Avaliação de resolvedores para problemas de localização de facilidades com capacidade limitada

Guilherme Akira Demenech Mori

May 26, 2023

Abstract

Contents

List of Tables

1 Introdução

2 Atividades desenvolvidas

Os estudos teóricos foram guiados pelos capítulos de Otimização Linear e Discreta de *Pesquisa operacional* [Arenales et al., 2011]. Para os testes computacionais foi utilizada a linguagem de *script Python* [Python Software Foundation, 2023], sendo feita leitura de instâncias e chamada dos resolvedores escolhidos.

1. **Estudo de modelos de otimização linear**: embora a modelagem linear seja bastante restrita, ela pode representar uma grande variedade de problemas de maneira exata ou como aproximação. As restrições desses modelos facilitam sua representação (como sistemas lineares ou matrizes) e oportunizam aos métodos de solução atalhos e generalizações eficientes e bem direcionados.

2. **Estudo de solução gráfica e método primal simplex**: para modelos com somente duas variáveis, a representação bidimensional permite resolução rápida e intuitiva dos problemas de otimização linear, basta identificar o gradiente da função objetivo e a região factível, encontrando a curva de nível (neste caso, o hiperplano perpendicular ao gradiente) no limite da factibilidade no sentido de minimização/maximização. O método simplex aborda o problema considerando que, se houver solução ótima, haverá ao menos um vértice ótimo. Partindo de uma partição básica factível inicial, segue pelas arestas que mais tentem à otimalidade (seja menor ou maior), observando se houver solução e se ela for limitada.

3. **Estudo de modelos de otimização inteira**: muitos problemas precisam de variáveis binárias ou discretas, geralmente representando escolhas e unidades indivisíveis. Essa maior restrição permite mais adequadamente a modelagem de relações lógicos e funções lineares por partes.

4. **Estudo do método *branch-and-bound***: ramificar e limitar descreve o processo de selecionar variáveis para dividir a região factível e buscar limitantes mais precisos. São aplicadas múltiplas vezes o método simplex (relaxando linearmente as variáveis inteiras) para obter vértices lineares mais próximos da solução ótima mista, descartando as partes da região em que o limitante não permita haver solução melhor que o melhor limitante conhecido naquele passo.

5. **Estudo da configuração padrão e dos recursos disponíveis de cada pacote computacional escolhido**: foram escolhidos dois resolvedores comerciais (*CPLEX* [IBM,] e *Gurobi* [Gurobi Optimization,]) e dois de

código-aberto (*Cbc* [Forrest and Lougee-Heimer, 2005] e *SCIP* [Zuse Institute Berlin, 2023]). Todos os resolvedores escolhidos aceitam modelos de otimização linear e inteira mista e aplicam de cortes. A interface de linha de comando deles pode ser acessada dentro do ambiente *Python* pela biblioteca *PuLP* [Roy et al., 2022].

6. **Implementação de um modelo clássico de otimização utilizando os pacotes computacionais escolhidos**: o problema de localização de facilidades com capacidade limitada (CFLP) minimizar o custo de transporte e de abertura de facilidades (como armazéns ou fábricas), respeitando suas capacidades pré-estabelecidas. Dados os clientes e as demandas que devem ser satisfeitas, designa-se as facilidades que devem ser abertas, que os devem atender e quanto devem entregar. Como várias facilidades podem entregar produtos para um mesmo cliente, chama-se a versão geral do problema de múltiplas fontes (MS-CFLP). A variação de fonte única (SS-CFLP) do problema adiciona a restrição de fornecedores por cliente: somente uma facilidade deve atender cada demanda.

7. **Testes computacionais e avaliação comparativa dos resultados**: o *script* desenvolvido lê as instâncias de *benchmark* utilizadas por outros estudos do MS-CFLP e SS-CFLP, monta o modelo misto e o executa em cada resolvedor com diferentes limites de tempo (2, 5 e 10 minutos para instâncias pequenas e 10, 30 e 60 para as grandes). Os resultados registrados nos arquivos de *log* pelos resolvedores *CPLEX*, *Gurobi* e *Cbc* são extraídos com auxílio da biblioteca *Orloge* [Peschiera, 2021] (que não oferece suporte para a saída do *SCIP*) e tabelados.

3 Revisão da literatura

4 Definição e modelagem dos problemas

No problema de localização de facilidades com capacidade limitada (*Capacitated Facility Location Problem*, CFLP) são minimizados os custos de instalação de facilidades e de designação de clientes a elas, de forma a respeitar as limitações de capacidade das facilidades e a satisfazer as demandas dos clientes. Se não forem definidas outras restrições além da capacidade das facilidades e da demanda dos clientes, nada impede que a demanda de um cliente seja satisfeita designando mais do que uma facilidade para ele, o que caracteriza o CFLP com múltiplas fontes (*multi-source*, MS-CFLP) [4.1].

Quando cada demanda deve ser atendida por somente uma facilidade, tem-se o CFLP com fonte única (*single-source*, SS-CFLP) [4.2].

Quando houver clientes que não possam ser atendidos por uma mesma facilidade, tem-se o CFLP com incompatibilidade de clientes (*customer incompatibilities*, CFLP-CI) [4.3].

4.1 MS-CFLP: problema de localização de facilidades com capacidade limitada e múltiplas fontes

São dados o conjunto de facilidades I e o conjunto de clientes J. Para cada facilidade $i \in I$, tem-se o custo de abertura $f_i \in \mathbb{R}$ e a capacidade máxima de atendimento $s_i \in \mathbb{Z}$. Tem-se também a demanda de produtos $d_j \in \mathbb{Z}$ para cada cliente $j \in J$. Para cada par de facilidade e cliente $\langle i, j \rangle \in I \times J$, há o custo $c_{ij} \in \mathbb{R}$ do atendimento de j por i por unidade do produto demandado.

O modelo pode ser descrito da seguinte forma:

$$\min \sum_{i \in I} (f_i y_i + \sum_{j \in J} c_{ij} x_{ij}) \tag{1}$$

sujeito a

$$\sum_{i \in I} x_{ij} \geq d_j \quad \forall j \in J \tag{2}$$

$$\sum_{j \in J} x_{ij} \leq y_i s_i \quad \forall i \in I \tag{3}$$

$$y_i \in \{0, 1\},\ x_{ij} \geq 0,\ x_{ij} \in \mathbb{Z} \quad \forall i \in I,\ j \in J \tag{4}$$

O atendimento da demanda do cliente $j \in J$ pela facilidade $i \in I$ é representado pela variável inteira não-negativa x_{ij}, cujo valor é a quantidade de unidades do produto são fornecidas por i para j. A restrição 2 garante que a soma de todos os atendimentos de cada cliente por todas as facilidades satisfaça sua demanda d_j. Os custos de atendimento são somados na função objetivo (1) de acordo com o custo c_{ij} por unidade.

A abertura da facilidade $i \in I$ é representada pela variável binária y_i, que assume valor 0 quando i estiver fechada e 1 se tiver sido instalada. A restrição 3 exige que, se a soma dos atendimentos prestados por i para todos os seus clientes for positiva, a facilidade deva ser aberta ($y_i = 1$). Essa restrição também limita a soma dos atendimentos à capacidade s_i da facilidade. O custo de instalação f_i da facilidade será somado ao custo na função objetivo (1) se $y_i = 1$.

Observação 1 *É importante ressaltar que esse modelo considera que o produto possui unidades indivisíveis e que, portanto, os atendimentos x_{ij}, as capacidades s_i e as demandas d_j são inteiros.*

Caso fossem fornecidas capacidades reais $s'_i \in \mathbb{R}$, bastaria utilizar $s_i = \lfloor s'_i \rfloor$. Se fossem demandas reais $d'_j \in \mathbb{R}$, bastaria utilizar $d_j = \lceil d'_j \rceil$.

Contudo, sendo necessário utilizar valores fracionários de atendimento, capacidade e demanda, bastará alterar a restrição 4 para que sejam reais as variáveis $x_{ij} \in \mathbb{R}$. A modelagem do problema funciona para atendimentos no domínio dos reais.

4.2 SS-CFLP: problema de localização de facilidades com capacidade limitada e fonte única

A modelagem pode ser essencialmente a mesma do MS-CFLP [4.1], simplesmente se alterando o domínio de cada $x_{ij} \in \{0, d_j\}$ na restrição 4. Porém a restrição para uma única fonte permite algumas alterações razoáveis.

Novamente são dados o conjunto de facilidades I e o conjunto de clientes J, bem como o custo de abertura $f_i \in \mathbb{R}$ e a capacidade máxima de atendimento $s_i \in \mathbb{Z}$ para cada facilidade $i \in I$. Então, para cada par de facilidade e cliente $\langle i, j\rangle \in I \times J$, há a demanda $p_{ij} \in \mathbb{Z}$ (quantas unidades do produto i deveria fornecer a j caso o atenda) e o custo $g_{ij} \in \mathbb{R}$ de atendimento de toda essa demanda. A possibilidade de a demanda ser diferente se for atendida por facilidades diferentes é uma generalização razoável permitida pela restrição maior das fontes de atendimento.

$$\min \sum_{i \in I} (f_i y_i + \sum_{j \in J} g_{ij} x_{ij}) \tag{5}$$

sujeito a

$$\sum_{i \in I} x_{ij} = 1 \quad \forall j \in J \tag{6}$$

$$\sum_{j \in J} p_{ij} x_{ij} \leq y_i s_i \quad \forall i \in I \tag{7}$$

$$y_i \in \{0, 1\}, \ x_{ij} \in \{0, 1\}, \quad \forall i \in I, \ j \in J \tag{8}$$

Dessa vez, x_{ij} é definida como variável binária, representando se a facilidade i atende o cliente j. A restrição 6 garante que exatamente uma facilidade atenda cada cliente. Os custos de atendimento são somados na função objetivo (5) de acordo com o custo total g_{ij}.

Novamente, y_i é a variável binária que representa se a facilidade $i \in I$ está aberta. A restrição 7 exige que, se a soma dos atendimentos prestados por i para todos os seus clientes for positiva, a facilidade deva ser instalada ($y_i = 1$). Essa restrição também limita a soma dos atendimentos (considerando a quantidade de produtos p_{ij} de cada um deles) à capacidade s_i da facilidade. Serão somados na função objetivo (5) os custos de instalação f_i das facilidades abertas ($y_i = 1$).

Observação 2 *Assim como o modelo MS-CFLP [4.1], esse modelo considera indivisibilidade de unidades do produto nas capacidades s_i e demandas p_{ij} inteiras. Mas se elas fossem reais não haveria nenhuma alteração de funcionamento, dispensando alteração de domínio das variáveis.*

Observação 3 *Embora tenha mudado a representação, não houve alteração nos dados de custo de atendimento. Pode-se substituir, neste modelo, $g_{ij} = c_{ij} p_{ij}$ e, no MS-CFLP [4.1], $g_{ij} = c_{ij} d_j$.*

Observação 4 *A relaxação linear das variáveis $x_{ij} \in \mathbb{R}$ permitiria a representação de um modelo MS-CFLP [4.1] com unidades divisíveis. Contudo, devido à variação da demanda de um mesmo cliente para cada facilidade que o possa atender, esse relaxação modelaria uma situação incomum e estranha. Dessa forma, não utilizaremos instâncias desse formato como se não tivessem a restrição de fonte única.*

4.2.1 De MS-CFLP para SS-CFLP

Para utilizar dados de uma instância do MS-CFLP [4.1] e utilizá-la SS-CFLP, basta que, para todo $i \in I$ e $j \in J$, $p_{ij} = d_j$ e $g_{ij} = c_{ij}p_{ij} = c_{ij}d_j$ (conforme a observação 3).

Os demais dados são os mesmos para ambos.

4.3 MS-CFLP-CI: problema de localização de facilidades com capacidade limitada e incompatibilidade de clientes

Algumas instâncias que foram utilizadas apresentavam uma restrição a mais. Além da função objetivo (1), da restrição de atendimento de demanda 2, de limitação de capacidade 3 e das variáveis definidas na restrição 4, é adicionado um conjunto de pares de clientes incompatíveis $\Gamma \subset J^2$. Os clientes desses pares não podem ser atendidos pela mesma facilidade.

$$x_{ij_1} \leq \lambda_{ij_1j_2}s_i, \ x_{ij_2} \leq (1 - \lambda_{ij_1j_2})s_i \quad \forall i \in I, \langle j_1, j_2 \rangle \in \Gamma \tag{9}$$

Para isso, são definidas as variáveis de disjunção $\lambda_{ij_1j_2}$

$$\lambda_{ij_1j_2} \in \{0, 1\} \quad \forall i \in I, \langle j_1, j_2 \rangle \in \Gamma \tag{10}$$

5 Experimentos computacionais

Foram escolhidos os resolvedores *Cbc* [Forrest and Lougee-Heimer, 2005], *CPLEX* [IBM,] e *Gurobi* [Gurobi Optimization,] para comparação de otimalidade, tempo e nós visitados. Os dados para comparação são extraídos dos arquivos de *log* com a biblioteca *Orloge* [Peschiera, 2021]. Como ela não oferece suporte para leitura de *log* do *SCIP* [Zuse Institute Berlin, 2023], esse resolvedor não foi incluído no estudo.

Foram utilizadas instâncias do CFLP de quatro diferentes conjuntos:

1. `sobolev` [Sobolev Institute of Mathematics, 2021]:
2. `beasley` [Beasley, 1990]:
3. `holmberg` [**?**]:
4. `mess` [Maia et al., 2022]:

As instâncias são lidas por um *script Python* [Python Software Foundation, 2023] que constrói o modelo de otimização mista da biblioteca *PuLP* [Roy et al., 2022] e executa os resolvedores pela interface de linha de comando.

Cada instância de `sobolev`, `beasley` e `holmberg` foi resolvida três vezes, com os limites de tempo de 2, 5 e 10 minutos. As instâncias de `mess` tiveram limitação maior, de 10, 30 e 60 minutos.

Os testes foram conduzidos em um computador *desktop* Dell XPS 8930 com sistema operacional Ubuntu 22.04.1 LTS 64-bit, processador Intel® Core™ i7-8700 CPU @ 3.20GHz $\times$ 12 e 16 GB de memória RAM. Os resultados estão nas tabelas 7, 7, 7, 7 e 5.

6 Referências

References

[Arenales et al., 2011] Arenales, M. N., Armentano, V. A., Morabito Neto, R., and Yanasse, H. H. (2011). *Pesquisa operacional*. Elsevier.

[Beasley, 1990] Beasley, J. E. (1990). OR-library: distributing test problems by electronic mail. *Journal of the Operational Research Society*, 41(11):1069–1072.

[Forrest and Lougee-Heimer, 2005] Forrest, J. and Lougee-Heimer, R. (2005). Cbc user guide. `https://www.coin-or.org/Cbc/`. Accessed: 2023-05-18.

[Gurobi Optimization,] Gurobi Optimization. Documentation. `https://www.gurobi.com/documentation/`. Accessed: 2023-05-18.

[IBM,] IBM. IBM CPLEX optimizer. `https://www.ibm.com/br-pt/analytics/cplex-optimizer`. Accessed: 2023-05-18.

[Maia et al., 2022] Maia, M. R., Reula, M., Parreño-Torres, C., Vuppuluri, P. P., Plastino, A., Souza, U. S., Ceschia, S., Pavone, M., and Schaerf, A. (2022). Metaheuristic techniques for the capacitated facility location problem with customer incompatibilities. *Soft Computing*, pages 1–14.

[Peschiera, 2021] Peschiera, F. (2021). pchtsp/orloge: log parser for mip and lp solvers. `https://github.com/pchtsp/orloge`. Accessed: 2023-05-18.

[Python Software Foundation, 2023] Python Software Foundation (2023). Python 3.10.11 documentation. `https://docs.python.org/3.10/`. Accessed: 2023-05-18.

[Roy et al., 2022] Roy, J., Mitchell, S., and Peschiera, F. (2022). PuLP 2.7.0. `https://pypi.org/project/PuLP/`. Accessed: 2023-05-18.

[Sobolev Institute of Mathematics, 2021] Sobolev Institute of Mathematics (2021). Discrete location problems: Benchmark library.

[Zuse Institute Berlin, 2023] Zuse Institute Berlin (2023). SCIP: Solving constraint integer programs. `https://www.scipopt.org/doc/html/`. Accessed: 2023-05-18.

7 Tabelas

Instâncias			**CBC**			**CPLEX**			**GUROBI**		
			120	**300**	**600**	**120**	**300**	**600**	**120**	**300**	**600**
`sobolev.ss`	**Gap**	**Média**	4,68	2,12	0,12	0	0	0	0,01	0	0
		Mediana	5,16	0	0	0	0	0	0	0	0
	Time	**Média**	114,48	231,07	310,80	17,76	17,77	17,70	28,70	28,87	28,76
		Mediana	119,91	275,41	274,85	14,44	14,47	14,49	19,77	19,75	19,73
	Nodes	**Média**	4383,73	8574,17	10707,35	12440,43	12440,43	12440,43	20989,02	21176,65	21176,65
		Mediana	4261	9601	10410,50	9752	9752	9752	14843	14843	14843
	#Opt		16	57	86	100	100	100	99	100	100
	#Fact		100	100	100	100	100	100	100	100	100
	#OfM		0	0	0	0	0	0	0	0	0
	#TL		0	0	0	0	0	0	0	0	0
`holmberg.ss`	**Gap**	**Média**	0,03	0,03	0,02	0	0	0	0	0	0
		Mediana	0	0	0	0	0	0	0	0	0
	Time	**Média**	7,72	12,79	19,48	0,57	0,57	0,57	0,59	0,60	0,59
		Mediana	0,36	0,36	0,35	0,14	0,14	0,14	0,13	0,13	0,13
	Nodes	**Média**	2103,03	3901,08	5994,70	718,46	718,46	718,46	747,58	747,58	747,58
		Mediana	2	2	2	0	0	0	0	0	0
	#Opt		69	69	70	71	71	71	71	71	71
	#Fact		71	71	71	71	71	71	71	71	71
	#OfM		0	0	0	0	0	0	0	0	0
	#TL		0	0	0	0	0	0	0	0	0
`beasley.small.ss`	**Gap**	**Média**	0	0	0	0	0	0	0	0	0
		Mediana	0	0	0	0	0	0	0	0	0
	Time	**Média**	0,26	0,26	0,26	0,07	0,07	0,07	0,06	0,06	0,06
		Mediana	0,17	0,17	0,17	0,04	0,04	0,04	0,04	0,04	0,04
	Nodes	**Média**	5,83	5,83	5,83	0,62	0,62	0,62	0	0	0
		Mediana	0	0	0	0	0	0	0	0	0
	#Opt		24	24	24	24	24	24	24	24	24
	#Fact		24	24	24	24	24	24	24	24	24
	#OfM		0	0	0	0	0	0	0	0	0
	#TL		0	0	0	0	0	0	0	0	0

Table 1:

Instâncias			CBC			CPLEX			GUROBI		
			120	**300**	**600**	**120**	**300**	**600**	**120**	**300**	**600**
`beasley.large.ss`	**Gap**	**Média**	80,44	80,05	55,24	4,24	1,01	0,58	5,36	1,13	0,53
		Mediana	78,50	78,06	77,94	3,13	0	0	5,27	0,41	0
	Time	**Média**	129,87	313,05	609,31	120,05	241,57	322,60	120,11	292,33	446,46
		Mediana	128,53	308,43	601,07	120,04	256,37	252,09	120,02	300,01	452,43
	Nodes	**Média**	0	1,58	128,83	77,58	1376,33	2211,25	1279,75	1858,75	3475,42
		Mediana	0	0	4,50	0	1079,50	1886	1284	1367	2813,50
	#Opt		0	0	0	0	7	9	0	3	9
	#Fact		12	12	12	12	12	12	12	12	12
	#OfM		0	0	0	0	0	0	0	0	0
	#TL		0	0	0	0	0	0	0	0	0

Table 2:

Instâncias			**CBC**			**CPLEX**			**GUROBI**		
			600	**1800**	**3600**	**600**	**1800**	**3600**	**600**	**1800**	**3600**
`mess.ss`	**Gap**	**Média**	19,53	17,34	15,14	2,40	1,32	0,92	1,72	1,25	0,82
		Mediana	29,47	21,24	9,41	1,99	0,93	0,59	1,46	1,07	0,73
	Time	**Média**	509,77	1517,66	2978,46	481,98	1333,36	2553,62	486,70	1410,30	2648,31
		Mediana	597,25	1794,17	3571,55	600,21	1800,38	3601	600,05	1800,06	3600,04
	Nodes	**Média**	4341	22587,15	65098,17	31113,92	75832,91	154435,20	29797,69	67568,38	140780,45
		Mediana	672	2127	10508	540	8167	39436	1813	6608	15406
	#Opt		2	2	2	3	3	3	3	3	3
	#Fact		7	7	7	13	11	10	13	13	11
	#OfM		0	0	0	0	2	3	0	0	1
	#TL		6	6	5	0	0	0	0	0	0

Table 3:

Instâncias			CBC 120	CBC 300	CBC 600	CPLEX 120	CPLEX 300	CPLEX 600	GUROBI 120	GUROBI 300	GUROBI 600
`holmberg.ms`	**Gap**	**Média**	0,03	0,01	0	0	0	0	0	0	0
		Mediana	0	0	0	0	0	0	0	0	0
	Time	**Média**	3,69	6,19	8,49	0,25	0,26	0,25	0,17	0,17	0,17
		Mediana	0,36	0,33	0,34	0,13	0,14	0,12	0,11	0,11	0,11
	Nodes	**Média**	188,49	354,62	583,15	72,42	72,42	72,42	10,76	10,76	10,76
		Mediana	2	2	2	0	0	0	0	0	0
	#Opt		70	70	71	71	71	71	71	71	71
	#Fact		71	71	71	71	71	71	71	71	71
	#OfM		0	0	0	0	0	0	0	0	0
	#TL		0	0	0	0	0	0	0	0	0
`beasley.small.ms`	**Gap**	**Média**	0	0	0	0	0	0	0	0	0
		Mediana	0	0	0	0	0	0	0	0	0
	Time	**Média**	0,39	0,39	0,39	0,08	0,08	0,08	0,07	0,07	0,07
		Mediana	0,11	0,11	0,11	0,06	0,06	0,06	0,04	0,04	0,04
	Nodes	**Média**	45,17	45,17	45,17	11,46	11,46	11,46	0	0	0
		Mediana	6	6	6	0	0	0	0	0	0
	#Opt		24	24	24	24	24	24	24	24	24
	#Fact		24	24	24	24	24	24	24	24	24
	#OfM		0	0	0	0	0	0	0	0	0
	#TL		0	0	0	0	0	0	0	0	0
`beasley.large.ms`	**Gap**	**Média**	82,74	82,45	69,94	1,92	0,59	0,42	5,31	1,70	0,31
		Mediana	80,94	80,61	80,50	0,32	0	0	4,01	0,75	0
	Time	**Média**	130,41	308,56	607,54	106,23	183,49	258,48	111,08	261,11	392,52
		Mediana	128,40	309,72	603,09	120,07	183,93	183,28	120,02	300,03	371,93
	Nodes	**Média**	0	0,17	22,33	580,83	2314,75	3291,58	1043,17	1182,50	1641,83
		Mediana	0	0	0	546,50	2847	2847	1262	1312,50	1546,50
	#Opt		0	0	0	5	9	9	2	2	10
	#Fact		12	12	12	12	12	12	12	12	12
	#OfM		0	0	0	0	0	0	0	0	0
	#TL		0	0	0	0	0	0	0	0	0

Table 4:

Instâncias			CBC			CPLEX			GUROBI		
			600	**1800**	**3600**	**600**	**1800**	**3600**	**600**	**1800**	**3600**
`mess.ms`	**Gap**	**Média**	10,16	8,55	6,16	0,13	0,07	0,03	0,11	0,04	0,02
		Mediana	2,57	2,07	2,19	0	0	0	0,09	0	0
	Time	**Média**	490,95	1439,28	2625,29	279,55	731,79	985,22	356,01	883,96	1104,66
		Mediana	591	1762,92	3437,64	209,53	210,01	45,26	600,10	810,70	152,61
	Nodes	**Média**	4285,33	10993,40	22013,58	10715,88	25935,38	44222,29	7694,91	21289,90	24029,50
		Mediana	1176	4239	3582,50	7133,50	15520,50	6271	2675	10121,50	14357
	#Opt		3	3	4	5	5	6	5	6	6
	#Fact		15	15	12	8	8	7	11	10	8
	#OfM		1	0	3	0	0	1	3	3	5
	#TL		0	0	0	0	0	0	0	0	0

Table 5: