Roteamento de veículos usando a heurística baseada em Centroide e Simulated Annealing

Juan C. E. O. Valdivia[1], **Eduardo A. Pacheco**[2], **Lucas F. da Nóbrega** [1],

[1]Escola de Engenharia de São Carlos - Universidade de São Paulo (USP)

[2]Instituto de Ciências Matemáticas e de Computação - Universidade de São Paulo (USP)

juan.valdivia@.usp.br, ea.pacheco@usp.br e lucas.fernandes.nobrega@usp.br

Abstract. *The capacited vehicle routing problem (CVRP) is a famous NP-hard problem in the literature. The solution is to serve all customers by generating a route with minimal cost for each vehicle that starts from the same depot. The vehicles have uniform capacity and need to go back and forth to the depot. Each customer has a known demand and should receive only one visit from a single vehicle. In this work we implemented the centroid-based heuristic algorithm to solve the CVRP in polynomial time. The algorithm consists of three phases: cluster construction, cluster adjustment and route establishment. The notion of geometric center guides the first two phases. After the establishment of the groups the shorter route of each cluster is found. An algorithm based on Simulated Annealing is used to find the best route that each vehicle should follow.*

Resumo. *O problema de roteamento de veículos capacitados (CVRP) é um famoso problema NP-difícil da literatura. O objetivo é atender todos os clientes gerando uma rota com o menor custo para cada veículo que parte de um mesmo depósito. Os veículos têm capacidade uniforme e precisam ir e voltar para o depósito. Cada cliente possui uma demanda conhecida e deve receber apenas uma visita de um único veículo. Neste trabalho da disciplina de Inteligência Artificial é implementado o algoritmo heurístico baseado em Centroide para resolver o VRPC em tempo polinomial. O algoritmo é composto por três fases: construção de clusters, ajuste de clusters e estabelecimento da rota. A noção de centro geométrico guia as duas primeiras fases. Após o estabelecimento dos grupos é encontrada a rota de cada cluster. Um algoritmo baseada em Simulated Annealing é utilizado para encontrar a rota subótima que cada veículo deverá seguir.*

1. Introdução

O Problema de Roteamento de Veículos(VRP) vêm sendo estudado cada vez mais por estar presente em situações do cotidiano. Suas aplicações no ambiente industrial têm ajudado na redução de custos nos processos produtivos, dessa forma os serviços conseguem ser otimizados. Para os pesquisadores, este problema é desafiador pois ele é calssificado como NP-Hard. Problemas desta classificação são resolvidos com o uso de algoritmos heurísticos e meta-heurísticos.

Neste trabalho será reosolvido o Problema de Roteamento de Veículos Capacitado (CVRP). Ele é uma variação da versão clássica do VRP e tem como meta encontrar um conjunto de rotas a serem seguidas por um conjunto de veículos homogêneos que devem atender a demanda de cada uma lista de clientes. O custo total das rotas deve ser minimizado considerando as seguintes restrições: todas as rotas iniciam e terminam no mesmo depósito; cada um dos clientes é visitado apenas uma única vez e sua demanda deve ser atendida apenas por um veículo; além disso as demandas dos clientes de uma mesma rota não podem exceder a capacidade do veículo.

Heurísticas produzem boas soluções subótimas com desempenho computacional satisfatório para o CVRP. A solução ótima não é garantida, mas resultados próximos da melhor solução conhecida são encontrados. Neste trabalho é apresentado o algoritmo heurístico baseado em Centroide que estabelece o conjunto de rotas a serem seguidas pelos veículos. A metaheurística do Simulated Annealing é usada para resolver o Problema do Caixeiro Viajante (TSP) encontrando a rota subótima para cada veículo.

2. O problema de roteamento de veículos capacitados

No problema de roteamento de veículos capacitados, seja $G = (V, A)$ um grafo não direcionado, em que $V = \{0, 1, ..., n\}$ é o conjunto dos nós e A é o conjunto das arestas. O nó 0 é o depósito e o conjunto $W = V - \{0\}$ de n nós corresponde aos clientes. Cada vértice $\{i, j\}$ de A possui um custo $(c_{i,j} \geq 0)$. Cada cliente i de W precisa de q_i unidades do depósito. Um conjunto de M veículos idênticos com capacidade Q está no depósito.

Uma solução factível para o problema é composta pela partição $S = \{r_1, r_2, ..., r_m\}$ de V e pela permutação i de r_i que especifica a ordem dos clientes na rota com a restrição de que a demanda total de todos os clientes suportados pela rota r_i não ultrapasse a capacidade Q do veículo. O custo da rota $r_i = \{v_0, v_1, ..., v_{k+1}\}$ é dada por:

$$C(r_i) = \sum_{i=0}^{k} c_{i,i+1} \tag{1}$$

O custo total da solução S é a soma do custo das rotas, isto é:

$$TC(S) = \sum_{i=0}^{m} C(r_i) \tag{2}$$

A função objetivo de custo mínimo requer que as rotas visitem todos os clientes exatamente uma vez, independentemente do número de veículos em uso.

Na Figura 1 é mostrado um exemplo de solução com 3 rotas para um problema com 15 clientes. Note que cada rota começa e termina no depósito. O custo da rota 1 é 25, da rota 2 é 22 e da rota 3 é 19. O custo total dessa solução é 25 + 22 + 19.

3. Heurística baseada em Centroide

O algoritmo proposto em [Shin and Han 2011] consiste em três fases:

- **Construção de clusters**: O grupo (cluster) de clientes é construído usando o cliente mais distante do depósito como a semente do grupo. Após isso, os clientes

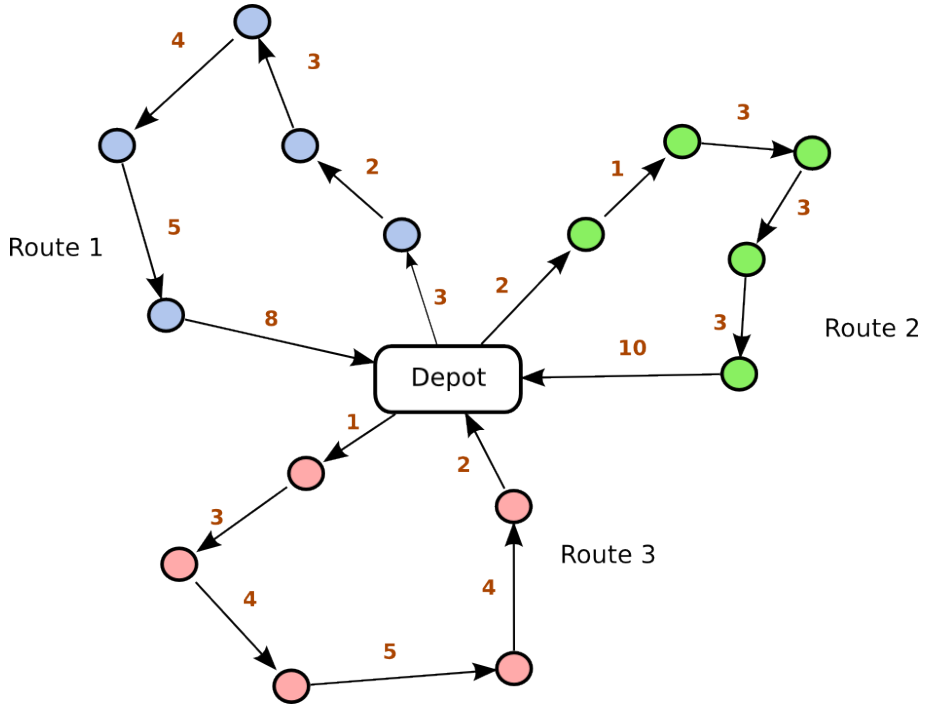

Figura 1. Solução de um exemplo de VRPC com 3 rotas.

mais próximos do centro geométrico (centroide) do cluster são selecionados, e o centroide é atualizado quando um cliente é adicionado. O processo continua enquanto a capacidade do veículo não for ultrapassada. A construção do grupo é finalizada quando isso ocorre. Após isso uma nova semente é selecionada e o processo de construção de novos grupos continua até que cada cliente esteja agrupado em um cluster.

- **Ajuste de clusters**: Os grupos formados na fase anterior são ajustados. Verifica-se para cada cliente se ele está mais próximo do centroide de um cluster vizinho do que o centroide de seu cluster atual. Caso isso ocorra e a capacidade do grupo vizinho não seja ultrapassada, esse cliente vai para o cluster vizinho e os centroides de ambos grupos são recalculados.
- **Estabelecimento da rota:** A rota subótima é encontrada visitando cada cliente apenas uma única vez dentro do cluster. Isso significa aplicar um algoritmo para resolver o *problema do caixeiro viajante (TSP)* usando a meta-heurística do Recozimento Simulado (Simulated Annealing) para cada grupo encontrado. Em cada cluster i, o cliente v_j faz parte do conjunto $l_i = \{v_0, v_1, ..., v_k\}$ de clientes que estão no cluster i. O centro geométrico (GC) do cluster l_i é definido por:

$$GC(l_i) = (\sum_{j=0}^{k} \frac{x_j}{k}, \sum_{j=0}^{k} \frac{y_j}{k}), \tag{3}$$

em que x_j e y_j são as coordenadas de v_j.

O algoritmo completo é mostrado no Algoritmo 1. Nas linhas 1 e 2 são chamados o algoritmo para construir os clusters e o algoritmo que resolve o problema TSP para cada cluster encontrado. Nas linhas 4 a 13 os clusters são ajustados e o algoritmo que resolve

o TSP para cluster é aplicado. Esses dois passos são repetidos até que não seja possível realizar nenhum ajuste. Nesse caso, é devolvida a melhor solução.

Algorithm 1 Algoritmo baseado em Centroide para o VRP Capacitado

Entrada: O número máximo de iterações para o algoritmo TSP que usa a heurística de Simulated Annealing

Saída : Melhor solução com as rotas e o custo total

```
1  constructClusters ()
2  solution= aplyingTSPClusters(maxIterSimulated)
3  bestSolution = solution
4  enquanto (true) faça
5      resp = clusterAdjustment()
       se (!resp) então
6          break
7      fim
8      senão
9          solution = aplyingTSPClusters(maxIterSimulated)
           se (solution.getTotalCost() < bestSolution.getTotalCost()) então
10             bestSolution = solution
11         fim
12     fim
13 fim
14 retorna bestSolution
```

4. Simulated Annealing para o TSP

O algoritmo heurístico Simulated Annealing [Russell and Norvig 2010] foi usado para resolver o *TSP*. O algoritmo Começa escolhendo uma rota inicial gerada aleatoriamente a partir do conjunto de todos as rotas válidas. A partir desse rota inicial, rotas vizinhas são geradas, verificando o quão boas elas são. Apesar de não poder testar todos as rotas válidas, o processo do Simulated Annnealing consegue encontrar uma solução subótima boa. Nesta heurística, a melhor solução é procurada na vizinhança da solução atual. Soluções vizinhas são geradas por meio da troca de dois clientes escolhidos de maneira aleatória na rota. Na Figura 2 é mostrado um exemplo em que os clientes F e D são trocados na rota da esquerda, gerando a rota da direita.

Na Figura 3 é mostrado um exemplo de funcionamento do algoritmo Simulated Annealing durante as iterações do algoritmo. Quando uma solução vizinha gerada for melhor, ela será a nova solução atual, ou seja ela é aceita incondicionalmente. Por outro lado, se ela for pior, ela só será aceita de acordo com o valor de uma probabilidade P cuja equação é:

$$P = e^{\frac{\Delta E}{Temperatura}}, \tag{4}$$

em que:

$$\Delta E = custo - novoCusto \tag{5}$$

Essa solução vizinha pior é aceita se $P > random(0, 1)$.

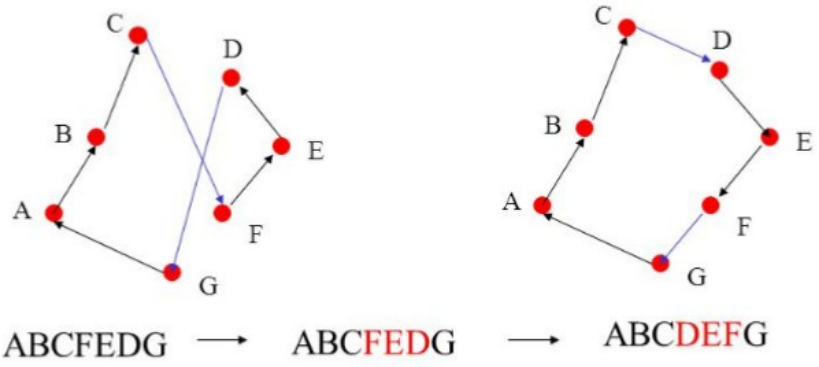

Figura 2. Troca dos clientes F e D da rota para gerar uma solução vizinha

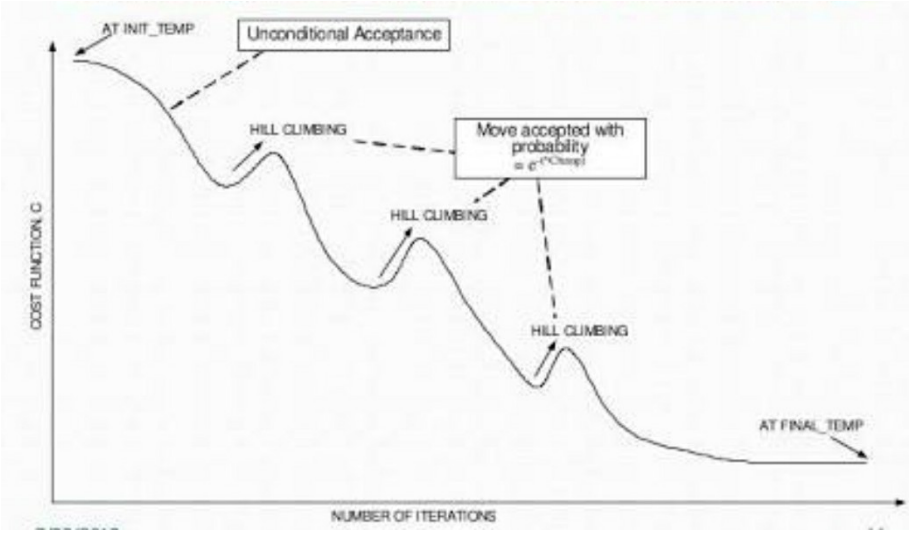

Figura 3. Metaheurística Simulated Annealing

Em cada iteração do algoritmo a temperatura é diminuída gradualmente usando um fator de esfriamento fr:

$$Temperatura = Temperatura - fr * Temperatura \tag{6}$$

Já que a temperatura é reduzida, a seleção de uma solução vizinha pior se torna cada vez mais difícil.

5. Resultados

O algoritmo foi testado no dataset de Augerat A, B e P. As instâncias do tipo A a localização dos clientes e suas demandas são geradas aleatoriamente; as instâncias da classe B são criadas por meio de clusters; e as instância do tipo P são versões modificadas de instâncias da literatura. O nome do arquivo A-n32-k5 indica que é um problema do tipo A, com 32 nós (31 clientes e 1 depósito) e com no mínimo 5 veículos necessários. Os

experimentos foram realizados no sistema operacional Windows 10 no processador Intel Core i7 8550U 1.8 GHz e 16 GB de RAM. O algoritmo foi implementado na linguagem Java no Eclipse.

Na Tabela 1 é mostrada o nome da instância, a melhor solução conhecida para o problema, o custo encontrado pelo algoritmo implementado, o número de rotas encontradas e o tempo de execução do algoritmo em milisegundos.

Tabela 1. Resultados do algoritmo para instâncias do dataset de Augerat

Instância	Melhor custo conhecido	Custo	Num. Rotas	Tempo (ms)
A-n32-k5	784	882.21	5	21
A-n33-k5	661	729.87	5	22
A-n33-k6	742	787.54	7	22
A-n45-k7	1146	1291.77	7	32
B-n31-k5	672	704.25	5	47
B-n34-k5	788	854.19	6	51
B-n35-k5	955	973.68	5	31
B-n38-k6	805	836.82	6	35
B-n44-k7	909	967.40	7	22
B-n66-k9	1374	1460.70	10	74
P-n16-k8	435	498.51	9	7
P-n19-k2	212	257.7	3	24
P-n20-k2	220	240.93	2	17
P-n23-k8	554	705.07	12	9
P-n40-k5	458	512.61	5	36
P-n50-k10	696	756.21	11	28

Note que o algoritmo implementado consegue encontrar uma solução subótima relativamente próxima da melhor solução conhecida em poucos milisegundos. Isso indica que esse algoritmo poderia ser utilizado caso seja necessário obter uma solução para o problema em um tempo curto.

6. Execução

A pasta Entradas contém os arquivos das instâncias dos problemas usados e a pasta src as classes do java implementadas, entre elas: CentroideVRP, Cluster, Customer, InstanceProblem, Route, Solution, Test, Truck e Tsp.

Para executar o programa com uma instância em particular, por exemplo A-n33-k5.vrp rode a classe Test e passe como parâmetro para o método inicializeInstanceFromFIle o caminho e o nome do arquivo:

```
instance.inicializeInstanceFromFIle ("C:\\Users\\Elias\\VRPFinal\\Entradas\\A-n33-k5.vrp");
```

7. Conclusão

Foi implementado o algoritmo baseado em Centroide proposto em [Shin and Han 2011] e para o passo de estabelecimento de rota foi usado o algoritmo Simulated Annealing. Os experimentos mostram que o algortimo consegue resolver problemas de até 66 clientes em um tempo menor que 74 milisegundos.

Referências

Russell, S. and Norvig, P. (2010). *Artificial Intelligence: A Modern Approach*. Series in Artificial Intelligence. Prentice Hall, third edition.

Shin, K. and Han, S. (2011). A centroid-based heuristic algorithm for the capacitated vehicle routing problem. *Computing and Informatics*, 30:721–732.