XXVIII Congresso {virtual} de Iniciação Científica da Unicamp 2020

Algoritmos para Problemas de Roteamento de Veículos com Incertezas

Aluno: Felipe Lopes de Mello
Orientador: Eduardo C. Xavier
Vigência do projeto: 01/08/2019 - 31/08/2020

1 Introdução

Um dos problemas mais conhecidos da ciência da computação é o problema de roteamento de veículos (em inglês *Vehicle Routing Problem* (VRP)). Proposto inicialmente por Dantzig e Ramser [1], o VRP consiste em encontrar m rotas de veículos disjuntas que iniciam e terminam em um depósito, atendendo n clientes e minimizando o custo total das rotas. Esta minimização pode ser tanto em relação ao tempo total do deslocamento, à distância percorrida, o gasto total, entre outros. Além disso, o VRP é geralmente acompanhado de restrições, como por exemplo cada veículo possuir uma capacidade máxima ou então clientes que devem ser atendidos dentro de um prazo limite.

Uma vez que este problema possui uma grande importância na prática (em razão de poder ser utilizado em vários problemas de logística), décadas de estudos sobre este tema já foram realizados ao longo do tempo [3]. Este projeto de iniciação científica tem como foco um problema que generaliza o VRP, chamado de problema de roteamento de veículos estocástico (em inglês *Stochastic Vehicle Routing Problem* (SVRP)). No SVRP, além do objetivo e das restrições dadas pelo VRP, cada um dos n clientes possuem presença e demanda dadas por uma distribuição de probabilidade conhecida.

Podemos definir o Problema de Roteamento de Veículos Estocástico da seguinte forma. Temos um conjunto $N = \{1, \ldots, n\}$ de clientes, cada um com uma demanda estocástica ξ_i e uma probabilidade de presença relacionada p_i. Para cada cliente i há uma distribuição de probabilidade discreta de demanda, que determina a probabilidade $P\{\xi_i = d\}$ de i requerer demanda d para cada possível valor de demanda d entre $[0, C]$. Além disso denotamos por μ_i a demanda média do cliente i. Temos ainda um depósito central, denotado por 0, de onde m veículos de entrega de capacidade idêntica C devem sair para realizar as entregas e retornar. Cada cliente deve ser visitado uma única vez por algum dos veículos.

Os custos de um veículo se mover entre pares de pontos podem ser modelados por um grafo onde o conjunto de vértices é $N \cup \{0\}$ e, para cada par de vértices i, j, há uma aresta $e = \{i, j\}$ com custo associado c_{ij}. O objetivo no primeiro estágio é projetar m rotas (uma para cada veículo) de custo esperado mínimo saindo e voltando para o depósito, tal que cada cliente esteja em exatamente uma rota e a demanda esperada de cada rota seja no máximo C. Associamos uma variável inteira x_e para cada aresta $e = \{i, j\}$ do grafo, de forma que $x_e = 1$ se um veículo trafega entre i e j e $x_e = 0$ caso contrário (arestas incidentes ao depósito podem ser paralelas, caso em que $x_e = 2$). Assim, considerando que $b(S) = \lceil \left(\sum_{i \in S} \mu_i\right) / C \rceil$ é um limite inferior para o número de veículos necessários para atender a demanda média dos clientes no conjunto $S \subseteq N$ e que a função de segundo estágio $D(x)$ representa o custo esperado das rotas planejadas, obtemos a seguinte formulação PLI do problema equivalente ao programa estocástico de dois estágios:

$$\min D(x) \tag{1}$$

$$\text{s.a.} \sum_{e=\{0,j\} \in E} x_e = 2m \tag{2}$$

$$\sum_{e=\{i,j\} \in E} x_e = 2 \qquad \forall\, i \in N \tag{3}$$

$$\sum_{\substack{e=\{i,j\} \\ i \in S, j \notin S}} x_e \geq 2b(S) \qquad \forall\, S \subseteq N, |S| > 1 \tag{4}$$

$$x_e \in \{0, 1\} \qquad \forall\, e = \{i, j\} \in E,\ i,\ j \neq 0 \tag{5}$$

$$x_e \in \{0, 1, 2\} \qquad \forall\, e = \{0, j\} \in E \tag{6}$$

Nesse modelo, a restrição (2) garante que as m rotas começam e terminam no depósito (vértice 0) e as restrições (3) que cada cliente faça parte de uma rota com duas arestas incidindo sobre ele. As restrições (4) são uma generalização das restrições de eliminação de subciclo do TSP. Dado um subconjunto $S \subseteq N$ sabemos que pelo menos $b(S)$ veículos são necessários para cobrir os vértices em S. Logo, no corte definido por S, pelo menos $2b(S)$ arestas devem estar presentes em uma solução válida.

Além disso, por se tratar de uma heurística e não um algoritmo aproximado, não apenas podem não ser a solução ótima como também não sabemos o quão próximo do valor ótimo estão. Sendo assim decidimos estudar o desenvolvimento de um algoritmo exato para o SVRP. Para isso, o *Método L-Shaped Inteiro*, inicialmente proposto por Laporte e Louveaux1998 [4], foi implementado e executado para o mesmo conjunto de instâncias. De forma simplificada, o *Método L-Shaped Inteiro* é um algoritmo de *Branch and Cut* aplicado na *relaxação* do problema. Por *relaxação* entendemos que as restrições de eliminação de subciclos e as restrições de integralidade são temporariamente removidas do problema, sendo reinseridas após uma solução ser encontrada.

1.1 Algoritmo Heurístico

Resumidamente, o modelo geral de uma heurística tabu consiste em começar com uma solução inicial, seja ela viável ou não, computar o valor de soluções vizinhas (soluções similares à inicial) e continuar o procedimento com a melhor entre essas. A alteração que gerou a melhor das soluções vizinhas é adicionada a uma lista, chamada de lista tabu, na qual movimentos são proibidos de serem realizados para evitar que as soluções trabalhadas fiquem estagnadas em um ciclo. A exceção é quando o movimento atende ao critério de aspiração, que geralmente é quando a solução encontrada é melhor do que a melhor solução global conhecida.

Ademais, o método para quando um certo número de iterações é atingido, seja ele um número constante ou dependente da última iteração na qual a melhor solução foi atualizada. Outro ponto importante de uma heurística tabu é a atribuição de penalidades para soluções inviáveis com o objetivo do algoritmo focar em soluções viáveis. Entretanto, a penalidade pode começar com valores baixos para não perdemos variabilidade de soluções. Para o problema do SVRP a penalidade na v-ésima iteração pode ser descrita como:

$$F(x^v) = D(x^v) + \rho|m^v - m| \tag{7}$$

1.2 Algoritmo exato

Em nosso contexto, o *Método L-Shaped Inteiro* se trata de um algoritmo *branch-and-cut*, ou seja, um método que realiza uma árvore de busca com o objetivo de se encontrar a solução ótima, visitando um nó por vez - chamado *problema atual* (PA) - e então decidindo se deve ramificar - criar novos nós, cada um com um cenário diferente - ou se deve *cortar* o nó. Inicialmente, no PA, o SVRP é *relaxado* de três maneiras: as restrições de integralidade (5)-(6) são relaxadas; as restrições de eliminação de subciclo e capacidade do veículo (4) são relaxadas; e $D(x)$ (custo do segundo estágio) é substituído por um limite inferior da função objetivo. O PA é então resolvido (por meio de um solucionador de otimização para programação linear) e iniciamos uma introdução gradual de condições de integralidade por meio do processo de ramificação e pela geração de restrições (4) à medida que são violadas. Além disso, temos um limite inferior para o custo esperado θ chamado de *corte de otimalidade*.

2 Resultados e Discussão

As instâncias seguem a proposta de Gendreau et al. [2] para cada instância, n vértices são gerados e posicionados em um espaço 100×100 de acordo com uma distribuição uniforme e distância entre dois vértices é definida como a distância euclidiana. Todos os clientes possuem presença estocástica com probabilidade de presença uniformemente distribuída no intervalo $(0, 1)$. Além disso, a demanda de cada cliente foi primeiramente atribuída à um intervalo $[1, 9]$, $[5, 15]$ ou $[10, 20]$, onde cada intervalo possui chances iguais de ser

escolhido. O valor da demanda real do cliente foi então selecionado de acordo com uma distribuição uniforme discreta neste intervalo. Finalmente, a capacidade C de cada veículo são iguais e dependem de um coeficiente de *enchimento* cuja fórmula é descrita a seguir:

$$\bar{f} = \frac{5n}{mC}$$

Como n e m variam de instância para instância, note que quanto maior o valor de f, menor será o valor da capacidade. Essa fórmula derivada da descrita em Gendreau et al. [2] é uma boa tática para entendermos como o valor da demanda esperada dos clientes em relação à capacidade impacta no custo e no tempo de execução de cada solução obtida.

Os resultados do *Método L-Shaped Inteiro* pode ser examinados nas Tabelas 1 e 2. Um fator em comum em todas estas tabelas é que apenas as mesmas 4 instâncias foram utilizadas (com n de 6 à 9 e $m = 2$). Como pode ser visto na Tabela 2, os tempos de execução crescem ainda mais rapidamente do que para o algoritmos de busca tabu. Para $n = 9$, o tempo de execução passou de 20 minutos, o que tornou impraticável o estudo de instâncias maiores.

Tabela 1: Custo da solução encontrada pelo algoritmo exato sem o critério guloso

Nº clientes incertos	1			(n - 1) / 2			n - 1		
n f	0,25	0,50	0,75	0,25	0,50	0,75	0,25	0,50	0,75
6	261,51	298,01	323,26	248,13	281,91	320,39	144,59	144,96	149,46
7	282,50	311,54	387,11	210,72	270,39	323,18	160,18	160,27	167,57
8	296,48	354,07	444,50	261,208	313,50	396,07	191,47	192,127	217,71
9	306,36	340,18	473,99	262,44	294,91	294,91	204,00	204,33	221,98

Tabela 2: Tempo de execução (em segundos) do algoritmo exato sem o critério guloso.

Nº clientes incertos	1			(n - 1) / 2			n - 1		
n f	0,25	0,50	0,75	0,25	0,50	0,75	0,25	0,50	0,75
6	0,06	0,05	0,05	0,10	0,12	0,11	0,09	0,11	0,11
7	0,44	0,46	0,46	0,38	0,38	0,34	0,39	0,37	0,42
8	8,89	8,51	8,72	8,27	8,56	8,67	8,83	8,81	8,79
9	1.273,61	1.267,34	1.251,65	1247,08	1.256,54	1.251,14	1.256,79	1238,11	1.245,31

Apesar do número limitado de vértices que as instâncias poderiam ter, ainda sim é possível comparar as soluções obtidas pelo algoritmo de busca tabu com o algoritmo exato, bem como seus tempos de execução.

Executando o algoritmo heurístico para o mesmo conjunto de instâncias analisado anteriormente, temos os resultados demonstrados nas Tabelas 3 e 4.

Tabela 3: Custo da solução encontrada pelo algoritmo heurístico para as mesmas instâncias do algoritmo exato.

Nº clientes incertos	1			(n - 1) / 2			n - 1		
n f	0,25	0,50	0,75	0,25	0,50	0,75	0,25	0,50	0,75
6	277,59	299,38	333,86	254,36	281,91	323,99	148,54	145,02	149,54
7	303,90	325,66	387,91	242,28	272,01	324,47	161,26	161,78	168,04
8	316,30	358,17	449,22	288,75	320,52	397,56	194,98	193,88	217,79
9	320,75	348,39	477,96	275,49	298,77	406,12	210,41	205,55	222,67

Tabela 4: Tempo de execução (em segundos) do algoritmo heurístico para as mesmas instâncias do algoritmo exato.

Nº clientes incertos	1			(n - 1) / 2			n - 1		
n f	0,25	0,50	0,75	0,25	0,50	0,75	0,25	0,50	0,75
6	0,06	0,07	0,07	0,07	0,07	0,07	0,07	0,07	0,07
7	0,12	0,14	0,16	0,12	0,15	0,15	0,12	0,13	0,14
8	0,26	0,30	0,29	0,24	0,31	0,32	0,24	0,24	0,28
9	0,44	0,52	0,51	0,44	0,48	0,56	0,41	0,42	0,45

Como o algoritmo de busca tabu não possui o critério guloso, devemos comparar estes resultados com as Tabelas 1 e 2. Nesta comparação, constatamos que, embora a solução ótima não seja encontrada em muitos casos, a solução obtida pelo algoritmo de busca tabu definitivamente foi, em todos os casos, muito próxima do ótimo. Somando isso à drástica redução de tempo de execução, podemos afirmar que o algoritmo heurístico projetado deve definitivamente ser considerado para uso na prática.

Referências Bibliográficas

[1] G. B. Dantzig and J. H. Ramser. The truck dispatching problem. *Manage. Sci.*, 6(1):80–91, October 1959.

[2] Michel Gendreau, Gilbert Laporte, and René Séguin. A tabu search heuristic for the vehicle routing problem with stochastic demands and customers. *Operations Research*, 44(3):469–477, 1996.

[3] Gilbert Laporte. Fifty years of vehicle routing. *Transportation Science*, 43(4):408–416, 2009.

[4] Gilbert Laporte and François Louveaux. Solving stochastic routing problems with the integer l-shaped method. 01 1998.