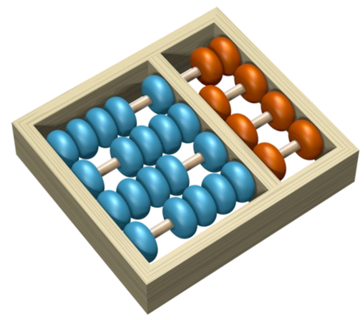
Problema do Caixeiro Viajante com Coleta e Entrega

Aluno: Fabrício Caluza Machado, e-mail: *fabcm1@gmail.com*
Orientador: Flávio Keidi Miyazawa, e-mail*: fkm@ic.unicamp.br*
Instituto de Computação – IC - UNICAMP



Financiamento: PIBIC / CNPQ
Palavras Chave: Otimização combinatória - Programação inteira - Caixeiro Viajante

Resumo

Neste projeto foram estudados métodos poliédricos e implementado um algoritmo branch and cut para a resolução do problema do caixeiro viajante com coleta e entrega (PDTSP). Os resultados obtidos pelo algoritmo implementado foram comparados com os resultados encontrados na literatura recente.

1. Introdução

O PDTSP é definido sobre um grafo com pesos associados às arestas, onde além da origem existem n pares de vértices s, t. O objetivo é encontrar uma rota orientada do caixeiro viajante com custo mínimo, tal que para cada i, o vértice s_i é visitado antes do vértice t_i. Este é um problema especialmente difícil do ponto de vista prático, visto que as instâncias resolvidas pelos algoritmos atuais são bem menores que as do caixeiro viajante (TSP) convencional.

A formulação do problema como um programa linear inteiro envolve o uso de uma variável binária x_e para cada aresta do grafo. Os vértices são os pares s_i, t_i além da origem, representada pelo par de vértices Oa, Ob (dizemos que o ciclo começa em Oa e termina em Ob). Dado um conjunto de vértices S, seja δ(S) a fronteira de S, isto é:

$$\delta(S) = \{\{i,j\} \in E \ ; \ i \in S, j \notin S\}$$

E dado um conjunto de arestas E', considere:

$$x(E') = \sum_{e \in E'} x_e$$

Com estas notações, podemos escrever as restrições do problema da seguinte maneira:

$$\begin{aligned}
&(1) \quad x_{Oa,Ob} = 1 \\
&(2) \quad x(\delta(i)) = 2 \quad \forall i \in V \\
&(3) \quad x(\delta(S)) \geq 2 \quad \forall S \subseteq V,\ 3 \leq |S| \leq |V|/2 \\
&(4) \quad x(\delta(S)) \geq 4 \quad \forall S \in \mathcal{U} \\
&(5) \quad x_e \in \{0,1\} \quad \forall e \in E
\end{aligned}$$

Sendo que U é a família de todos os subconjuntos S tais que Oa está em S, Ob não e existe t_i em S com s_i fora.

2. Metodologia

O problema é resolvido através de um algoritmo branch and cut que atua sobre a relaxação linear do problema. Inicia-se apenas com as restrições (1) e (2) (note que há um número exponencial de restrições dos tipos (3) e (4)) e encontra-se uma solução ótima para o problema relaxado. A seguir, o algoritmo busca inserir iterativamente *planos de corte* (seguidos de nova otimização), isto é, restrições extras dos tipos (3) e (4) que eliminam soluções inviáveis. Também foram consideradas outras classes de planos de corte (geralmente associados a propriedades combinatórias do problema) que eliminam soluções ótimas fracionárias mas não são violados por nenhuma solução inteira. Dada uma solução fracionária que satisfaz as restrições (1) a (4), nem sempre o algoritmo é capaz de encontrar restrições deste segundo tipo, então, o programa divide o problema atual em dois subproblemas (*branching*).

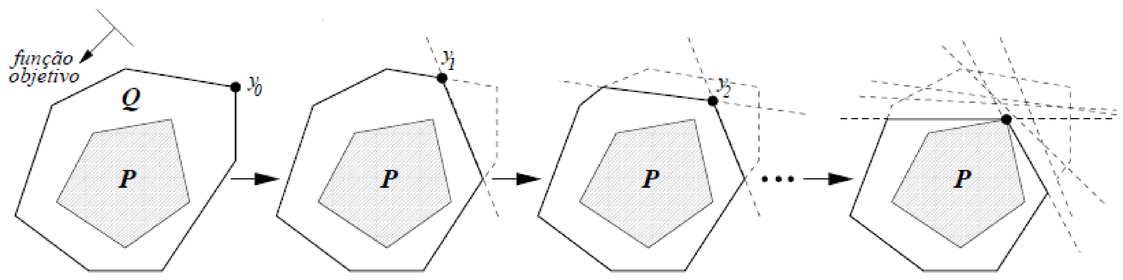


Figura 1: Sequência de inserções de planos de corte.

A seguir, algumas ilustrações de soluções intermediárias de uma instância do problema. Os vértices s_i estão representados em azul, os vértices t_i em vermelho, as arestas com valores fracionários aparecem em vermelho. As soluções são obtidas após inclusão sucessiva das restrições do problema. No fim, a solução ótima da instância.

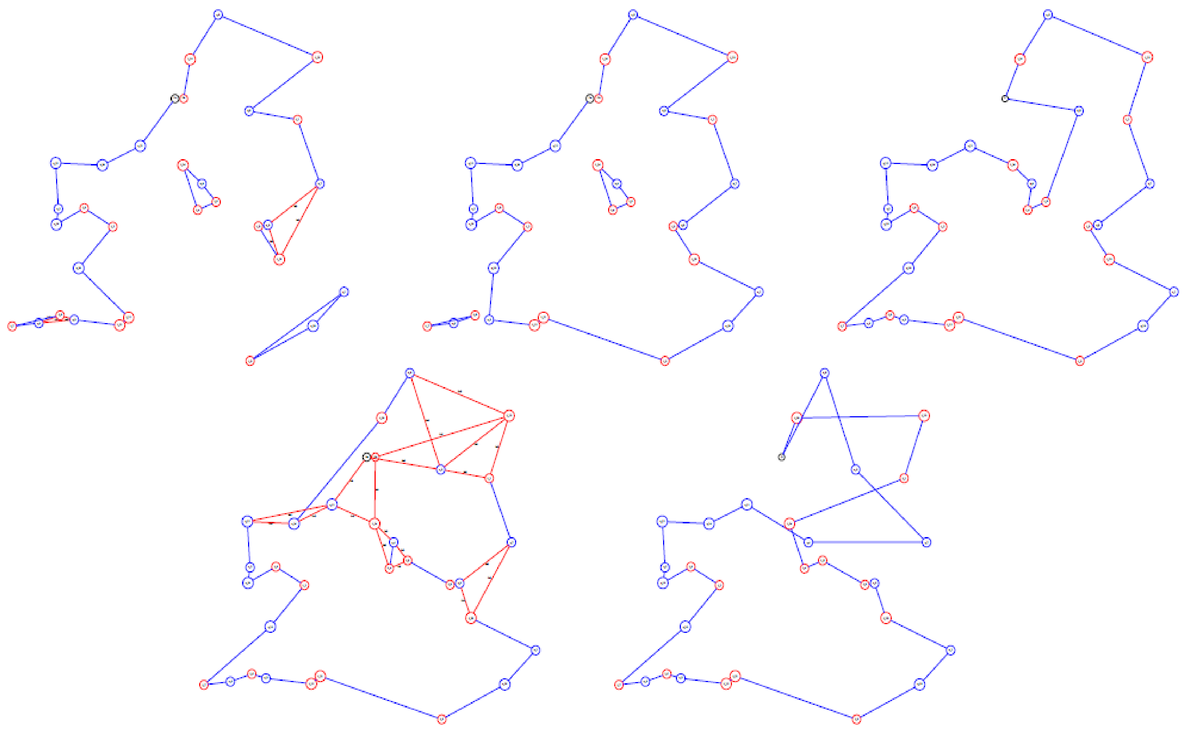


Figura 2: Exemplos de soluções intermediárias. No canto inferior direito, a solução ótima da instância.

3. Resultados

Comparando-se os resultados do programa implementado com resultados da literatura recente, foram obtidos bons resultados nas instâncias pequenas e médias, entretanto não foi possível obter os mesmos resultados nas instâncias maiores.