Produto & Produção, vol. 10, n. 1, p. 122-135, fev. 2009

Um GRASP para o Problema da Rotulação Cartográfica de Pontos: Novas Soluções

Gildásio Lecchi Cravo
lecchi@fsjb.edu.br
Faculdade de Aracruz – UNIARACRUZ

Glaydston Mattos Ribeiro
glaydstonribeiro@ceunes.ufes.br
Universidade Federal do Espírito Santo – UFES
Instituto Nacional de Pesquisas Espaciais – INPE

Dr. Luiz Antonio Nogueira Lorena
lorena@lac.inpe.br
Instituto Nacional de Pesquisas Espaciais – INPE

O Problema da Rotulação Cartográfica de Pontos (PRCP) é uma importante etapa no processo de geração de mapas em um sistema de informações geográficas e consiste em posicionar os rótulos dos pontos em posições que não ocasionam sobreposições. O PRCP é um problema da classe NP-difícil e por isso, várias abordagens foram propostas usando heurísticas/metaheurísticas para resolvê-lo no sentido de se obter soluções polinomiais e de boa qualidade. Seguindo essa idéia, esse trabalho propõe um GRASP para o PRCP baseado em seu grafo de conflitos. Os resultados encontrados para instâncias da literatura mostram que essa metaheurística é uma boa estratégia, pois a mesma produziu soluções de melhor qualidade que todos os resultados informados na literatura, em um tempo computacional razoável.

Palavras-chave: GRASP; Rotulação Cartográfica de Pontos; Heurística.

The point-feature cartographic label placement problem (PFCLP) is an important task in map generation process mainly in geographic information systems. It consists in placing point labels in clear and legible positions in a map or diagram. The PFCLP is a NP-Hard problem consequently in the literature, there are several approaches using heuristics/metaheuristics for producing good solutions in reduced times. Following this idea, in this paper we proposed a GRASP that uses the conflict graph produced by the PFCLP. Considering instances proposed in the literature, our results show that this metaheuristic is a good strategy. We had better solution than all those reported in the literature in reasonable computational times.

Keywords: GRASP; Map Labeling; Heuristic.

1 Introdução

O Problema da Rotulação Cartográfica de Pontos (PRCP) é uma tarefa importante na Cartografia e consiste em posicionar os rótulos dos pontos em posições pré-definidas de tal forma que não gerem sobreposições. De maneira geral, rótulos carregam informação sobre objetos (ou características) em exibições como gráficos, redes, diagramas, ou mapas cartográficos (WOLFF, 1999). Cada característica que precisa ser rotulada tem posições, nas quais, seu rótulo pode ser colocado. Porém, como já dito, é essencial que todos os rótulos sejam colocados de maneira clara e com pouca "poluição visual".

Existem três tipos de rotulação cartográfica: a de pontos (ex: cidades), de áreas (ex: país) e a rotulação de linhas (ex: rodovias). Esse trabalho considera somente a rotulação cartográfica de pontos por ter recebido maior atenção por parte dos pesquisadores, e por se tratar de um problema de difícil solução (WOLFF; STRIJK, 2005). A Figura 1 mostra um exemplo do PRCP. Note que existem vários rótulos sobrepostos, reduzindo assim a legibilidade do mapa.

Figura 1 - Um exemplo de um mapa com alguns rótulos sobrepostos (*Fonte: Ribeiro; Lorena, 2006*)

Na literatura existem diversas técnicas de solução que consideram heurísticas, metaheurísticas e métodos exatos. Muitas destas foram propostas porque o PRCP pode ser representado por um grafo de conflitos, em que os vértices representam as possíveis posições dos rótulos e as arestas, os possíveis conflitos (sobreposições). Seguindo também essa idéia, esse trabalho apresenta um GRASP (FEO; RESENDE, 1989) para o PRCP.

O GRASP (*Greedy Randomized Adaptive Search Procedure*) é um método iterativo que apresenta duas fases distintas. A primeira fase de construção cria uma solução gulosa, porém, aleatória. A segunda fase é uma busca local, aplicada sobre a solução aleatória e gulosa obtida pela primeira fase. Nesta fase, a vizinhança da solução atual é avaliada na busca de uma solução melhor. Essas duas fases são repetidas durante um certo número de iterações e a melhor solução obtida é retornada pelo GRASP.

O GRASP para o PRCP foi testado em instâncias propostas na literatura, produzindo melhores resultados que todos os apresentados e em tempos computacionais razoáveis.

O restante do trabalho está como segue. A próxima seção apresenta uma revisão sobre o PRCP, e a Seção 3 apresenta as principais abordagens para o PRCP. A Seção 4 apresenta uma breve revisão sobre o GRASP, assim como o GRASP proposto. A Seção 5 mostra os resultados computacionais sobre as instâncias da literatura, seguida das conclusões e sugestões para pesquisas futuras.

2 O Problema da Rotulação Cartográfica de Pontos

O PRCP pode ser visto como um problema de otimização combinatória, no qual deseja-se maximizar a legibilidade sujeito a restrições de sobreposições.

Cada ponto tem um conjunto de locais nos quais podem ser colocados os seus rótulos. Esses locais são conhecidos como posições candidatas e são definidos a partir de uma padronização cartográfica (CHRISTENSEN *et al*,1995). Nesta padronização, para cada posição candidata é associado um peso indicando a preferência cartográfica; quanto menor o peso, maior a preferência pela rotulação na posição indicada. A Figura 2 mostra um grupo de 8 posições candidatas para um ponto onde os números indicam as preferências cartográficas. A posição número 1 é a melhor.

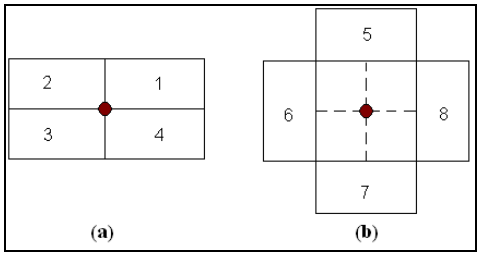

Figura 2 – Conjunto de oito posições candidatas para um ponto (*Fonte: CHRISTENSEN et al, 1995*)

A partir da definição de posições candidatas, o PRCP pode ser representado por um grafo de conflitos. Seja N o número de pontos a serem rotulados e P o número de posições candidatas para cada ponto. $G= \{V, A\}$ é o grafo de conflitos correspondente, onde $V= \{v_1, v_2, ..., v_{N*P}\}$ é o conjunto de posições candidatas e $A=\{(v_i, v_j): i, j \in V, i \neq j\}$ os conflitos entre posições candidatas.

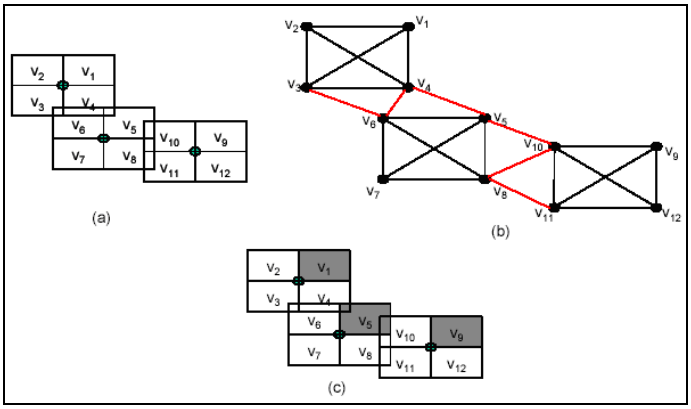

Figura 3 – Grafo de conflito para o PRCP. (a) Problema; (b) Grafo de conflitos e (c) Solução ótima (*Fonte: RIBEIRO; LORENA, 2006*)

A Figura 3(b) mostra o grafo de conflitos gerado a partir do problema da Figura 3(a), e a Figura 3(c) mostra a solução ótima para esse problema. Geralmente na literatura, a proporção de rótulos livres em relação ao número de pontos a serem rotulados, avalia a qualidade da rotulação (RIBEIRO; LORENA, 2005; YAMAMOTO *et al*, 2002; CHRISTENSEN *et al*, 1995). Assim, considerando o problema mostrado na Figura 3, a solução fornecida na Figura 3(c) tem 100% de rótulos livres.

3 Principais Abordagens Para o PRCP e Técnica de Redução do Grafo de Conflitos

Para o PRCP com um conjunto discreto de posições candidatas para os pontos, como descrito na Figura 2, existem três abordagens diferentes na literatura. A primeira busca o maior número de rótulos livres, mesmo não rotulando todos os pontos. Esse problema pode ser visto como o Problema do Máximo Conjunto Independente (PMCI) (ZORASTER, 1990; STRIJK *et al*, 2000). A segunda também procura o máximo número de rótulos livres, porém, neste caso, todos os pontos devem ser

rotulados. Esse problema é conhecido como Problema do Máximo Número de Rótulos Sem Conflitos (PMNRSC) (RIBEIRO; LORENA, 2005) e vários trabalhos usaram essa abordagem (CHRISTENSEN *et al*, 1994; 1995; VERNER *et al*, 1997; YAMAMOTO; LORENA, 2005). A última abordagem considera o Problema da Minimização do Número de Conflitos (PMNC) e foi introduzida primeiramente por Ribeiro e Lorena (2005; 2006). Essa aproximação também rotula todos os pontos, mas minimiza o número de conflitos na solução.

Considerando o PRCP como um PMCI, existem pesquisas significativas na literatura mostrando algoritmos e técnicas. Zoraster (1990) formulou matematicamente o PRCP, com restrições de conflitos e posições candidatas extras de custo alto, que são utilizadas quando os pontos não podem ser rotulados sem conflitos. Strijk *et al* (2000) propuseram uma modelagem de programação inteira binária que usa restrições de corte para reduzir o número de restrições de conflito presentes no modelo. Os autores utilizaram uma relaxação de programação linear e aplicaram um algoritmo *Branch and Bound* para encontrar as soluções ótimas para os problemas por eles testados, e ainda aplicaram e propuseram várias heurísticas, sendo elas: *Simulated Annealing,* Busca em Vizinhança Diversificada, *k-Opt* e Busca Tabu.

Agora, considerando o PRCP como um PMNRSC, existem vários trabalhos na literatura. Christensen *et al* (1994; 1995) propuseram um método denominado Busca Exaustiva, que faz uma procura por soluções melhores alternando posições de rótulos previamente posicionados. Christensen *et al* (1995) também propuseram um algoritmo guloso com sucessivas otimizações locais e um algoritmo denominado *Discrete Gradient Descent* que considera posições alternativas dos rótulos, porém, esse algoritmo tem dificuldades para escapar de máximos locais. Verner *et al* (1997) aplicaram um Algoritmo Genético com máscara tal que se um rótulo se encontra em conflito, é permitida a troca por outra posição candidata através de cruzamento e com isso, procura-se eliminar o conflito. Yamamoto *et al* (2002) propuseram um algoritmo de Busca Tabu que obteve bons resultados quando comparados com a literatura. Schreyer e Raidl (2002) aplicaram um sistema de Colônia de Formigas, mas os resultados não foram interessantes quando comparados aos obtidos por Yamamoto *et al* (2002). Yamamoto e Lorena (2005) desenvolveram um algoritmo exato para instâncias pequenas do PRCP e aplicaram um Algoritmo Genético Construtivo (CGA) em instâncias de grande porte. O algoritmo exato foi aplicado a instâncias de 25 pontos e o CGA foi aplicado a instâncias de até 1000 pontos. Recentemente, Yamamoto *et al* (2005) propuseram um algoritmo denominado FALP que mostrou resultados melhores que o CGA usando a mesmas instâncias propostas por Yamamoto *et al* (2002). Considerando por último o PRCP como um PMNC, Ribeiro e Lorena (2005; 2006) propuseram dois modelos matemáticos baseados em programação linear inteira binária e também uma Relaxação Lagrangeana com divisão em clusters (LagClus) que apresentou os melhores resultados da literatura para as instâncias propostas por Yamomoto *et al* (2002). A segunda formulação proposta pelos autores é apresentada abaixo.

$$v(PMNC) = Min \sum_{i=1}^{N} \sum_{j=1}^{P_i} \left(w_{i,j} x_{i,j} + \sum_{c \in C_{i,j}} y_{i,j,c} \right) \tag{1}$$

Sujeito a:

$$\sum_{j=1}^{P_i} x_{i,j} = 1 \quad \forall\, i = 1...N \tag{2}$$

$$\left| C_{i,j} \right| x_{i,j} + \sum_{(k,t) \in S_{i,j}} x_{k,t} - \sum_{c \in C_{i,j}} y_{i,j,c} \le \left| C_{i,j} \right| \quad \forall\, i = 1...N \quad \forall\, j = 1...P_i \tag{3}$$

$$x_{i,j}, x_{k,t} \text{ e } y_{i,j,c} \in \{0,1\} \quad \forall\, i = 1...N \quad \forall\, j = 1...P_i \quad c \in C_{i,j} \qquad (4)$$

Sendo:

- N o número de pontos a serem rotulados e P_i o conjunto de posições candidatas do ponto i;
- $x_{i,j}$ uma variável binária com $i \in N$ e $j \in P_i$;
- $w_{i,j}$ a preferência cartográfica associada a cada posição candidata. Permite priorizar algumas posições candidatas como descrito na Figura 2;
- $S_{i,j}$ um conjunto de par de índices (k,t): $k>i$ de posições candidatas tal que $x_{k,t}$ tem conflito (aresta) com $x_{i,j}$;
- $C_{i,j}$ um conjunto com todos os pontos que contêm posições candidatas em conflito com a posição candidata $x_{i,j}$;
- $y_{i,jc}$ uma variável de conflito entre a posição candidata $x_{i,j}$ e o ponto $c \in C_{i,j}$: $c>i$.

A Restrição (2) garante que cada ponto deve ser rotulado com uma posição candidata. A Restrição (3) garante que, se vértices com conflitos forem escolhidos para compor a solução, a função objetivo descrita na Equação (1) será penalizada. E a Restrição (4) indica que todas as variáveis no modelo são binárias.

O grafo de conflitos do PRCP pode ser muito grande, dificultando o processo de resolução. Wagner *et al* (2001) apresentaram um método para reduzir o grafo de conflitos obtido para um PRCP. O método é composto de três regras que não alteram o conjunto de soluções ótimas, e estão mostradas a seguir:

1. Se um ponto p tem uma posição candidata livre de conflitos, deve-se usar essa posição candidata na solução do problema e todas as outras posições candidatas do ponto p devem ser eliminadas;
2. Se um ponto p tem uma posição candidata p_i que está em conflito somente com uma posição candidata q_k, e o ponto q tem uma posição candidata q_j ($j \neq k$) que está sobreposta somente por uma posição candidata p_l ($l \neq i$), então adicione p_i e q_j na solução do problema e elimine as demais posições candidatas de p e q;
3. Se um ponto p tem somente uma posição candidata p_i restante, e todas as posições candidatas que sobrepõem p_i formam uma clique, então adicione p_i na solução do problema e elimine todas as posições candidatas que sobrepõem p_i.

As regras são aplicadas exaustivamente, ou seja, após a eliminação de uma posição candidata p_i, deve-se aplicar recursivamente as regras na vizinhança de p_i. Essa redução apresenta bons resultados e, para mais detalhes, ver Wagner *et al* (2001).

4 Um GRASP para o PRCP

O GRASP, proposto por Feo e Resende (1989; 1995), é um método iterativo constituído de duas fases: a de construção e a de busca local. Na fase de construção, é gerada uma lista de candidatos, ordenados de acordo com sua contribuição na função objetivo. Uma solução é então construída elemento a elemento. Essa construção é probabilística, pois a escolha do novo elemento que deverá compor a solução é feita aleatoriamente a partir de uma lista, denominada lista de candidatos restritos (LCR) que é formada pelo melhores elementos da lista de candidatos. A heurística também é adaptativa, pois a cada iteração da fase de construção, os custos (pesos utilizados pela função de ordenação) são atualizados para refletir as mudanças ocasionadas pela seleção do elemento na iteração anterior. Tendo em vista que essa construção é probabilística, as soluções geradas nesta fase provavelmente não serão localmente ótimas. Daí a importância da segunda fase do GRASP, que tenta melhorar a solução construída na fase anterior, trabalhando na sua vizinhança.

O parâmetro de aleatoriedade que determina o tamanho da LCR, ou seja, quantos melhores elementos farão parte da LCR, é um parâmetro importante a ser ajustado em um procedimento GRASP. Devido a sua facilidade implícita, existem na literatura diversas aplicações práticas usando o GRASP. Desde 1980, o GRASP vem sendo aplicado em grande escala na pesquisa de operacional e em problemas de otimização na indústria (FESTA; RESENDE, 2002). Desses incluem problemas em roteamento, lógica, particionamento, localização, teoria dos grafos, transporte, telecomunicações, projeto VLSI, etc.

Resende e Feo (1996) apresentaram quatro implementações do GRASP para resolver o problema SAT, demonstrando assim, como diferentes versões do GRASP podem ser conseguidas mudando a função gulosa da fase de construção e a busca local.

Algumas variações do GRASP também foram propostas. Prais e Ribeiro (2000) propuseram um GRASP Reativo, que é um GRASP no qual não se usa um valor fixo para o parâmetro que define o tamanho da LCR durante a fase construtiva. O GRASP Reativo previamente se auto-ajusta de acordo com a qualidade das soluções encontrada. Laguna e Martí (1999) incorporaram ao GRASP uma estratégia de *path relinking*, para tentar melhorar os resultados. Recentemente, Resende e Ribeiro (2005) apresentaram vários avanços e aplicações para o GRASP com *path relinking*. Para uma revisão completa sobre o GRASP e suas aplicações, veja Festa e Resende (2002), Aiex *et al* (2003) e Resende e Ribeiro (2003).

4.1 Fase Construtiva do GRASP para o PRCP

Observando que o PRCP pode ser representado por um grafo de conflitos, o GRASP proposto usa um método que é baseado no grau dos vértices, como determinado em alguns trabalhos da literatura, como, por exemplo, em Abello *et al* (1999) e Feo *et al* (1994). O grau de um vértice é o número de arestas incidentes naquele vértice, representando uma medida de conflitos de um rótulo.
Seja $G=\{V,A\}$ um grafo de conflitos como descrito no Capitulo 2, sendo V um conjunto de vértices (posições candidatas) e A um conjunto de arestas (conflitos entre vértices), e *LCR_Tamanho* o tamanho da lista de candidatos restrita. Assim, o pseudocódigo da fase construtiva pode ser escrito como mostrado na Figura 4.

```
Procedimento Contrucao_Aleatoria_e_Gulosa()
1 Sol ← {}
2 V_Guloso ← V
3 Enquanto V_Guloso ≠ {} Faça
4  CriarLCR(LCR,V_Guloso,Sol,LCR_Tamanho)
5  v ← Seleção_Aleatoria_do_Elemento (LCR)
6  Sol ← Sol ∪ v
7  Reduzir_Vértices_Ativos(V_Guloso, v)
   Fim Enquanto
8 Retorne Sol
```

Figura 4 – Uma heurística construtiva para o PRCP

O método *CriarLCR* é responsável por criar a lista de candidatos restrita. Primeiro, considerando o grafo de conflitos atual, esse método computa um peso para cada vértice baseado no seu grau, preferências cartográficas e potenciais conflitos com a solução atual *Sol*, e em seguida ordena os vértices de forma crescente segundo seus pesos. Em seguida, os primeiros *LCR_Tamanho* vértices são considerados como restritos. Essa lista de candidatos restrita é criada a cada iteração porque a cada repetição do procedimento mostrado na Figura 4, o grafo de conflitos é reduzido. Quando o procedimento seleciona um vértice para entrar na solução atual (linha 5 do algoritmo), esse vértice é uma posição candidata de algum ponto *i* e deve ser aplicado um método para reduzir o grafo atual e a lista de vértices ativos. Esse método, denominado *Reduzir_Vértices_Ativos*, recebe a lista de vértices

atual (*V_Guloso*) e o vértice selecionado *v*, em seguida remove de *V_ Guloso* todas as posições candidatas do ponto *i* que tem *v* como uma posição candidata.

O grafo restante pode ainda apresentar alguns vértices em potencial conflito com a solução atual *Sol*. Com isso, para evitar selecionar vértices em conflitos com a solução atual, a lista de candidatos restrita deve estar de acordo com a preferência cartográfica, o grau e o potencial conflito de cada vértice ativo com a solução atual, ou seja, conforme a seguinte equação:

$$Peso\ (x) = Preferencia\_Cartografica(x) + Grau(x) + M*PotenciaisConflitos(Sol,x) \qquad (5)$$

Onde:

- *Preferência_Cartografica (x)* é a preferência cartográfica do vértice *x*;
- *Grau(x)* - O grau do vértice *x* considerando o grafo de conflitos atual;
- *PotenciaisConflitos(Sol, x)* - O número de potenciais conflitos entre *x* e a solução atual *Sol*; e
- *M* é um coeficiente que penaliza o vértice *x* se este apresentar potenciais conflitos com a solução atual *Sol*.

A função representada pela Equação (5) assegura que se um vértice *x* tem grau alto e vários conflitos potenciais com a solução atual *Sol*, *x* aparecerá no final da lista quando os vértices forem ordenados, evitando por enquanto conflitos.

A Figura 5 mostra duas iterações de um exemplo teórico, no qual, todas as preferências cartográficas são iguais a 1 para todas as posições candidatas (vértices). Para esse exemplo, *LCR_Tamanho = 4* e *M =10* na Equação (5). Note que na primeira iteração, representada na Figura 5(a), o vértice selecionado aleatoriamente foi v_1 e, conseqüentemente, todas as posições candidatas do respectivo ponto foram removidas do grafo de conflitos. Na segunda iteração (veja Figura 5(b)), o grafo de conflitos está reduzido e todos os graus dos vértices são recalculados. O vértice v_{13} tem um potencial conflito com o vértice v_1 selecionado na iteração anterior. Com isso, esse vértice é penalizado e aparece no final da lista ordenada, reduzindo assim suas chances de ser selecionado nesta iteração.

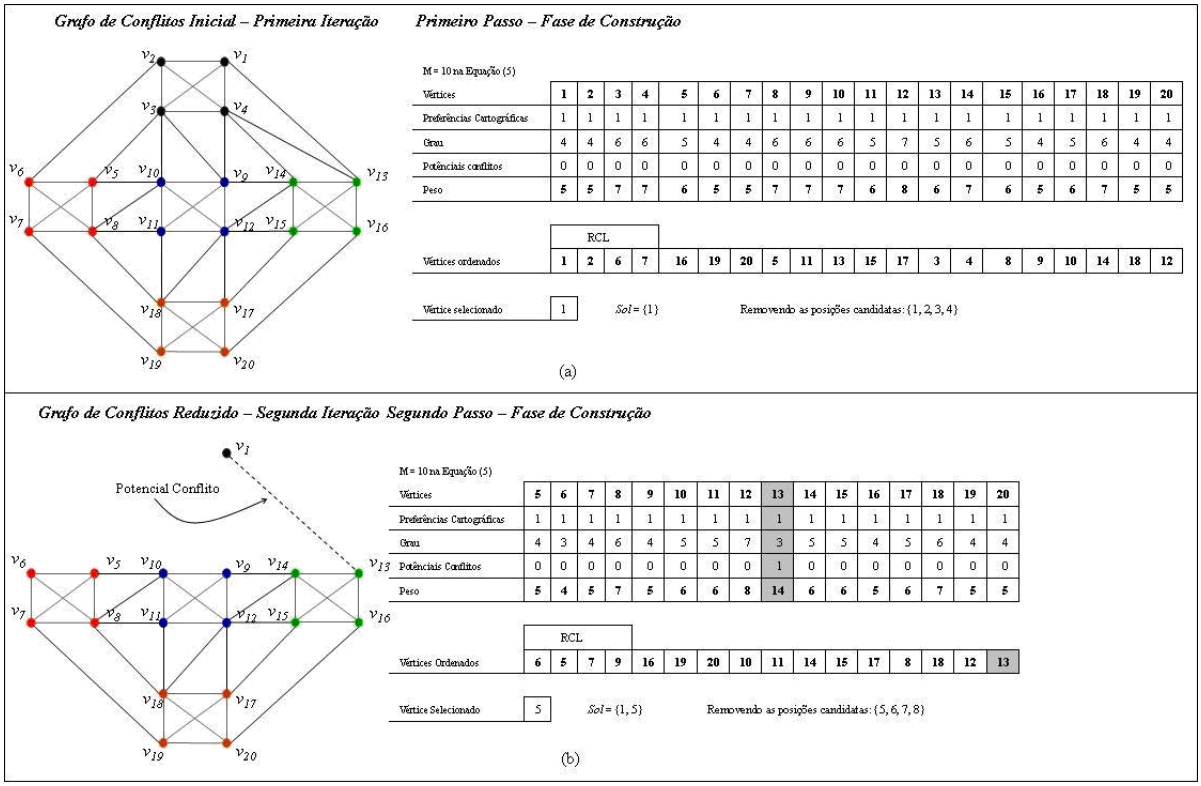

Figura 5 – Um exemplo da heurística construtiva para o PRCP

Este processo é repetido até que uma solução viável seja obtida. Se o comprimento da LCR em alguma iteração for menor que o número restante de vértices, o tamanho da LCR deve ser igual a este número.

4.2 Fase de Busca Local do GRASP para o PRCP

A segunda fase do GRASP para o PRCP usa uma heurística de busca local que trabalha alterando a posição candidata obtida para cada ponto *i* na primeira fase, por uma outra posição válida $j \in P_i$ procurando obter uma solução que reduz a função objetiva descrita na Equação (1). Essa heurística de busca local está descrita na Figura 6.

Note que, conforme a Figura 6, a solução gerada através da heurística de construção é modificada (linha 11) e o valor obtido é comparado com a melhor solução obtida até então (*fCorrente*). Note também que é calculado o total de rótulos livres (linha 17) à medida que uma nova solução é encontrada. Com isso, o GRASP para o PRCP pode ser representado através da Figura 7. Observe na linha 6 desse algoritmo que se a melhor solução encontrada (*f**) for igual ao valor da função objetivo de uma solução *x* gerada pela busca local, deve-se comparar qual das duas soluções (*x** ou *x*) apresenta o maior número de rótulos livres.

```
                Procedimento Busca_Local(x)
// Seja:
// - FO(Sol) a função objetiva descrita na Equação (1)
// - NumeroRotulosSemConflitos (Sol) uma função que conta
o //   número de rótulos sem conflitos presentes na
solução Sol
1  Sol ← x
2  EncontrarNovaSolução ← verdadeiro
3  fCorrente ← FO (Sol)
4  RotulosLivres ← NumeroRotulosSemConflitos (Sol)
5  Enquanto EncontrarNovaSolução Faça
6    EncontrarNovaSolução ← falso
7    Para i=1,…, N Faça
8      PosiçãoCandidataCorrente = Sol[i]
9      Para ∀j∈P_i Faça
10       Se j==Sol[i] Então Continue
11         Sol[i] ← j
12         Se (FO(Sol)<fCorrente) Então
13           fCorrente ← FO(Sol)
14           EncontrarNovaSolução ← verdadeiro
15           MelhorVizinho ← j
16            PontoAlterado ← i
17           RotulosLivres ← NumeroRotulosSemConflitos
(Sol)
           Fim Se
       Fim Para
18     Sol[i] ← PosiçãoCandidataCorrente
     Fim Para
19   Se EncontrarNovaSolução Então
20     Sol[PontoAlterado] ← MelhorVizinho
     Fim Se
   Fim Enquanto
21 x ← Sol
22 Retorne (FO(Sol) e RótulosLivres)
```

Figura 6 – Heurística de Busca Local

Como pode ser visto, o GRASP é dependente do número de iterações e do tamanho da lista de candidatos restrita. Assim, para definir os melhores valores para estes parâmetros, esse GRASP foi aplicado em um conjunto padrão de instâncias gerado e proposto por Yamamoto *et al* (2002) e disponível em *http://www.lac.inpe.br/~lorena/instancias.html*.

```
Dados : Número de iteracões i_max
                Resultado   : Solucão x* ∈ X
1 f* ← ∞
2 Rótulos_Livres* ← ∞
3 Para i=1,...,i_max Faça
4  x ← Contrucao_Aleatoria_e_Gulosa()
5  [f(x), Rótulos_Livres] ← Busca_Local(x)
6  Se (f(x) < f*) OU (f(x)= f* E Rótulos_Livres >
Rótulos_Livres*) Então
7    f* ← f(x)
8    x* ← x
9    Rotulos_Livres* ← Rótulos_Livres
10  Fim Se
11 Fim Para
12 Retorne (f*, Rótulos_Livres* e x*)
```

Figura 7 – Algoritmo principal do GRASP proposto

5 Resultados Computacionais

O GRASP proposto foi implementado em C++ e os testes foram executados em um computador com um processador Pentium IV 2.80 GHz e 512 MB de memória. As instâncias usadas foram as propostas por Yamamoto e Lorena (2002), e são compostas de 25 (vinte e cinco) testes para cada número de pontos *N*. Como feito por Zoraster (1990), Yamamoto *et al* (2002), Yamamoto e Lorena (2005) e Ribeiro e Lorena (2005), para todos os problemas não foram consideradas as preferências cartográficas, sendo fixadas em *1* todas as posições candidatas e sendo igual a *4* o total dessas posições para cada ponto. Esse procedimento permite comparar os resultados encontrados com os da literatura. Após vários experimentos, o número máximo de iterações (i_{max}) foi fixado em 100. Com um número maior, os resultados não melhoram como esperado. Porém, o tamanho da lista de candidatos restrita influenciou nos resultados significativamente, mas novamente com listas maiores que 10 todos os experimentos não mostraram nenhuma melhoria, sendo assim, os testes foram restringidos a valores entre 2 e 10 para a LCR, com 100 iterações.

As Tabelas de 1 a 4 relatam os principais resultados médios encontrados variando o tamanho da LCR para 10 execuções do GRASP. Nestas tabelas, a primeira coluna indica o número de pontos a serem rotulados. Para cada execução do GRASP foi calculado, para cada conjunto de instâncias, o valor médio gerado. Para avaliar a robustez do GRASP, as tabelas apresentam resultados médios para 10 execuções do GRASP. Assim, a segunda coluna mostra os resultados médios de 10 execuções para o número de arestas restantes (conflitos), para o número de rótulos em conflitos e para as porcentagens de rótulos livres, calculados como *((N – Numero de Rótulos em Conflitos) / N) *100*. A quarta coluna apresenta os tempos computacionais médios. Da sexta à nona, são apresentadas as melhores soluções fornecidas pelo GRASP entre as 10 execuções.

Percebe-se que os melhores resultados foram encontrados quando foram consideradas listas de candidatos restritas com 5 e 6 elementos. Considerando assim somente as Tabelas 2 e 3, o tempo computacional para os problemas mais difíceis com 750 e 1000 pontos, é de aproximadamente 40 e 112 segundos, respectivamente.

Tabela 1 – Média dos resultados obtidos pelo GRASP com *RCL_Tamanho* = 2

Número de Pontos	Média Geral de 10 execuções				Melhor média das 10 execuções			
	Número de arestas	Número de rótulos livres	Proporção de rótulos livres (%)	Tempo (s)	Número de arestas	Número de rótulos livres	Proporção de rótulos livres (%)	Tempo (s)
25	3,00	5,13	79,50	0,023	3,00	5,13	79,50	0,023
100	0,00	0,00	100	0,003	0,00	0,00	100	0,003
250	0,00	0,00	100	0,029	0,00	0,00	100	0,028
500	0,96	1,80	99,64	8,203	0,96	1,80	99,64	8,199
750	9,84	18,88	97,48	40,410	9,84	18,88	97,48	40,386
1000	43,12	80,80	91,92	112,522	43,12	80,80	91,92	112,512

Tabela 2 – Média dos resultados obtidos pelo GRASP com *RCL_Tamanho* = 5

Número de Pontos	Média Geral de 10 execuções				Melhor média das 10 execuções			
	Número de arestas	Número de rótulos livres	Proporção de rótulos livres (%)	Tempo (s)	Número de arestas	Número de rótulos livres	Proporção de rótulos livres (%)	Tempo (s)
25	2,75	4,66	81,35	0,023	2,75	4,62	81,50	0,022
100	0,00	0,00	100,00	0,004	0,00	0,00	100,00	0,002
250	0,00	0,00	100,00	0,038	0,00	0,00	100,00	0,034
500	0,85	1,66	99,67	8,119	0,840	1,64	99,67	7,757
750	9,39	17,86	97,62	40,364	9,24	17,84	97,62	40,370
1000	42,98	81,43	91,86	116,856	42,40	81,08	91,89	112,49

Tabela 3 – Média dos resultados obtidos pelo GRASP com *RCL_Tamanho* = 6

Número de Pontos	Média Geral de 10 execuções				Melhor média das 10 execuções			
	Número de arestas	Número de rótulos livres	Proporção de rótulos livres (%)	Tempo (s)	Número de arestas	Número de rótulos livres	Proporção de rótulos livres (%)	Tempo (s)
25	2,79	4,80	80,80	0,026	2,75	4,62	81,50	0,025
100	0,00	0,00	100,00	0,004	0,00	0,00	100,00	0,002
250	0,00	0,00	100,00	0,049	0,00	0,00	100,00	0,036
500	0.848	1,66	99,67	9,055	0,84	1,64	99,67	8,268
750	9,36	17,84	97,62	42,25	9,20	17,68	97,64	40,363
1000	42,84	80,99	91,90	112,46	42,28	79,56	92,04	112,457

Tabela 4 – Média dos resultados obtidos pelo GRASP com *RCL_Tamanho* = 10

Número de Pontos	Média Geral de 10 execuções				Melhor média das 10 execuções			
	Número de arestas	Número de rótulos livres	Proporção de rótulos livres (%)	Tempo (s)	Número de arestas	Número de rótulos livres	Proporção de rótulos livres (%)	Tempo (s)
25	2,80	4,70	81,20	0,026	2,75	4,62	81,50	0,0215
100	0,00	0,00	100,00	0,003	0,00	0,00	100,00	0,002
250	0,00	0,00	100,00	0,052	0,00	0,00	100,00	0,034
500	0,90	1,72	99,66	8,831	0,88	1,60	99,68	8,149
750	9,87	18,92	97,48	40,397	9,72	18,68	97,51	40,401
1000	43,44	81,78	91,82	116,117	42,76	81,08	91,89	116,556

Tentando reduzir os tempos computacionais, a técnica proposta por Wagner *et al* (2001) foi aplicada para reduzir os grafos de conflitos. Os resultados do GRASP com a redução são mostrados nas Tabelas de 5 a 8, que possuem a mesma estrutura das Tabelas 1, 2, 3 e 4.

Tabela 5 – Média dos resultados obtidos pelo GRASP com *RCL_Tamanho* = 2 usando a redução proposta por Wagner *et al* (2001)

Número de Pontos	Média Geral de 10 execuções				Melhor média das 10 execuções			
	Número de arestas	Número de rótulos livres	Proporção de rótulos livres (%)	Tempo (s)	Número de arestas	Número de rótulos livres	Proporção de rótulos livres (%)	Tempo (s)
25	3,00	5,63	77,50	0,016	3,00	5,63	77,50	0,011
100	0,00	0,00	100,00	0,000	0,00	0,00	100,00	0,000
250	0,00	0,00	100,00	0,003	0,00	0,00	100,00	0,001
500	0,92	1,76	99,65	0,337	0,92	1,76	99,65	0,335
750	9,96	19,04	97,46	5,210	9,96	19,04	97,46	5,207
1000	43,12	81,28	91,87	26,823	43,12	81,28	91,87	26,673

Tabela 6 – Média dos resultados obtidos pelo GRASP com *RCL_Tamanho* = 5 usando a redução proposta por Wagner *et al* (2001)

Número de Pontos	Média Geral de 10 execuções				Melhor média das 10 execuções			
	Número de arestas	Número de rótulos livres	Proporção de rótulos livres (%)	Tempo (s)	Número de arestas	Número de rótulos livres	Proporção de rótulos livres (%)	Tempo (s)
25	2,75	4,68	81,25	0,018	2,75	4,625	81,50	0,015
100	0,00	0,00	100,00	0,000	0,00	0,00	100,00	0,000
250	0,00	0,00	100,00	0,002	0,00	0,00	100,00	0,000
500	0,85	1,65	99,67	0,362	0,84	1,64	99,67	0,346
750	9,13	17,28	97,70	5,214	9,04	16,96	97,74	5,216
1000	42.09	79,82	92,02	26,82	41,92	79,28	92,07	26,718

Tabela 7 – Média dos resultados obtidos pelo GRASP com *RCL_Tamanho* = 6 usando a redução proposta por Wagner *et al* (2001)

Número de Pontos	Média Geral de 10 execuções				Melhor média das 10 execuções			
	Número de arestas	Número de rótulos livres	Proporção de rótulos livres (%)	Tempo (s)	Número de arestas	Número de rótulos livres	Proporção de rótulos livres (%)	Tempo (s)
25	2,78	4,625	81,50	0,016	2,75	4,625	81,50	0,014
100	0,00	0,00	100,00	0,000	0,00	0,00	100,00	0,000
250	0,00	0,00	100,00	0,000	0,00	0,00	100,00	0,000
500	0,84	1,656	99,67	0,371	0,84	1,64	99,67	0,345
750	9,14	17,35	97,69	5,220	9,00	17,12	97,72	5,216
1000	41,88	79,388	92,06	26,788	41,60	77,96	92,20	26,688

Tabela 8 – Média dos resultados obtidos pelo GRASP com *RCL_Tamanho* = 10 usando a redução proposta por Wagner *et al* (2001)

Número de Pontos	Média Geral de 10 execuções				Melhor média das 10 execuções			
	Número de arestas	Número de rótulos livres	Proporção de rótulos livres (%)	Tempo (s)	Número de arestas	Número de rótulos livres	Proporção de rótulos livres (%)	Tempo (s)

25	2,75	4,79	80,85	0,017	2,75	4,75	81,00	0,016
100	0,00	0,00	100,00	0,000	0,00	0,00	100,00	0,000
250	0,00	0,00	100,00	0,000	0,00	0,00	100,00	0,000
500	0,86	1,68	99,66	0,376	0,84	1,64	99,67	0,347
750	9,42	18,03	97,60	5,230	9,40	17,88	97,62	5,217
1000	42,58	80,42	91,96	26,796	42,20	79,68	92,03	26,691

Novamente os melhores resultados são encontrados quando *LCR_Tamanho* assume valores entre 5 e 6. Com a redução, a qualidade das soluções foi melhorada e o tempo computacional reduziu. Considerando as Tabelas 6 e 7, para instâncias com 750 e 1000 pontos, os tempos de computacionais são de aproximadamente 5 e 26 segundos, respectivamente.

Considerando as Tabelas de 5 a 8 e as colunas que mostram os resultados médios para rótulos sem conflitos, para *LCR_Tamanho* = 5, o GRASP produziu melhores resultados para as instâncias com 750 pontos e para *LCR_Tamanho* = 6, melhorou as instâncias com 25 e 1000 pontos. Para as demais instâncias, os resultados são similares.

Tabela 9 – Comparando os resultados do GRASP com a literatura

Algoritmos	Proporção de Rótulos Livres				
	Problemas				
	100	250	500	750	1000
$GRASP_{Melhor,\ RCL\ \ Tamanho\ =\ 6}$	100,00	100,00	99,67	97,72	92,20
$GRASP_{Médio,\ RCL\ \ Tamanho\ =\ 6}$	100,00	100,00	99,67	97,69	92,06
$GRASP_{Melhor,\ RCL\ \ Tamanho\ =\ 5}$	100,00	100,00	99,67	97,70	92,02
$GRASP_{Médio,\ RCL\ \ Tamanho\ =\ 5}$	100,00	100,00	99,67	97,74	92,07
LagClus (Ribeiro e Lorena, 2006)	100,00	100,00	99,67	97,65	91,42
CGA_{Melhor} (Yamamoto and Lorena, 2005)	100,00	100,00	99,60	97,10	90,70
FALP (Yamamoto *et al*, 2005)	100,00	100,00	99,50	96,70	90,12
$CGA_{Médio}$ (Yamamoto e Lorena, 2005)	100,00	100,00	99,60	96,80	90,40
Busca Tabu (Yamamoto *et al*, 2002)	100,00	100,00	99,30	96,80	90,00
GA com máscara (Verner *et al*, 1997)	100,00	99,98	98,79	95,99	88,96
GA (Verner *et al*., 1997)	100,00	98,40	92,59	82,38	65,70
Simulated Annealing (Christensen *et al*, 1995)	100,00	99,90	98,30	92,30	82,09
Zoraster (Zoraster, 1990)	100,00	99,79	96,21	79,78	53,06
3-opt Gradient Descent (Christensen *et al*, 1995)	100,00	99,76	97,34	89,44	77,83
2-opt Gradient Descent (Christensen *et al*, 1995)	100,00	99,36	95,62	85,60	73,37
Gradient Descent (Christensen *et al*, 1995)	98,64	95,47	86,46	72,40	58,29
Algoritmo Guloso (Christensen *et al*, 1995)	95,12	88,82	75,15	58,57	43,41

Comparando-se agora os melhores resultados encontrados neste trabalho com a literatura, tem-se a Tabela 9. Pode-se observar que GRASP proposto produz soluções com qualidade superior a todos os outros algoritmos. Além disso, cabe ressaltar que existem algoritmos com abordagens diferentes sendo comparados, conforme Seção 2.

6 Conclusões e Recomendações

Este trabalho apresentou um GRASP para o problema da rotulação cartográfica de pontos. Como mostrado, o mesmo apresentou soluções de boa qualidade em um tempo computacional baixo. Sendo assim, acredita-se que esse algoritmo pode ser usado embutido em aplicações comerciais usadas pelos

cartógrafos devido a sua facilidade implícita ou, dependendo talvez do propósito, em aplicações que produzem mapas de tela para Sistemas de Informações Geográficas e aplicações *Web*.
Sob o ponto de vista de otimização, o GRASP proposto apresentou resultados melhores que todos os informados na literatura, principalmente quando se utiliza uma técnica de redução do grafo de conflitos. Os resultados do GRASP foram melhores que metaheurísticas conhecidas e produziu melhores resultados que a LagClus de Ribeiro e Lorena (2006).

Como pesquisas adicionais, acredita-se que esse GRASP com *path relinking* durante as suas iterações ou em uma fase de pós-otimização, pode melhorar ainda mais a qualidade das soluções, porém é necessário um estudo de como aplicar o *path relinking* sem que produza um aumento nos tempos computacionais.

Agradecimentos

Ribeiro e Lorena agradecem ao Conselho Nacional de Desenvolvimento Científico e Tecnológico - CNPq pelo apoio financeiro dado ao desenvolvimento deste trabalho.

Referências

ABELLO, J.; PARDALOS, P. M.; RESENDE, M. G. C. *On maximum clique problems in very large graphs.* In: ABELLO, James., VITTER, Jeffrey. (org.). External memory algorithms and visualization. v. 50 of DIMACS Series on Discrete Mathematics and Theoretical Computer Science, 1999.

AIEX, R. M.; BINATO, S.; RESENDE, M. G. C. *Parallel GRASP with path-relinking for job shop scheduling.* Parallel Computing, v. 29, p. 393-430, 2003.

CHRISTENSEN, J.; MARKS, J.; SHIEBER, S. *Placing text labels on maps and diagrams*. Graphics GEMS IV. Academic Press Professional: San Diego, 1994. 10p.

CHRISTENSEN, J.; MARKS, J.; SHIEBER, S. *An empirical study of algorithms for point-feature label placement.* ACM Transactions on Graphics, v. 14, n. 3, p. 203-232, 1995.

FEO, T. A.; RESENDE, M. G. C. *A probabilistic heuristic for a computationally difficult set covering problem.* Operations Research Letters, v. 8, p. 67-71, 1989.

FEO, T. A.; RESENDE, M. G. C.; SMITH, S. H. *A Greedy Randomized Adaptive Search Procedure for Maximum Independent Set*. Operations Research, v. 42, p. 860-878, 1994.

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

FESTA, P.; RESENDE, M. G. C. *GRASP: An annotated bibliography*. In: RIBEIRO, Celso Carneiro, HANSEN, Pierre. (org.) Essays and Surveys on Metaheuristics. Kluwer Acadmic Publishers, 2002. p. 325-367.

LAGUNA, M.; MARTÍ, R. *GRASP and path relinking for 2-layer straight line crossing minimization*. INFORMS Journal on Computing, v. 11, p. 44-52, 1999.

PRAIS, M.; RIBEIRO, C. C. *Reactive GRASP: An application to a matrix decomposition problem in TDMA traffic assignment.* INFORMS Journal on Computing, v. 12, p. 164-176, 2000.

RESENDE, M. G. C.; FEO, T. A. *GRASP for satisfiability*. In: JOHNSON, David, TRICK, Michaela. (org.) Cliques, Coloring, and Satisfiability: Second DIMACS Implementation Challenge, v. 26, p.

499-520. DIMACS Series on Discrete Mathematics and Theoretical Computer Science, American Mathematical Society, 1996.

RESENDE, M. G. C.; RIBEIRO, C. C. *Greedy randomized adaptive search procedures*. In: GLOVER, Fred, KOCHENBERGER, Gary (org.) Handbook of Metaheuristics. Kluwer Academic Publishers, p. 219-249, 2003.

RESENDE, M. G. C.; RIBEIRO, C. C. *GRASP with path-relinking: recent advances and applications.* In: IBARAKI, Toshihide., NONOBE, Koji, YAGIURA, Mutsunori(org.) Metaheuristics: Progress as Real Problem Solvers. Kluwer Academic Publishers, p. 29-63, 2005.

RIBEIRO, G. M.; LORENA, L. A. N. *Heuristics for cartographic label placement problems*. Computers and GeoSciences, v. 32, n. 6, p. 739-748, 2005.

RIBEIRO, G. M.; LORENA, L. A. N. *Lagrangean relaxation with clusters for point-feature cartographic label placement problems*. Computers and Operations Research, 2006. To Appear.

SCHREYER, M.; RAIDL, G. R. *Letting ants labeling point features*. In: PROC. OF THE 2002 IEEE CONGRESS ON EVOLUTIONARY COMPUTATION AT THE IEEE WORLD CONGRESS ON COMPUTATIONAL INTELLIGENCE, p. 1564-1569, 2002.

STRIJK, T.; VERWEIJ, B.; AARDAL, K. *Algorithms for maximum independent set applied to map labeling*. Disponível em ftp://ftp.cs.uu.nl/pub/RUU/CStechreps/CS-2000/2000-22.ps.gz., 2000.

VERNER, O. V.; WAINWRIGHT, R. L.; SCHOENEFELD, D. A. *Placing text labels on maps and diagrams using genetic algorithms with masking.* INFORMS Journal on Computing, v. 9, n. 3, p. 266-275, 1997.

WAGNER, F.; WOLFF, A.; KAPOOR, V.; STRIJK, T. *Three rules suffice for good label placement*. Algorithmica, v. 30, p. 334-349, 2001.

WOLFF, A. *Automated label placement in theory and practice*. Tese de Doutorado. Fachbereich Mathematik und Informatik, Freie Universität. Berlin, pp. 153, 1999.

WOLFF, A.; STRIJK, T. *The map labeling bibliography*. URL analisada em 06 de Junho de 2005. http://i11www.ilkd.uni-karlsruhe.de/~awolff/map-labeling/bibliography/, 2005.

YAMAMOTO, M.; CAMARA, G.; LORENA, L. A. N. *Tabu search heuristic for point-feature cartographic label placement.* GeoInformatica and International Journal on Advances of Computer Science for Geographic Information Systems. Kluwer Academic Publisher, Netherlands, v. 6, n. 1, p. 77-90, 2002.

YAMAMOTO, M.; LORENA, L. A. N. *A constructive genetic approach to point-feature cartographic label placement.* In: IBARAKI, Toshihide., NONOBE, Koji, YAGIURA, Mutsunori (org.) Metaheuristics: Progress as Real Problem Solvers. Kluwer Academic Publishers, p. 285-300, 2005.

YAMAMOTO, M.; CAMARA, G.; LORENA, L. A. N. *Fast point-feature label placement for real time screen maps*. In: GEOINFO 2005 - VII Brazilian Symposium on GeoInformatics, 7., São Paulo, 2005. Anais… Disponível em http://www.lac.inpe.br/~lorena/missae/sbc_Missae.pdf., 2005.

ZORASTER, S. *The solution of large 0-1 integer programming problems encountered in automated cartography.* Operations Research, v. 38, n. 5, p. 752-759, 1990.