Algoritmos para o Problema de Roteamento de Veículos com Incertezas

Orientador:

Eduardo Candido Xavier

Aluno:

Eduardo Barros Innarelli

1. Introdução

Neste trabalho estudamos um problema de roteamento de veículos estocástico no qual os clientes e suas demandas são incertos. No problema clássico, o VRP, temos um depósito de onde m caminhões devem sair para fazer entregas para n clientes. Deve-se construir m rotas disjuntas nos clientes iniciando e terminando no depósito, tal que cada cliente seja visitado exatamente uma vez, com o objetivo de minimizar o custo total. Na versão estocástica do problema, alguns clientes são já conhecidos e para um outro conjunto de clientes temos apenas suas probabilidades de ocorrência, o que significa que devem ser atendidos por alguma rota caso sua ocorrência se realize. Nessa versão as demandas dos clientes não são fixas, podendo variar conforme uma distribuição probabilística. Neste caso, o objetivo é computar m rotas de custo *esperado* mínimo tal que a demanda de todos clientes sejam atendidas. O problema clássico de roteamento de veículos é NP-Difícil e a versão estocástica generaliza este com a inclusão de clientes e demandas incertos. Assumindo a hipótese de que P $\neq$ NP, sabemos que não existem algoritmos polinomiais para resolver estes problemas, mas há uma série de técnicas que podem ser aplicadas tentando encontrar boas soluções.

Existe uma diferença marcante entre as formulações de VRP determinísticos e estocásticos: para todas as variantes destes, o tomador de decisão deve decidir a solução (ao menos parcialmente) antes que os valores exatos de todos os parâmetros sejam completamente conhecidos. Em algumas situações, as restrições podem ser violadas quando os valores reais dos parâmetros são realimentados. No SVRP, por exemplo, a demanda total realizada de uma rota planejada pode exceder a capacidade do veículo, caso em que a rota "falha" quando é exercida com os dados obtidos.

Uma forma comum de modelar um problema estocástico é como um programa estocástico com recurso, do inglês *Stochastic Program with Recourse* (SRP). Em um SRP, permite-se falhas, mas o tomador de decisão deve definir uma política corretiva (ou recurso) descrevendo quais ações tomar para reparar a solução após uma falha. Neste caso os problemas são considerados de 2 ou mais estágios. Em um primeiro estágio é gerada uma solução com base nas informações determinísticas do problema, e no segundo, após serem realizados os eventos aleatórios, são aplicadas as políticas de recurso, que podem incorrer em novos custos. No caso de um problema de roteamento de veículos estocástico formulado como um SRP o objetivo é otimizar o custo de transporte esperado, que consiste no custo da viagem planejada mais o custo esperado das políticas de recurso levando em conta todas as possíveis realizações.

2. Soluções

O Problema de Roteamento de Veículos Estocástico pode ser formalmente definido 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. 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 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:

$$\min\ D(x) \tag{2.1}$$

$$\text{s.a.} \sum_{e=\{0,j\}\in E} x_e = 2m \tag{2.2}$$

$$\sum_{e=\{i,j\}\in E} x_e = 2 \qquad \forall\ i \in N \tag{2.3}$$

$$\sum_{\substack{e=\{i,j\}\\ i\in S, j\notin S}} x_e \geq 2 \qquad \forall\ S \subseteq N, |S| > 1 \tag{2.4}$$

$$x_e \in \{0, 1\} \qquad \forall\ e = \{i, j\} \in E,\ i,\ j \neq 0 \tag{2.5}$$

$$x_e \in \{0, 1, 2\} \qquad \forall\ e = \{0, j\} \in E \tag{2.6}$$

Nesse modelo, a restrição (2.2) garante que as m rotas começam e terminam no depósito (vértice 0) e as restrições (2.3) que cada cliente faça parte de uma rota com duas arestas incidindo sobre ele. As restrições (2.4) eliminam subciclos como no problema do caixeiro viajante: pelo menos 2 arestas devem estar presentes no corte definido pelo subconjunto $S \subseteq N$.

O cálculo de $D(x)$, dada uma solução de primeiro estágio x, é separável em relação as rotas. Seja $D^{k,\gamma}$ o custo esperado de segundo estágio da rota k na orientação γ. O custo esperado total de segundo estágio é computado por:

$$D(x) = \sum_{k=1}^{m} \min\{D^{k,1}, D^{k,2}\} \tag{2.7}$$

Para calcular $D^{k,\gamma}$ adaptamos a fórmula de custo apresentada por Bertsimas [1], cuja complexidade computacional, sendo $\bar{l}$ o nº máximo de demandas que um cliente pode ter, é da ordem de $O(n^2 + \bar{l}nC)$, conforme demonstrado em Gendreau et al. [2]. Quanto as políticas de recurso adotadas para uma rota, estamos considerando a estratégia **b** também proposta por Bertsimas [1], que consiste em pular os clientes não presentes na realização em particular (assumindo que tal informação é conhecida antes do percurso iniciar) e retornar ao depósito sempre que a capacidade do veículo for alcançada, continuando a rota no próximo cliente, ou excedida, retornando ao cliente no qual a falha ocorreu.

2.1. Método L-Shaped Inteiro

Para resolver o problema de forma exata, escolhemos implementar o algoritmo L-Shaped Inteiro inicialmente proposto por Laporte e Louveaux [3]. Nele, um procedimento *Branch-and-Cut* é aplicado sob uma relaxação do problema em que o termo $D(x)$ é limitado superiormente por uma variável θ.

Em alto nível, sempre que uma solução inteira viável da relaxação é encontrada e o valor de θ^v na solução violar $\theta \geq D(x^v)$, o corte de optimalidade $\theta \geq D(x^v)(\sum_{e\in E^v} x_e - n + m + 2)$, $E^v = \{e = (i, j) | i, j > 0 \text{ e } x_e^v = 1\}$ é gerado, conforme proposto por Gendreau et al. [2]. A desigualdade é válida pois:

$$\begin{aligned}
&\sum_{e\in E^v} x_e \leq \sum_{e\in E^v} x_e^v - 1 \\
\Rightarrow &\sum_{e\in E^v} x_e \leq n - m - 2 \\
\Rightarrow &\sum_{e\in E^v} x_e - n + m + 2 \leq 0 \\
\Rightarrow &(\sum_{e\in E^v} x_e - n + m + 2) D(x^v) \leq \theta
\end{aligned}$$

Onde $\sum_{e\in E^v} x_e^v = n - 1 - m$ devido a inexistência de subciclos entre clientes.

2.2. Heurística Tabu

Como no problema estudado, o SVRP, temos que a presença de um cliente e sua demanda são incertos, o número de possíveis cenários distintos é dependente tanto do número de clientes quanto de suas possíveis demandas. É comum que as instâncias usualmente estudadas na literatura possuam cerca de 50 clientes, cada um com 10 possíveis demandas. Nesses casos, é notável que computar as m melhores rotas para todos os cenários possíveis é computacionalmente inviável, uma vez que o cálculo de apenas um cenário possui complexidade exponencial. Visando contornar essa dificuldade, adotamos uma abordagem heurística baseada na busca tabu para o SVRP, conforme proposto por Gendreau et al. [4]. Heurísticas são regularmente desenvolvidas e aprimoradas para problemas de complexidade exponencial pois, ainda que sem nenhuma garantia teórica, costumam retornar boas soluções em tempos razoáveis.

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 (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 termina 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.

3. Implementação e Resultados Computacionais

3.1. Detalhes de Implementação

As técnicas estudadas foram implementadas na linguagem de programação C++, tanto por questões de performance quanto por essa ser uma boa oportunidade para praticar mais a linguagem. O primeiro método escrito foi o custo do segundo estágio detalhado em **??**, que por si só revelou-se um desafio em termos de armazenamento e manipulação de estruturas de dados. Nesse processo, foi definida uma classe de grafos utilizada ao longo de todo o trabalho para tratar e salvar as instâncias.

Resolvido o segundo estágio, partimos para a implementação da Heurística Tabu, descrita na seção 2.2, e do método L-Shaped, discutido em 2.1. Para o L-Shaped, foi estudado e utilizado o solucionador comercial de otimização matemática **Gurobi**. Assim, bastaria definir as restrições e uma callback que checa se uma solução inteira foi encontrada e, se sim, impor um corte de optimalidade ao modelo caso a solução violar $\theta \geq D(x^v)$.

O maior empecilho com o qual esbarramos no desenvolvimento do método exato foi o número exponencial de restrições de eliminação de subciclos (3.2), que torna inviável a alocação de todas em memória. Não a toa, restrições dessa natureza costumam ser incluídas dinamicamente, assim como fazemos com os cortes de optimalidade. Para tal, é preciso resolver um problema de separação que verifica em tempo de execução se uma solução violou ou não alguma(s) dessas restrições. Acontece que estudar o problema de separação para soluções **fracionárias** do SVRP - o que seria o ideal tendo em vista a quantidade exponencial de restrições (3.2) - estava fora do nosso escopo técnico, teórico e de tempo.

Como forma de mitigar essa adversidade, resolvemos solucionar o problema de separação para soluções **inteiras** - i.e., simplesmente checar se alguma componente conexa da solução forma um subciclo. Mesmo que essa saída comprometesse a performance do algoritmo, esperávamos obter resultados para instâncias pequenas que permitissem alguma comparação prática entre os dois métodos estudados no projeto.

3.2. Instâncias

O software recebe, como entrada, o número de vértices $n > 1$, de veículos m entre $[1, n-1]$, de clientes incertos x entre $[0, n-1]$ e um coeficiente de preenchimento $\bar{f}$ entre $(0, 1]$, definido mais a frente. Para gerar uma instância, o modelo descrito em Gendreau et al. [4] foi adaptado:

- os n vértices são posicionados em um espaço 100×100, de acordo com uma distribuição uniforme, tal que a distância entre dois vértices é definida como a distância euclidiana;
- os x clientes incertos possuem presença estocástica com probabilidade de presença uniformemente distribuída no intervalo $(0, 1)$;
- a demanda de todo cliente é atribuída a um valor do conjunto $\{5, 10, 15\}$, de acordo com uma distribuição uniforme discreta;
- as capacidades C dos veículos são iguais e dependem do coeficiente de preenchimento $\bar{f} = \frac{5n}{mC}$.

Quanto aos testes, as instâncias foram geradas iterando n unitariamente para $6 \leq n \leq 10$, de 2-em-2 para $10 < n \leq 20$ e de 5-em-5 para $n > 20$, até o momento em que o algoritmo estoura o limite imposto de 20 minutos. Para cada n testado, o número de clientes incertos x variou entre $\{1, \frac{n-1}{2}, n-1\}$ e, para cada x, o coeficiente $\bar{f}$ variou entre $\{0.25, 0.5, 0.75\}$. O número de veículos foi fixado em $m = 2$, dado que nosso principal interesse é analisar como os métodos se comportam com o crescimento de n e da incerteza.

3.3. Resultados

A Heurística Tabu foi rodada 10 vezes para cada instância com $n \leq 20$ e 3 vezes para instâncias com $n > 20$, visto que as decisões aleatórias do algoritmo podem culminar em resultados distintos para uma mesma entrada. Para avaliar a qualidade das soluções obtidas pela heurística é preciso verificar o quão longe elas estão do ótimo. O método exato L-Shaped, por conta das limitações discutidas na seção 3.1, somente encontrou a solução ótima até $n = 9$, estourando o limite de tempo a partir de então. No entanto, como demonstrado na **Tabela 1**, tais soluções são suficientes para indicar que a Busca Tabu alcançou bons resultados, devido a distância pequena entre os custos retornados por ambos os métodos.

A **Figura 1** mostra as rotas planejadas que os dois algoritmos encontraram para uma instância com 6 vértices, 2 clientes incertos e coeficiente de preenchimento $\bar{f} = 0.5$. O depósito é identificado em vermelho e os clientes incertos em verde. Repare que os percursos se cruzam na solução, o que parece contraintuitivo a priori. Todavia, faz sentido se lembrarmos que os diferentes cenários possíveis são levados em conta no cálculo do custo esperado e que, em parte desses cenários, os clientes incertos são ignorados nas rotas planejadas por não estarem presentes, ocasionando rotas muito menores.

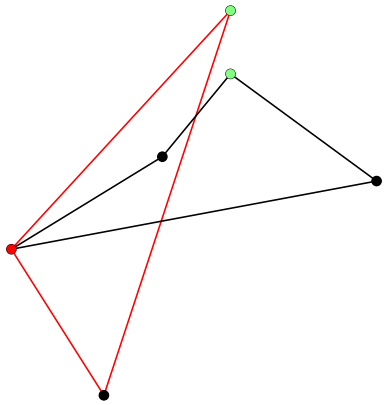

Figura 1: Solução Ótima para um Grafo com 6 Vértices, 2 Clientes Incertos e $\bar{f} = 0.5$.

Referências

[1] D. Bertsimas, "A vehicle routing problem with stochastic demand," *Operations Research*, vol. 40, pp. 574–585, 06 1992.

[2] M. Gendreau, G. Laporte, and R. Séguin, "An exact algorithm for the vehicle routing problem with stochastic demands and customers," *Transportation Science*, vol. 29, pp. 143–155, May 1995.

[3] G. Laporte and F. V. Louveaux, "The integer l-shaped method for stochastic integer programs with complete recourse," *Oper. Res. Lett.*, vol. 13, pp. 133–142, Apr. 1993.

[4] M. Gendreau, G. Laporte, and R. Séguin, "A tabu search heuristic for the vehicle routing problem with stochastic demands and customers," *Operations Research*, vol. 44, no. 3, pp. 469–477, 1996.

		1 cliente incerto			(n-1)/2 clientes incertos			n-1 clientes incertos		
n		**f = 0.25**	**f = 0.5**	**f = 0.75**	**f = 0.25**	**f = 0.5**	**f = 0.75**	**f = 0.25**	**f = 0.5**	**f = 0.75**
6 (tabu)	tempo (s)	0,06	0,07	0,07	0,07	0,07	0,07	0,07	0,07	0,07
	custo (m)	277,59	299,38	333,86	254,36	281,91	323,99	148,54	145,02	149,54
6 (exato)	tempo (s)	0,06	0,05	0,05	0,10	0,12	0,11	0,09	0,11	0,11
	custo (m)	261,51	298,01	323,27	248,13	281,91	320,39	144,59	144,96	149,46
	Distância da heurística para o ótimo (%)	6,15	0,46	3,28	2,51	0,00	1,12	2,74	0,04	0,06
7 (tabu)	tempo (s)	0,12	0,14	0,16	0,12	0,15	0,15	0,12	0,13	0,14
	custo (m)	303,90	325,66	387,91	242,28	272,01	324,47	161,26	161,78	168,04
7 (exato)	tempo (s)	0,44	0,46	0,46	0,38	0,38	0,34	0,39	0,37	0,42
	custo (m)	282,50	311,54	387,11	210,72	270,39	323,18	160,18	160,27	167,57
	Distância da heurística para o ótimo (%)	7,58	4,53	0,21	14,98	0,60	0,40	0,67	0,94	0,28
8 (tabu)	tempo (s)	0,26	0,30	0,29	0,24	0,31	0,32	0,24	0,24	0,28
	custo (m)	316,30	358,17	449,22	288,75	320,52	397,56	194,98	193,88	217,79
8 (exato)	tempo (s)	8,89	8,51	8,72	8,27	8,56	8,67	8,83	8,81	8,79
	custo (m)	296,48	354,07	444,50	261,21	313,50	396,07	191,47	192,13	217,71
	Distância da heurística para o ótimo (%)	6,68	1,16	1,06	10,54	2,24	0,37	1,84	0,91	0,04
9 (tabu)	tempo (s)	0,44	0,52	0,51	0,44	0,48	0,56	0,41	0,42	0,45
	custo (m)	320,75	348,39	477,96	275,49	298,77	406,12	210,41	205,55	222,67
9 (exato)	tempo (s)	1273,61	1267,34	1251,65	1247,08	1256,54	1230,06	1256,79	1238,11	1245,31
	custo (m)	306,36	340,18	473,99	262,44	294,91	404,32	204,00	204,33	221,98
	Distância da heurística para o ótimo (%)	4,70	2,41	0,84	4,97	1,31	0,44	3,14	0,60	0,31

Tabela 1: Comparação entre Resultados da Heurística Tabu e do Algoritmo Exato L-Shaped para o SVRP