

1 Cluster-First Route-Second Method

Este método faz uma simples clusterização do conjunto de vértices e então determina uma rota de veículos em cada cluster. Descreveremos o próximo algoritmos:

- Fisher and Kaikumar
- The Petal algorithm
- The Sweep algorithm
- Taillard

1.1 Algoritmo de Fisher and Jalkumar

O algoritmo de Fisher e Jaikumar [referência](#) é bem conhecido. Resolve um Problema Genérico de Tarefa (GAP) para formar clusters. O número de veículos k é fixo. O algoritmo pode ser descrito como segue:

1. Seed Selection. Escolha pontos semente J_k em V para inicializar cada cluster k .
2. Allocation of Customers to Seeds. Calcule o custo d_{ik} da alocar cada cliente i para cada cliente k com

$$d_{ijk} = \min(c_{0i} + c_{ijk} + c_{J_k 0}, c_{0J_k} + c_{J_k i} + c_{i0}) - c_{0J_k} + c_{J_k 0}.$$

3. Generalized Assignment. Resolver um GAP com custo d_{ij} , peso do cliente q_i e capacidade do veículo Q .
4. TSP Solution. Resolve o TSP para cada cluster correspondente para a solução do GAP.

1.2 Algoritmo Pétala

Uma extensão natural do algoritmo de varredura é gerar muitas rotas, chamadas pétalas [referência](#), e fazer uma seleção final resolvendo um problema de partição de conjunto da forma:

$$\min \sum_{k \in S} d_k x_k$$

sujeito a:

$$\sum_{k \in S} a_{ik} x_k = 1 \quad (i = 1, \dots, n)$$

com

$$x_k = 0, 1, k \in S,$$

onde S é o conjunto de rotas, $x_k = 1$ se, e somente se, a rota k pertence à solução, a_{ik} é o parâmetro binário igual a 1 somente se o vértice i pertence a rota k , e d_k é o custo da rota da pétala k . Se as rotas corresponde a setores contínuos de vértices, então esse problema possui o propriedade de coluna circular e é resolvido em tempo polinomial [referência](#).

1.3 O Algoritmo de varredura

O Algoritmo de varredura aplica-se a instâncias planares do VRP. Consiste de duas partes:

- Split: Clusters possíveis são iniciados formando rotação e centro de raio no depósito
- TSP: Uma rota de veículo é então obtida para cada cluster resolvendo o TSP.

Algumas implementações incluem uma fase de otimização posterior na qual os vértices são mudados entre clusters adjacentes, e as rotas são reotimizadas. Uma implementação simples deste método é como segue. onde assumimos que cada vértice i é representado por suas coordenadas polares (θ_i, ρ_i) , onde θ_i é o ângulo e ρ_i é o comprimento de raio. Atribuindo um valor $\theta^* = 0$ para um vértice arbitrário i^* e calculando os ângulos remanescentes de $\text{IMG6}(0, i^*)$. Graduando os vértices em ordem crescente de sua IMG2.

1. Route Initialization. Escolha um não utilizado veículo k .
2. Route Construction. Inicie dos vértices não roteados tendo ângulo menor, atribuindo vertices ao vértice k enquanto sua capacidade ou o comprimento máximo da rota não é excedido.
3. Route Optimization. Otimizar cada rota de veículos separadamente resolvendo o correspondente TSP (exatamente ou aproximadamente).

1.4 Algoritmo de Talliard

O algoritmo de Talliard [referência](#) define-se vizinhança usando o λ -intermutação mecanismo de Geração [referência](#). Rotas individuais são reotimizadas usando algoritmo de otimização de [referência](#). Um traço nobel do algoritmo de Talliard é a decomposição dos problemas essenciais em subproblemas.

Em problemas planares, estes subproblemas são obtidos inicialmente particionando os vértices em setores centradas no depósito, e em regiões concentricas dentro de cada setor. Cada subproblema pode ser resolvido independentemente, mas movimentos periódicos de vértices para setores adjacentes são necessários. Isso faz sentido quando o depósito é centrado e os vértices são uniformemente distribuídos no plano.

Para problemas não planares, e para problemas planares não possuindo esta propriedade, o autor sugere uma método de partição diferente baseado no calculo da mais curta espaço de arborecência enraizadas no depósito. Esse método de decomposição é particularmente bem conviniente para implementações paralelas com subproblemas podem então ser distribuídos através de varios processadores.

A combinação dessas estratégias produz excelentes resultados computacionais.

2 Método de roteamento seguido de cluster

O método de roteamento seguido de cluster constroi numa primeira fase uma grande rota de TSP, observando as restrições laterais, e depositando esta rota dentro das

possíveis rotas de veículos numa segunda fase. Essa ideia aplicasse a problemas com uma número livre de veículo. Foi primeiro posto por Beasley o qual observou que o problema da segunda fase é um problema de caminho mínimo padrão em grafo acíclico e pode ser assim resolvido em tempo $O(n^2)$. No algoritmo de caminho mínimo, o custo d_{ij} da viagem entre os nós i e j é igual a $c_{0i} + c_{0j} + l_{ij}$, onde l_{ij} é o custo da viagem de i a j na rota TSP.

Não estamos conscientes de algumas experiências computacionais mostrando que a heurística de roteamento primeiro seguido de cluster são competitivas com outras aproximações.