1092,46	831,18	40,43	8,27	24,63	-

Tabela 5.2: Resultados para instâncias de dimensão $15 \leq n \leq 18$.

Instâncias	Ótimo	Z_{medT}	Z_{minT}	ρ_{medT}	ρ_{minT}	t_m	t_o
had20	3461	3503,87	3469,62	1,57	-	-	4,33
nug20	1285	1348,73	1298,00	5,28	0,94	7,80	-
rou20	362761	380601,12	364125,18	4,64	0,45	6,03	-
tai20a	351741	37251,78	354791,56	5,90	0,87	4,88	-
scr20	55015	60621,10	55529,00	10,54	-	-	13,51
tai25a	583628	617351,69	593885,69	5,78	1,76	9,51	-
chr25a	1898	3501,93	2501,31	88,68	34,94	63,96	-
nug30	3062	3220,34	3091,43	4,98	1,00	48,82	-
kra30a	44450	49022,14	46161,56	10,70	3,83	58,86	-
kra30b	45710	49219,82	46238,75	8,19	1,19	58,82	-
esc32b	84	107,69	94,00	27,83	10,11	167,68	-
ste36a	4763	5962,17	5283,87	25,67	12,09	199,78	-
ste36b	7926	16796,96	14459,56	107,02	75,23	325,86	-
sko49	11693	12177,57	11902,69	4,14	1,79	301,26	-
sko42	7906	8247,96	8025,50	4,87	2,16	207,66	-
sko56	17229	17993,83	17663,00	4,43	2,24	749,81	-
tai60a	3604286	3794000,25	3710183,00	5,26	2,82	713,05	-
sko64	24249	29512,36	28672,38	21,71	18,24	1219,06	-
sko81	45499	47138,16	46506,31	3,60	2,21	3029,26	-

Tabela 5.3: Resultados para instâncias de dimensão $20 \leq n \leq 81$.

5.2- Discussão dos Resultados:

A Variação Média Percentual permite avaliar o quão distante estiveram as soluções dos ótimos durante as buscas locais, enquanto a Variação Mínima Percentual fornece informação sobre qual foi a mínima distância em relação ao ótimo.

Pode-se observar nas tabelas 5.1 e 5.2 que, para instâncias de ordem $12 \leq n \leq 18$, as variações médias percentuais estão abaixo de 12,08%, atingida por scr15, com exceção das instâncias de

Christofides (ver tabela) que apresentam grande dificuldade na busca de suas soluções. O melhor resultado foi obtido com esc16b (0,03%). A variações mínimas percentuais não são apresentadas em 17 das 24 instâncias, nas quais os ótimos são alcançados; nas demais instâncias, as variações são de 0,36% para nug18, 0,08% para had16 e para as instâncias de Christofides elas variam entre 6,94% e 22,18%.

Para as instâncias de ordem $n \geq 20$, contidas na tabela 5.3, as variações médias percentuais estão abaixo de 10,70%, alcançada por kra30a, com exceção das instâncias chr25a com 88,68%, esc32b com 27,83%, ste36a com 25,67%, ste36b com 107,02% e sko64 que obteve 21,71%. Nota-se também que as variações mínimas percentuais da maioria das instâncias em que o ótimo não foi obtido estão abaixo de 3,83%, dado por kra30a, com exceção das mesmas instâncias do parâmetro anterior.

Comparando-se esses resultados com os obtidos em [AQB99] observa-se uma melhoria, principalmente em relação às instâncias de ordem $n \geq 20$, nas quais as variações médias e mínimas percentuais, calculados nesse trabalho, são menores. Para a had20 e a scr20 foi encontrado o ótimo, o que não aconteceu no REDINV-SA original.

Estes resultados indicam que o algoritmo apresentado é mais eficiente, inclusive para muitas das instâncias de ordem superior ou igual a 20. O tempo de processamento depende da máquina em que o teste é realizando, dificultando a comparação desse parâmetro.

Agradecimentos: Os autores agradecem ao CNPq pelo apoio a esta pesquisa através do Projeto ProTem-CC.

6- Bibliografia

[AQB99] Abreu, N.M.M., Querido, T.M., Boaventura-Netto, P.O., *REDINV-SA: a simulated annealing for the quadratic assignment problem*, RAIRO Operations Research, vol. 33 (1999), pp. 249-273.

[BKR97] Burkard, R.E., Karisch, S.E. & Rendl, F. (1997). *QAPLIB - A Quadratic Assignment Problem Library*. Journal of Global Optimization, 10**,** 391-403.
Acesso internet: http://www.imm.dk/~sk/qaplib/

[BR83] Burkard, R.E. e Rendl, F., *A termodynamically motivated simulation procedure for combinatorial optimization problems*, European Journal Operations Research, vol. 17 (1983), pp. 169-174.

[Co90] Connolly, D.T. (1990). *An improved annealing scheme for the QAP*, European Journal Operations Research, 46, 93-100.

[GP66] Gavett, J.W. & Plyter, N.V. (1966). *The optimal assignment of facilities to locations by branch and bound*. Operations Research, 14, 210-232.

[HLP52] Hardy, G.H., Littlewood, J.E. & Pólya, G. (1952), *Inequalities*, Cambridge University Press.

[KGV83] Kirkpatrick, S., Gelatt, C.D. e Vecchi, M.P., *Optimization by Simulated Annealing*, Science, vol. 20 (1983), pp. 671-680.

[MRRTT53] Metropolis, N., Rosenbluth, A., Rosenbluth, M., Teller, A. e Teller, E., *Equation of state calculations by fast computer machines*, Journal of Chemical Physics, vol. 21 (1953), pp. 1087-1092.

[Ran00] Rangel, M.C. (2000). *Contribuições algébricas para o problema quadrático de alocação*. Tese de D.Sc. Programa de Engenharia de Produção - COPPE/UFRJ, Brasil.