SPOLM 2009

ISSN 2175-6295 Rio de Janeiro- Brasil, 05 e 06 de agosto de 2009.

084/2009 - HEURÍSTICA ILS PARA O PROBLEMA DA ROTULAÇÃO CARTOGRÁFICA DE PONTOS.

Celso de Oliveira, Sebastián Alberto Urrutia, Thiago Ferreira de Noronha
Universidade Federal de Minas Gerais (UFMG)
Av. Antônio Carlos, 6627 - Pampulha - Belo Horizonte - MG CEP 31270-901
celso.olv@gmail.com, surrutia@dcc.ufmg.br, thiago.noronha@gmail.com

RESUMO

O problema de rotulação cartográfica de pontos (PRPC) consiste em posicionar rótulos nas entidades pontos de um mapa cartográfico proporcionando uma boa visualização, reduzindo ao máximo as sobreposições destes rótulos e respeitando as convenções cartográficas. A abordagem do PRCP como problema de Otimização Combinatória e o estudo do grafo de conflitos proporcionou um ambiente para o experimento da metaheurística ILS como proposta para o PRCP. Neste trabalho apresentamos uma solução dividida em duas etapas: Pré-processamento onde é realizado a redução do grafo de conflitos e uma heurística ILS onde a partir de uma solução inicial de qualidade é efetuado seu refinamento. Os resultados apurados pelo processamento das instâncias propostas pela literatura apresentam baixo custo computacional e qualidade comparável com os relatados até o presente.

Palavras-chave: Rotulação cartográfica, Heurística, Otimização Combinatória, ILS.

ABSTRACT

The Point-Feature Cartographic Label Placement Problem (PFCLP) it consists in placing point labels on map entity providing a cleam view and reduce the overlap for obtain a quality labeling placement. The approach of the PFCLP as a Combinatorial Optimization problem, and the conflict graph provided an environment for ILS metaheuristic experiments for the PFCLP. This work presents a solution in two stages: pre-processing and a heuristic ILS. The computacional results of the instances proposed in the literature present a better solutions and computacional performance than all those reported.

Keywords: Point-feature label placement. ILS algorithm. Conflict graph, Combinatorial Optimization.

1. Introdução:

O problema da rotulação cartográfica de pontos, PRCP, pode ser definido como o processo de posicionar rótulos ou identificadores de entidades pontos em mapas cartográficos, garantindo qualidade sob forma de estética e visibilidade. Para proporcionar clareza na identificação dos pontos cartográficos é necessário evitar as sobreposições dos rótulos respeitando as convenções cartográficas, Yamamoto (2005). A produção automatizada realizada por computador de material cartográfico com estas características, necessita ser em baixo custo computacional para atender a crescente demanda por sistemas de informações geográficas em ambiente WEB´s e sistemas on-line de geoprocessamento. Por estes motivos a rotulação cartográfica automatizada se tornou uma área importante de estudo, se estendendo e influenciando aplicações como análises de recursos naturais, transportes, comunicação planejamento urbano, etc.

Conforme descrito na literatura o problema de rotulação de entidades pontos como cidades, edificações, etc., consiste em posicionar um texto informativo em uma região delimitada, denominada posição candidata, respeitando a padronização conhecida de Christensen et al. Na Figura 1 as regiões de 1 a 8 representam as posições candidatas para o posicionamento do texto relativo ao ponto cartográfico, sendo quanto menor o número da posição maior é sua preferência. O ponto cartográfico pode possuir também 4 posições candidatas como será apresentado adiante.

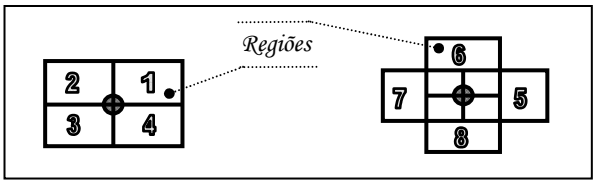

Figura 1 - Conjunto de 8 posições candidatas para a rotulação de 1 ponto cartográfico.

Devido ao adensamento de informações que são disponibilizadas nos mapas, há situações que existem conflitos ou sobreposições nas regiões dos rótulos, veja Figura 2.

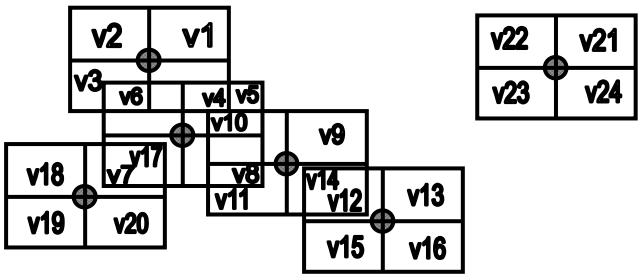

Figura 2 – Exemplo de um problema de sobreposição.

A Figura 2 representa uma situação hipotética onde há seis pontos cartográficos para o posicionamento de rótulos onde cada ponto possui *4* regiões ou posições candidatas. Neste contexto pode ser definido um grafo de conflitos $G = \{V, A\}$, sendo V o conjunto de vértices $V = \{v1, v2,\ldots, v(n*p)\}$, p o número de pontos cartográficos, n o número de posições candidatas por ponto e $A = \{(vi, vj) : vi, vj \in V, i \neq j\}$ as possíveis sobreposições entre as posições disponíveis Ribeiro e Lorena (2005). O grafo definido pelo PRCP é um grafo não orientado. Na Figura 3 é apresentado o exemplo das possíveis sobreposições nos vértices, onde é caracterizado pelas arestas do grafo de conflitos: { *(v3, v6), (v4, v5), (v4, v6), (v4, v10), (v7, v17), (v8, v10), (v8, v11) e (v12, v14)* }, sendo o grau de cada vértice o número de arestas de conflitos.

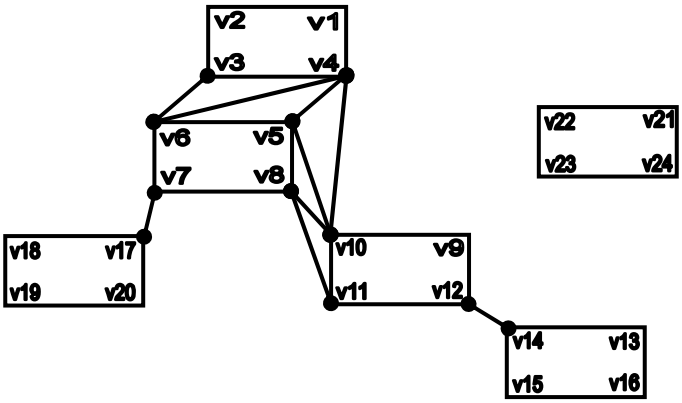

Figura 3 – Grafo de conflitos correspondente a Figura 2.

Devido às dimensões e características do grafo de conflitos, este problema caracteriza-se por ser de difícil solução sendo classificado como NP-difícil, Scheiber(1991). O estudo realizado sobre o PRCP sendo um problema de minimização de conflitos se apresenta como um ambiente de pesquisa para Otimização Combinatória onde as estratégias de abordagem heurísticas, metaheurísticas e métodos exatos podem ser estudadas e seus resultados avaliados quanto ao custo computacional e qualidade da solução final. Seguindo esta linha, apresentamos uma solução baseada no grafo de conflitos composta de duas etapas: Pré-processamento e Heurística baseada em ILS, (Iterated Local Search) Lourenço, et. Al (2002). Este trabalho se divide nas seguintes seções: Introdução, Revisão bibliográfica, Estratégia da solução, Resultados computacionais, Conclusão e Referências bibliográficas.

2. Revisão bibliográfica:

O PRCP pode ser tratado como um problema de Otimização Combinatória, consistindo em combinar as posições dos rótulos em posições candidatas, atendendo a um conjunto de restrições, maximizando ou minimizando uma função objetivo, Ribeiro e Lorena (2005). A abordagem do grafo de conflitos, tratando o problema como minimização do número de conflitos, foi proposto por Ribeiro & Lorena (2004) e posteriormente apresentaram nova formulação baseado nos pontos a serem rotulados, Ribeiro & Lorena (2005), na qual obteve uma redução do número de restrições. Cravo, Ribeiro & Lorena (2006), apresentaram um algoritmo guloso e um GRASP onde ambas propostas obtiveram resultados de qualidade. Cravo, Ribeiro & Lorena (2006), apresentaram também uma solução heurística composta de duas etapas onde um algoritmo guloso constrói uma solução inicial e um GRASP realiza o refinamento da solução, alcançando bons resultados para as instâncias propostas.

Na literatura encontramos as propostas de Christensen et al.(1993,1995) para os algoritmos Busca Exaustiva, Algoritmo Guloso, Discret Gradiente Descendent e Simulated Anneling. De Verner et al , (1997) Algoritmo genético e de Yamamoto (2002), temos Busca Tabu e Algoritmo genético construtivo, Yamamoto, Lorena (2005). Zoraster (1991) apresentou Relaxação Lagrangeana em problema de programação inteira 0-1.

3. Estratégia da solução.

A abordagem do PRCP como um problema de minimização de conflitos, proporcionou o estudo do PRCP como um problema de Otimização Combinatória. Apesar das formulações matemáticas, Ribeiro & Lorena (2006), e dos avanços ainda não foi obtido uma solução final para o problema. Diante disto nossa proposta é uma solução baseada na heurística ILS focada na estratégia de minimização de conflitos e rotulação de todos os pontos.

Sendo grafo de conflitos do PRCP esparso e podendo de acordo com o problema se tornar de grandes proporções, a proposta do pré-processamento para redução do grafo de conflitos vai ao encontro de reduzir o tamanho do problema fornecendo aos algoritmos de rotulação, problemas menores que tendem a ser mais fáceis de serem resolvidos. Seguindo nesta idéia percebemos também que após analise dos conflitos, estes conflitos tendiam a ocorrer nos vértices de maior grau, desta constatação surgiu à proposta de ordenação da lista de vértices em ordem crescente do grau. Este procedimento proporcionou a pesquisa de posições candidatas à possibilidade de encontrar uma resolução para os conflitos logo no início das listas.

A recursividade foi outro ponto importante como estratégia adotada. O algoritmo recursivo proporcionou ao pré-processamento, que a lista de vértices fosse processada uma única vez eliminando todos os pontos sem conflitos. Na construção da solução inicial e no refinamento, permitiu também que os conflitos fossem resolvidos pelo algoritmo de "BackTracking" através de buscas em profundidade. Esta estratégia associada à inserção de elementos adaptativos ao algoritmo do ILS permitiu sua adaptação às características das instâncias e nas características da solução em construção.

3.1 Pré-processamento.

O algoritmo de pré-processamento efetua a redução do grafo de conflitos baseado na técnica proposta por Wagner (2001), onde propôs à aplicação exaustiva das três regras aos vértices do grafo de conflitos, conforme descrito:

a) Dado um ponto p, se p_i é uma posição candidata de p sem conflitos, declare p_i como parte da solução e elimine todas as outras candidatas de p.
b) Dado dois pontos p e q, se p_i é uma posição candidata de p que esta em conflito com apenas um q_k de q e o ponto q possui a posição q_j $(j \neq k)$ que esta sobreposta somente por p_l $(l \neq i)$, então adicione p_i e q_j a solução e elimine todas as outras posições candidatas de p e q.
c) Dado um ponto p, se p possui apenas uma posição candidata p_i restante e as posições que sobrepõem p_i formam um clique, adicione p_i a solução e elimine todas as posições que sobrepõe p_i.

Somente a regra *"a"* foi tomada como base na proposta de redução do grafo de conflitos, sendo modificada e aplicada recursivamente a cada vértice do grafo, de tal forma que ao eliminar as posições candidatas p_i de um ponto cartográfico p, foram eliminadas também as arestas do grafo de conflito que as posições p_i participavam. Ao realizar este processo, os pontos adjacentes que tiveram reduzido a zero *(0)* o grau de um de seus vértices foram removidos também do grafo de conflitos pelo algoritmo recursivo, ficando a regra formulada conforme descrito a seguir:

a') Dado um ponto p, se existe p_i posições candidatas de p sem arestas de conflitos, declare a posição p_i tendo i o menor índice como parte da solução e elimine todas as outras posições de p do grafo de conflitos.
Sendo um ponto q adjacente a p, elimine todas as arestas entre p e q do grafo de conflito e caso ao eliminar as arestas uma posição q_j tenha reduzido a zero o seu grau, inicie o procedimento *a'* para o ponto q.

3.2 Heurística ILS.

A metaheurística ILS consiste em buscar um bom resultado para o problema caminhando por ótimos locais. A cada iteração do algoritmo ILS é realizado perturbações na solução corrente baseadas nas soluções já encontradas. Nesta nova solução é realizado a busca local e apos aplicados os critérios de aceitação, a nova solução poderá ser aceita ou não. Este processo iterativo se repete até que a condição de parada seja atingida. A qualidade da solução inicial é importante por afetar o desempenho do algoritmo, dado que soluções iniciais de baixa qualidade demandam de custo computacional para serem refinadas em contrapartida, soluções iniciais de boa qualidade melhoram significativamente a performance do algoritmo Lourenço, et. Al (2002).

Os pontos mais importantes do ILS são a perturbação, busca local e critério de aceitação. A perturbação é responsável pelo deslocamento da pesquisa num sub espaço de busca definido por diferentes soluções que são ótimos locais de um determinado procedimento de otimização Lourenço, et. Al (2002). Estas perturbações necessitam ser forte o suficiente para realizar os deslocamentos permitindo a busca local explorar diferentes soluções escapando de ótimos locais e fraca o suficiente para não deslocar a pesquisa para posições muito distante que poderia levar a uma busca aleatória. A busca local procura por um ótimo local para solução perturbada e caso esta solução atenda ao critério de aceitação ela será aceita, sendo então atribuída à solução corrente.

A heurística ILS proposta se divide em construção da solução inicial por heurística construtiva e heurística ILS composta de: perturbação, busca local e critérios de aceitação, os quais são descritos a seguir juntamente com os detalhes de implementação.

3.2.1 Construção da solução inicial.

A construção da solução inicial foi realizada por uma heurística construtiva baseada no algoritmo "backtracking" utilizado na Busca Exaustiva de Christensen et al.(1993,1995) e também no Algoritmo exato para conjunto independente de vértices proposto por Dowsland (1987). Para o início do processamento a lista de vértices é ordenada em ordem crescente do grau dos vértices, sendo também a lista de vértices adjacentes de cada vértice, ordenada da mesma forma. O algoritmo de "backtracking" proposto, gerencia o nível de busca local em profundidade através de um parâmetro **φ** que indica o **Limite para chamadas recursivas** que poderá ser realizado. Após o processamento da lista de vértices, uma solução é fornecida, tendo selecionado para cada ponto cartográfico p uma posição candidata i, com o menor número de conflitos e vértices adjacentes.

No processo de construção da solução, o algoritmo recebe um vértice $v_k \in p$, p um ponto cartográfico qualquer. O ponto cartográfico p é analisado quanto à existência de conflitos, sendo construído um conjunto $S(p_i)$ para cada posição $p_i \in p$, contendo os vértices adjacentes em conflito com p_i e o número de conflitos para cada vértice.

Após a construção dos conjuntos $S_i(p_i)$, $p_i \in p$, $(i=1,..n)$, o algoritmo decide entre duas situações.

a) Caso exista $S_i(p_i)=\emptyset$ $(i=1,..n)$, o vértice p_i é selecionado para solução, o ponto cartográfico p é definido como resolvido e os vértices restantes do ponto cartográfico p são colocados como inativos.
b) Caso todos os conjuntos $S_i(p_i)\neq\emptyset$ $(i=1,..n)$, então todos os vértices $p_i \in p$ possuem conflitos. O algoritmo seleciona vértice a vértice p_i para a solução corrente, em ordem crescente de número de conflitos e grau, marcando-o como participante de resolução de conflitos. Caso o vértice p_i participe de um ciclo, ele estará marcado anteriormente, sendo então descartado e o próximo vértice p_{i+1} será selecionado. A partir do vértice p_i o algoritmo executa uma busca em profundidade sendo chamando recursivamente, de acordo com o parâmetro φ, para cada vértice em conflito adjacente a p_i. Caso o conflito para p_i seja resolvido após as chamadas recursivas, o algoritmo executa o processo da etapa *a* para este vértice. Caso após o processamento todos os vértices continuem em conflitos, o vértice com o menor número de conflitos e menor grau será selecionado para compor a solução corrente.

O algoritmo de "backtracking" foi utilizado para a busca local em um ponto sendo chamado com o parâmetro **φ** com o nível de profundidade em zero *(1)*.

3.2.2 Perturbação.

Para realizar a perturbação o algoritmo seleciona na *LVS* um vértice em conflito $v_k \in p$, p um ponto cartográfico qualquer e seleciona para os pontos cartográficos q adjacentes a p outras posições para compor a solução corrente. Quando um movimento aumenta o número de conflitos, o algoritmo busca, aleatoriamente, nos vértices adjacentes de todas as posições do ponto cartográfico, um movimento que restaura ou reduz o número de conflitos. A perturbação aumenta gradativamente em profundidade sendo o algoritmo chamado recursivamente para cada vértice adjacente em conflito, tendo o limite de profundidade o menor valor entre o **contador de busca sem melhora** e o **Limite para chamadas recursivas**.

Quando o **contador de busca sem melhora** atinge o **Limite para perturbação** o algoritmo de perturbação passa a relacionar todos os vértices adjacentes e não somente os vértices adjacentes em conflitos e desta maneira distanciando a solução perturbada cada vez mais da posição da solução corrente, atendendo a **restrição de movimentos**.

3.2.3 Busca Local

A busca local pode ser realizada de duas formas, sendo a **Busca local combinada com o ponto adjacente** executada imediatamente a perturbação e a **Busca local combinada com um conjunto de pontos adjacentes**, de acordo parâmetro **Limite para busca conjunto**.

3.2.3.1 Busca local combinada com o ponto adjacente.

A busca local combinada com o ponto adjacente realiza a busca local pelo algoritmo de "backtracking" num ponto cartográfico p e seus pontos adjacentes, armazenando em uma lista o número de conflitos gerados pela seleção de $p_i \in p$ e $q_j \in q$ ponto adjacente a p, a soma do número de vértices adjacentes de p_i e q_j e os vértices p_i e q_j. Desta lista será selecionado o melhor movimento de melhora com os vértices de menor grau. Não havendo estes movimentos, os movimentos da lista são selecionados

aleatoriamente sendo aceito o primeiro que não piora a solução e atenda ao critério de **restrição de movimentos**.

3.2.3.2 Busca local combinada com um conjunto de pontos adjacentes.

O algoritmo da busca local combinada com um conjunto de pontos adjacentes efetua o mesmo processo que o movimento anterior para os vértices adjacentes em conflito. Dado um ponto cartográfico *p* em conflito, para cada vértice $p_i \in p$ o algoritmo seleciona aleatoriamente, limitado ao **índice de construção da LVBL**, os vértices em conflito q_k adjacentes a p_i, para compor a LVBL e efetua a busca local em cada ponto cartográfico $q \in q_k$. Desta maneira é construído uma lista de movimentos que possuirá um conjunto de vértices, que poderão compor a solução, sendo selecionados simultaneamente respeitando o critério de **restrição de movimentos**. Este movimento possui um custo computacional alto, entretanto apresenta excelentes resultados para problemas com um número elevado de vértices adjacentes.

3.2.4 Critérios de aceitação

Um movimento é a seleção de uma posição candidata $p_i \in p$, *(i=1,..n),* para receber o rótulo, sendo aceitos somente os movimentos de melhora ou os que não alteram a solução corrente. Os movimentos são coordenados por dois critérios: **restrição de movimentos** e **restrição de vértices**. Ao ser selecionado o vértice armazena a **restrição de movimentos** que é calculada pela soma do **contador de iterações**, a **base de restrição de movimento** e um número aleatório entre *0* e **o índice de restrição de movimento**. O coeficiente do **índice de restrição de movimento** é calculado pela soma de *10* mais *10%* do tamanho da **LVS**, sendo limitado em *200*. A **base de restrição de movimento** é calculada com *20%* do valor do **índice de restrição de movimento**.

Os movimentos somente serão aceitos caso seja um movimento de melhora ou o valor da restrição de movimentos seja maior que o **contador de iterações**. Com esta estratégia os movimentos mais recentes são proibidos, impedindo que o algoritmo cicle em torno de um ótimo local ou fique preso em uma região com diversas soluções com um mesmo número de conflitos.

O critério de **restrição de vértices** é aplicado na construção da lista de vértices adjacentes candidatos para busca local, *LVBL*. O **índice de construção da LVBL** será calculado utilizando o **tempo máximo sem melhora**. Durante o processamento da **LVBL**, quando o **tempo máximo sem melhora** é ultrapassado, o algoritmo incrementa o **índice de construção da LVBL** até o valor máximo de *6* sendo retornado a *1* quando ocorre uma resolução de conflitos.

3.2.5 Detalhes de implementação.

A estratégia adotada para o refinamento da solução foi resolver os conflitos partindo dos vértices em conflitos e seus vértices adjacentes, efetuando perturbações e busca local nas listas com estes vértices selecionados, **LVS**. Esta lista é menor que a lista com todos os vértices do grafo de conflitos, podendo assim proporcionar ao algoritmo um ganho em custo computacional. Para tornar o ILS proposto mais inteligente, durante o processamento, variáveis de adaptação orientam ao algoritmo sobre o nível de perturbação a ser realizada, como proceder na busca local e nos critérios de aceitação. Estas variáveis se modificam de acordo com as características do problema e da solução corrente proporcionando uma adaptação do algoritmo a estas condições.

Para efetuar estes processos o algoritmo utiliza os seguintes parâmetros:

- **Tempo máximo de processamento.**
- **Número de LVS.**
- **Tempo máximo sem melhora.**
- **Limite para chamadas recursivas.**
- **Limite para busca conjunto**
- **Limite para perturbação.**

O algoritmo realiza o refinamento das soluções construindo e processando as **LVS**, até que seja atingido o **Número de LVS a serem processadas** ou o **Tempo máximo de processamento** ou uma solução sem conflitos. Durante o processamento, quando o intervalo de tempo entre o último conflito resolvido e o valor atual ultrapassar o valor do **tempo máximo sem melhora**, o algoritmo incrementa o **contador de busca sem melhora**. Durante o processamento o algoritmo verifica a cada **LVS** processada, se o número de conflitos reduziu no mínimo *2%*, caso contrário é incrementado o **contador de busca sem melhora**.

Desta maneira enquanto o tempo entre conflitos resolvidos for menor ou igual ao **tempo máximo sem melhora** ou *2%* dos conflitos forem resolvidos, o algoritmo concentrará o processamento nos vértices em conflitos e seus vértices adjacentes. Na medida que os conflitos forem resolvidos e o tempo entre resoluções aumentar, o algoritmo expandirá sua busca através da expansão da **LVS** para os vértices sem conflitos. Neste sentido a **LVS** aumentará de tamanho enquanto os conflitos não forem resolvidos e diminuirá enquanto melhorar a solução corrente. Ajustando o tamanho da **LVS** de acordo com as condições em que as soluções são modificadas é proporcionar ao algoritmo adaptar-se dinamicamente de acordo com as características de cada problema e as características do problema durante o processamento.

A **LVS** é construída a partir da lista de vértices da solução corrente, sua construção é modificada de acordo com o comportamento dos resultados obtidos pelo algoritmo utilizando o **contador de busca sem melhora**. Inicialmente o **contador de busca sem melhora é zero** *(0)*, sendo assim, a primeira **LVS** é construída somente com os vértices em conflitos e seus vértices adjacentes. A partir da lista de vértices do grafo de conflitos, o algoritmo verifica se o vértice esta em conflito, caso afirmativo o vértice será selecionado juntamente com todos os vértices adjacentes. A lista de vértices adjacentes será verificada e caso algum vértice adjacente esteja em conflito, o algoritmo será chamado recursivamente para este vértice até que o **contador de busca sem melhora** seja alcançado ou a lista completa dos vértices do grafo de conflitos seja selecionada.

O Parâmetro **Limite para busca conjunto** é responsável pela ativação do movimento **Busca local combinada com um conjunto de pontos adjacentes**. Este movimento é ativado quando o **contador de busca sem melhora for superior** ao **Limite para busca conjunto**. O Parâmetro **Limite para perturbação** é responsável pela ativação da perturbação para vértices sem conflitos sendo o movimento ativado quando o **contador de busca sem melhora** for superior ao **Limite para perturbação**.

4. Resultados computacionais.

Os testes foram realizados em dois conjuntos de instâncias, sendo o 1º conjunto proposto por Lorena (2006) e disponibilizado em http://www.lac.inpe.br/~lorena/instancias.html[1], o 2º foi proposto por Taillard(2009) e

disponibilizado em http://mistic.heig-vd.ch/taillard/problemes.dir/problemes.html[2], totalizando 153 instâncias, conforme tabela a seguir.

	Lorena[1]						Taillard[2]
Nº de pontos cartográficos	25	100	250	500	750	1000	13206
Nº de instâncias	8	25	25	25	25	25	20

Tabela 1. Relação de pontos por instâncias.

Os testes computacionais foram realizados em um PC com processador Pentium®IV 3.0 GHz, 1 GigaBytes de RAM e sistema operacional Linux SlackWare.

A heurística foi implementada em C ANSI no ambiente GCC V4.4.1, utilizando as técnicas de programação mais adequadas para tratamento de ponteiros, listas múltiplas encadeadas, ordenação de listas e processamento recursivo Ziviane (2003).

Para cada conjunto de instâncias, com os parâmetros relacionados abaixo, à aplicação com a heurística implementada foi executada 10 vezes, sendo tomada para análise à média dos resultados de cada conjunto de instâncias.

Parâmetro	Valor
Tempo máximo de processamento	**30 Segundos**
Número de LVS	**1000**
Tempo máximo sem melhora	**0,100 Segundos**
Limite para chamadas recursivas	**4**
Limite para busca conjunto	**0,500 Segundos**
Limite para perturbação	**1**

Tabela 2. Parâmetros do algoritmo.

Na Tabela 3 são apresentados os resultados dos testes para as instâncias propostas por Lorena (2006) contendo o percentual de rótulos livres e o desvio padrão para as etapas de *0.5, 1, 2, 3, 4 ,5 ,10* e *30* segundos de processamento.

Para as instâncias de *25* pontos cartográficos o desvio padrão é alto devido cada rótulo equivaler a *4%* da amostra, sendo o desvio em relação à média equivalente a aproximadamente *2* rótulos. Para as instâncias de *100* e *250* pontos cartográficos não são feito análises devido à solução ótima ter sido encontrada em tempo computacional baixíssimo. Para as instâncias de *500, 750* e *1000* pontos cartográficos, o desvio padrão em função da média do número de rótulos livres são aproximadamente *2, 5* e *10* rótulos respectivamente.

O coeficiente de variação para as instâncias de *25* pontos cartográficos é aproximadamente *9*, podendo ser considerado alto devido ao pequeno número de pontos. Para as instâncias de *500, 750 e 1000* pontos cartográficos, o coeficiente de variação são aproximadamente *0.39, 0.72* e *0.97* denotando um baixo percentual de dispersão.

Os valores em percentuais de rótulos livres de Alvim&Taillard (2009) são alcançados em *0,5s* de processamento, sendo igualado para *500* pontos e superados para *750* e *1000* pontos cartográficos. Em *30* segundos de processamento, para as instâncias de *500* pontos, foi possível melhorar o percentual de rótulos livres em *0,01%*, enquanto para *750* e *1000* pontos cartográficos os percentuais melhoraram *0,18%* e *1,08%* respectivamente.

Na Tabela 4 são apresentados os resultados dos testes para as instâncias propostas por Taillard (2009) contendo o percentual de rótulos livres para as etapas de *0.5, 1, 2, 3, 4 ,5 ,10* e *30* segundos de processamento. Para este conjunto de instâncias o desvio padrão, para 10 execuções foi inferior a *0,01* denotando um coeficiente de dispersão baixíssimo. Para este conjunto de instâncias, os resultados da heurística construtiva foram superior em *19* das *20* instâncias propostas. Para a instância CH08_06B a melhora foi de 2,02% enquanto para a instância CH16_06B a melhora foi de 8,68%.

Resultado / Pontos	Alvim &Taillard		Solução Inicial		0,5s Execução		1s Execução		2s Execução		3s Execução		4s Execução		5s Execução		10s Execução		30s Execução	
	%Rótulos Livres	Desvio Padrão	%Rótulos Livres	Desvio Padrão	%Rótulos Livres	Desvio Padrão	%Rótulos Livres	Desvio Padrão	%Rótulos Livres	Desvio Padrão	%Rótulos Livres	Desvio Padrão	%Rótulos Livres	Desvio Padrão	%Rótulos Livres	Desvio Padrão	%Rótulos Livres	Desvio Padrão	%Rótulos Livres	Desvio Padrão
25	-----	-----	82,00	8,000	83,50	7,540	83,50	7,540	83,50	7,540	83,50	7,540	83,50	7,540	83,50	7,540	83,50	7,540	83,50	7,540
500	**99,67**	-----	99,46	0,363	99,67	0,392	99,68	0,389	99,68	0,388	99,68	0,388	99,68	0,388	99,68	0,387	99,68	0,387	**99,68**	0,387
750	97,73	-----	96,74	0,955	97,80	0,706	97,85	0,703	97,89	0,701	97,91	0,699	97,92	0,705	97,93	0,705	97,94	0,707	97,95	0,702
1000	**92,68**	-----	90,56	0,938	92,92	0,947	93,03	0,958	93,32	0,953	93,52	0,898	93,60	0,892	93,64	0,904	93,69	0,909	**93,76**	0,909

Tabela 3 - Soluções e desvio padrão por grupo de instâncias em função do tempo.

% Tempo / Instância	Alvim& Taillard	Solução Inicial	0,5s Execução	1s Execução	2s Execução	3s Execução	4s Execução	5s Execução	10s Execução	30s Execução
CH24_02B	92,55	92,71	92,94	92,95	93,32	93,68	93,91	94,02	94,09	94,65
CH16_03B	88,73	89,08	89,20	89,20	89,47	89,85	89,96	90,12	90,30	91,05
CH12_04B	86,69	87,37	87,46	87,47	87,75	88,06	88,16	88,19	88,24	88,91
CH08_06B	***84,42***	85,83	85,83	85,84	85,89	85,98	85,99	86,03	86,12	***86,43***
CH32_02B	84,73	86,06	86,09	86,10	86,54	86,93	87,07	87,25	87,36	88,47
CH16_04B	75,49	78,87	78,87	78,88	78,98	79,15	79,30	79,43	79,49	79,89
CH08_08B	71,16	75,72	75,72	75,72	75,77	75,94	75,98	76,01	76,05	76,34
CH04_16B	84,42	78,34	78,34	78,34	78,46	78,66	78,80	78,92	78,98	79,68
CH24_03B	74,28	78,14	78,14	78,15	78,23	78,35	78,55	78,69	78,76	79,47
CH18_04B	69,96	74,60	74,60	74,60	74,65	74,92	75,12	75,25	75,32	75,97
CH12_06B	66,66	71,95	71,95	71,95	71,95	72,07	72,13	72,19	72,27	72,85
CH09_08B	65,10	71,34	71,34	71,34	71,35	71,51	71,60	71,66	71,75	72,17
CH42_02B	74,15	77,97	77,97	77,97	78,17	78,56	78,71	78,78	78,85	79,48
CH28_03B	67,12	72,30	72,30	72,31	72,33	72,52	72,67	72,78	72,86	73,51
CH21_04B	62,86	69,12	69,12	69,12	69,13	69,27	69,32	69,40	69,48	70,22
CH14_06B	58,87	65,64	65,64	65,64	65,64	65,68	65,72	65,74	65,77	66,27
CH48_02B	68,17	73,28	73,28	73,28	73,31	73,48	73,58	73,70	73,73	74,75
CH32_03B	60,44	67,27	67,27	67,27	67,27	67,40	67,49	67,54	67,61	68,22
CH24_04B	56,55	63,73	63,73	63,73	63,74	63,77	63,84	63,90	63,95	64,32
CH16_06B	***51,98***	60,01	60,01	60,01	60,02	60,05	60,11	60,15	60,17	***60,66***

Tabela 4 - Soluções por instâncias em função do tempo.

O baixo percentual de dispersão pode ser percebido nos Gráficos *1, 2 e 3* onde há uma concentração das soluções no intervalo entre mais e menos uma unidade do desvio padrão em relação à média, onde as soluções obtidas em 0,5 segundos estão em vermelho e as obtidas em 30 segundos estão em verde. Pode-se observar nos gráficos que a melhora da qualidade das soluções, nestes dois momentos, tendem a ser homogênea para as instâncias propostas por Lorena.

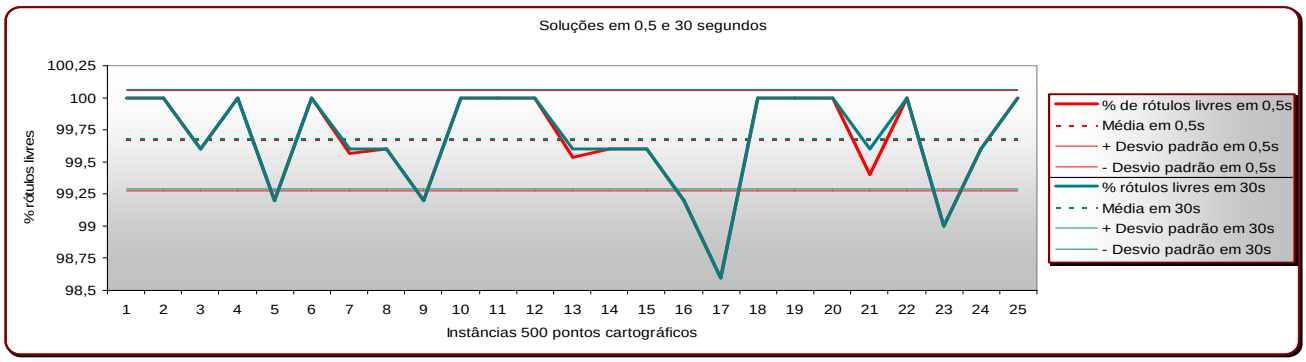

Gráfico 1 Desvio padrão das instâncias de 500 pontos cartográficos.

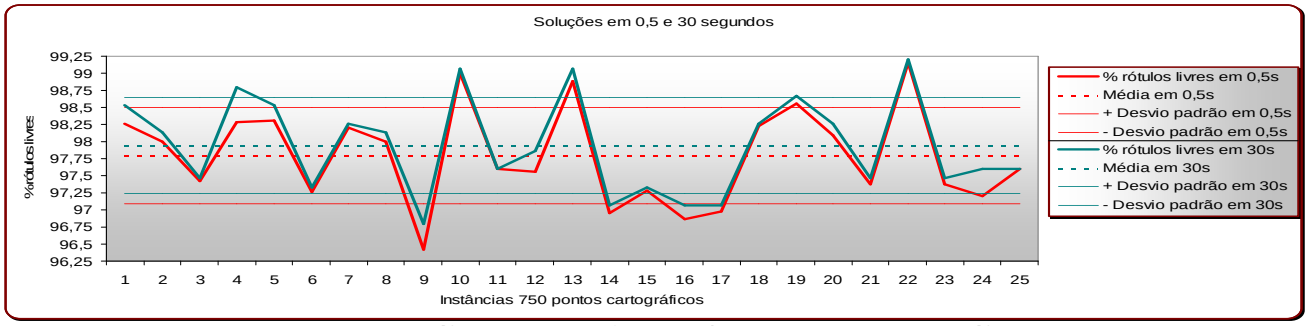

Gráfico 2 Desvio padrão das instâncias de 750 pontos cartográficos.

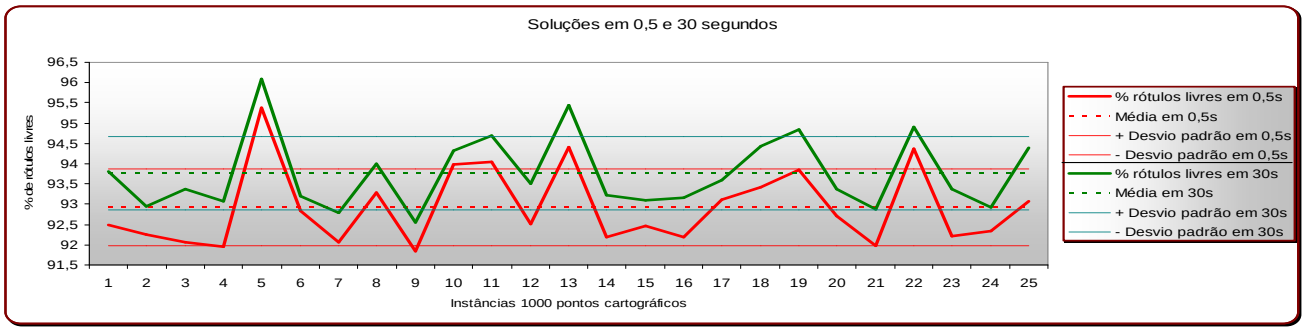

Gráfico 3 Desvio padrão das instâncias de 1000 pontos cartográficos.

5. Conclusão.

Pelos resultados obtidos podemos concluir que a heurística proposta demanda um tempo computacional aceitável e os resultados em rótulos livres são satisfatórios. Para as instâncias com *500*, *750* e *1000* pontos cartográficos foi encontrado um baixo desvio padrão e coeficiente de dispersão revelando uma homogeneidade das soluções. Para as instâncias com *13206* pontos cartográficos os resultados apresentados pela heurística construtiva superam aos melhores resultados encontrados na literatura, revelando que a heurística ILS proposta é robusta e apresenta bons resultados mesmo em grafos densos. A estratégia da heurística em se adaptar de acordo com as características das instâncias e dos conflitos em resolução pôde proporcionar um ganho final nos resultados de conflitos resolvidos.

6. Referencias Bibliográfica

[01] CRAVO, G.L.; RIBEIRO, M.G. LORENA L.A.N. Heurística gulosa para o problema da rotulação cartográfica de pontos. 2006. Revista Educação Tecnológica, Ano 2, Número 1 Abr/Set 2006– Departamento de Ciência da Computação e Informática Faculdade Aracruz - Instituto Nacional de Pesquisas Espaciais, São José dos Campos, 2006.

[02] CRAVO[1], G.L.; RIBEIRO[1 2], M.G. LORENA[2] L.A.N. Um Grasp eficiente para o problema da rotulação de pontos. 2007. Departamento de Ciência da Computação e Informática Faculdade Aracruz - Instituto Nacional de Pesquisas Espaciais, São José dos Campos, 2007.

[03] 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.

[04] FREMAN, H . , (1991) Computer name placement. In: Maguire, D. J.; Goodchild, M. F., Rhind, D. W. eds. Geographic Information Systems: Principles and aplications. New York, Longman Scientific, v 1 , p 445-456.

[05] LOURENÇO, H.R., MARTIN, O., STÜTZLE, T.. Iterated Local Search. In F. Glover and G. Kochenberger, editors, Handbook of Metaheuristics, pages 321–353. Kluwer Academic Publishers, Norwell, MA, 2002

[06] Ribeiro G. M., Lorena, L. A. N. (2005), Heuristics for cartographic label placement problems, Computers and GeoSciences, In Press.

[07] Verner O. V., Wainwright R. L., Schoenefeld D. A. (1997), Placing text labels on maps and diagrams Using Genetic Algorithms with Masking, *INFORMS Journal on Computing*, 9, 266-275.

[08] Ziviani, N. Projetos de algoritmos com implementação em Pascal e C. Capitulo 4, Editora Thompson , 2003

[09] YAMAMOTO, M. Novos algoritmos para o problema de rotulação cartográfica de pontos. 2003. Tese (Doutorado em Computação em Matemática Aplicada) – Instituto Nacional de Pesquisas Espaciais, São José dos Campos, 2003.

[10] YAMAMOTO M.; CAMARA G.; LORENA L.A.N. Tabu search heuristic 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.

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

[12] Wolff, A. e Strijk, T. The map labeling bibliography. URL on June 6th, 2005. http://i11www.ilkd.unikarlsruhe.de/~awolff/map-labeling/bibliography/

[13] Alvim, Adriana C.F., Taillard, Eric D., T. POPMUSIC for the point feature label placement problem. European Journal Of Operational Research 192 (2009)396-413