O problema de projeto de redes não capacitadas com mono-produto e custos fixos: Relaxação Lagrangeana

Waldir Vieira de Oliveira Neto[1], Rhogger Freitas Silva[2], Jordania Louse Silva Alves[3], Rodrigo Francisco Borges Lourenço[4], Fabíola Medeiros Costa[5], Darlan Marques da Silva[6,7]

[1]Graduando do curso de Engenharia Mecânica, Universidade de Rio Verde. Aluno de Iniciação Científica – PIBIC.
[2]Graduando do curso de Engenharia de Software, Universidade de Rio Verde. Aluno de Iniciação Científica – PIBIC
[3] Prof. Dr(a). do Departamento de Engenharia de Produção, Universidade Federal do Amazonas, Manaus-AM.
[4]Prof. Dr. da Faculdade de Engenharia Mecânica, Universidade de Rio Verde.
[5]Co-orientadora, Prof. Dr(a). da Faculdade de Engenharia Mecânica, Universidade de Rio Verde.
[6]Doutorando em Engenharia de Produção/PPGEP-UFMG, Universidade Federal de Minas Gerais, Belo Horizonte-MG.
[7] Orientador, Prof. MSc. da Faculdade de Engenharia de Produção, Universidade de Rio Verde. darlan@unirv.edu.br.

Fomento:
Programa PIBIC/PIVIC UniRV/CNPq 2022-2023

Resumo: O problema de projeto de redes não capacitadas com mono-produto e custos fixos apresenta importantes aplicações em sistemas logísticos e de telecomunicações. Existem diversas formulações para retratar esta abordagem. Visto isto, surgiu uma lacuna em implementar em Python a partir da formulação (S) o método da Relaxação Lagrangeana. Foi aplicado em 10 instâncias diferentes para obter os limites (*bounds*) superior e inferior a partir de um algoritmo. Foi possível desenvolver o algoritmo e implementá-lo para obter os *bounds*. Os resultados foram bastantes satisfatórios quando comparados a implementação do Modelo S. É válido destacar que esta pesquisa está inserida dentro de um projeto maior, sendo um fragmento deste projeto global.

Palavras-Chave: Problema logístico de grande escala. Relaxação Lagrangeana. Implementação em Python.

The problem of designing uncapacitated networks with single-product and fixed costs: Lagrangian relaxation

Abstract: *The problem of designing uncapacitated networks with a single product and fixed costs has essential applications in logistics and telecommunications systems. There are several formulations to portray this approach. Given this, a gap arose in implementing the Lagrangian Relaxation method based on formulation (S) in Python. It was applied in 10 different instances to obtain an algorithm's upper and lower bounds. Developing the algorithm and implementing it to get the bounds was possible. The results were entirely satisfactory when*

compared to the implementation of Model S. It is worth highlighting that this research is part of a larger project, being a fragment of this global project.

Keywords: *Large-scale logistical problem. Lagrangian relaxation. Implementation in Python.*

Introdução

O problema de projeto de redes não capacitadas com mono-produto e custos fixos consiste em projetar uma rede transporte de forma que os fornecedores sejam capazes de suprir com produtos as demandas no menor custo possível, tanto em questões de transporte quanto de instalação da infraestrutura. Tal problema é também conhecido como single commodity uncapacitated fixed charge network design problem ou ainda uncapacitated fixed charge network design problem (UFCNDP).

O UFCNDP inicialmente formulado por Hirsch e Dantzig (1968) teve destaque nas primeiras duas décadas seguintes, sendo tratado por trabalhos como Barr *et al.* (1981), Cabot e Erenguc (1984), Magnanti *et al.* (1986), Palekar *et al.* (1990), e Guisewite e Pardalos (1990). Nas últimas duas décadas, os trabalhos de Cruz *et al.* (1998), Ortega e Wolsey (2003), Gendron (2019) e Zetina *et al.* (2019) são as principais referências da literatura.

Problemas envolvendo cargas fixas evoluíram no decorrer desse período. Os precursores Hirsch; Dantzig (1968) reduzem o problema de programação linear comum baseado no problema de carga fixa. Tais autores destacam a relação entre o problema geral de carga fixa e problemas lineares sem restrições à não negatividade e às variáveis inteiras. O problema do caixeiro viajante, por exemplo, já era computacionalmente viável de ser implementado naquela época, mas a principal limitação do estudo desses autores foi a possível falta de recursos para implementar seus métodos em um computador e verificar a eficiência de seus modelos.

Uma técnica alternativa para obter os limites de uma formulação pode estar associada a Relaxação Lagrangeana. Held e Karp (1970, 1971) foram os primeiros a usar multiplicadores Lagrangeanos para dualizar restrições e obter uma estratégia de relaxação para ser usada dentro de uma árvore de enumeração implícita (branchand-bound) desenvolvida para resolver o problema do caixeiro viajante. Apesar de outros autores (Lorie; Savage, 1955; Everett, 1963; Gilmore; Gomory, 1963) terem usados multiplicadores Lagrangeanos para problemas discretos antes, a estratégia de Held e Karp deu origem ao que conhecemos hoje como o método Lagrangeano para problemas discretos.

Vários modelos foram desenvolvidos ao longo dos anos e técnicas de resolução de formulação. Sendo assim, existe uma lacuna em realizar o desenvolvimento de um algoritmo da Relaxação Lagrageana para o UFCNDP-S (proposto por Ortega (2003) e obter os limites inferiores e superiores (*bounds*).

Material e Métodos

A formulação para o UFCNDP, seja um grafo (rede) direcionado D = (N, A) com os conjuntos de nós N= {1, ..., n} e de arcos $A \subseteq \{ (i, j) \in N \times N : i \neq j\}$. Cada arco (i, j) A possui um custo unitário de transporte $c_{ij} \geq 0$ e um custo fixo $a_{ij} \geq 0$ de instalação; enquanto que cada nó de oferta $j \in N$ tem um valor $d_j > 0$, ou como nó de oferta, quando $d_j < 0$, ou como nó de transbordo, quando $d_j = 0$. Para termos um problema viável, temos que $\Sigma_{j \in N} d_j = 0$. Definimos também $K+ = \{ j \in N: d_j > 0\}$, $K- = \{ j \in N: d_j < 0\}$, e $K^0 = \{ j \in N: d_j = 0\}$ como os conjuntos de nós de demanda, oferta, e transbordo, respectivamente, sendo ainda $D = \Sigma_{j \in K+} d_j$.

Usando as variáveis de decisão binárias $x_{ij} \in \{0, 1\}$ e de fluxo $f_{ij} \geq 0$ para representar se um arco $(i, j) \in A$ é instalado ($x_{ij} = 1$) ou não ($x_{ij} = 0$), e a quantidade de fluxo passando pelo arco (i, j), respectivamente, pode-se formular o UFCNDP como (ORTEGA; WOLSEY, 2003):

$$\emptyset_S(f, x) = min \sum_{(i,j) \in A} \left(c_{ij} f_{ij} + a_{ij} x_{ij}\right) \quad (1)$$

$$s.a.: \sum_{(i,j)\in A} f_{ij} - \sum_{(j.i)\in A} f_{j,i} = d_j \qquad \forall\, j \in N \qquad (2)$$

$$f_{ij} \leq Dx_{ij} \qquad \forall\, (i,j) \in A \qquad (3)$$

$$x_{ij} \in \{0,1\} \qquad \forall\, (i,j) \in A \qquad (4)$$

$$f_{ij} \geq 0 \qquad \forall\, (i,j) \in A \qquad (5)$$

A função objetivo (1) minimiza os custos totais de instalação de rede de fluxo. As restrições (2) são de conservação ou balanço de fluxo pela rede. Se $d_j < 0$, então d_j unidades de fluxo sairão do nó $j \in N$; enquanto que se $d_j > 0$, então d_j unidades de fluxo serão deixados no nó $j \in N$; sendo que se $d_j = 0$, então a mesma quantidade de fluxo que entra no nó $j \in N$ sairá dele. As restrições (3) são restrições de ativação, exemplo, só pode existir fluxo passando no arco $(i, j) \in A$, se o arco (i, j) estiver instalado. Finalmente, restrições (4) e (5) mostram o domínio das variáveis.

Para a formulação S, apesar de termos algumas alternativas de dualização, vamos dualizar as restrições (3) através dos multiplicadores Lagrangeanos $v_{ij} \geq 0$ para obtermos o problema Lagrangeano abaixo:

$$L(v) = \min \sum_{(i,j)\in A} (c_{ij} + v_{ij}) f_{ij} + \sum_{(i,j)\in A} (a_{ij} - Dv_{ij}) x_{ij} \qquad (6)$$

$$\text{s.a.}: \sum_{(i,j)\in A} f_{ij} - \sum_{(i,j)\in A} f_{ji} = d_j \qquad (7)$$

$$x_{ij} \in \{0,1\} \qquad \forall\, (i,j) \in A \qquad (8)$$

$$f_{ij} \geq 0 \qquad \forall\, (i,j) \in A \qquad (9)$$

As restrições (7) de conservação de fluxo. A fij é contínua, relacionada ao fluxo nas restrições (9) e ligado ao domínio das variáveis junto com (8). Para achar o limite inferior Lagrangeano mais próximo, ou seja, "justo" de Φ_S (f, x), temos de resolver o seguinte problema dual Lagrangeano de S relativo as restrições (3) através de (10):

$$\emptyset(L) = \max_{v \geq 0} L(v) \qquad (10)$$

Um limite de relaxação Lagrangeano ou simplesmente um limite Lagrangeano é o valor de Φ(L) e não pelo valor de L(v) obtido a partir de um v arbitrário. Temos então que achar v de forma que este nos dê um limite Lagrangeano válido. Aqui, vamos determinar o valor de v de forma aproximada pelos métodos do subgradiente (Held e Karp, 1971). O método do subgradiente é um procedimento iterativo que aproxima o valor ótimo de v a partir de um valor inicial v0 e uma sequência de {vh} obtida através da regra (equações (11)):

$$v_{ij}^{h+1} = \max\left(0, v_{ij}^{h} + \sigma^{h}\left(f_{ij}^{h} - Dx_{ij}^{h}\right)\right) \qquad (11)$$

O tamanho do passo a ser usado comumente na prática como (12):

$$\sigma^{h} = y^{h} \frac{(LS - L(v^{h}))}{\sum_{(i,j)\in A} (f_{ij}^{h} - Dx_{ij}^{h})^2} \qquad (12)$$

A partir destas informações será desenvolvido o algoritmo.

Foi gerado as coordenadas dos nós de 10 instâncias de forma aleatória numa área quadrada de 100 x 100 com o número de nós da rede variando de |N| = {10, 20, 30, 40, 50}. Depois foi calculada a distância Euclidiana entre todos os pares de nós, porém arredondando a distância para o valor inteiro mais próximo. Para determinar o conjunto de arcos, usou a triangulação de Delaunay que encontra as arestas dos triângulos cujas circunferências não têm nenhum nó de N dentro. Fixou o custo fixo de

cada arco proporcionalmente ao seu comprimento multiplicado por um fator igual a 50. O número de nós de demandas selecionados aleatoriamente do conjunto N foi selecionado de acordo com | K+ | = {5, 10, 15}, sendo as demandas d_j, $j \in K+$, geradas seguindo uma distribuição uniforme U[1, 10] de valores inteiros. Os nós fontes do conjunto K- foram selecionados aleatoriamente entre N\ K+ de forma que a demanda total seja aleatoriamente distribuída entre eles, porém com o sinal trocado.

Com a geração das instâncias em extensão .py e a implementação das formulações foram através do Windows 7 Ultimate com Intel(R) Core(TM) i5 − 3337U CPU @1:80 GHz processador, 6 GB RAM, e 64x arquitetura. O software Python 3:8 e os pacotes computacionais como Python-MIP foram aplicados para resolver os modelos. Posteriormente foram obtidos os limites superiores e inferiores para cada instância através do algoritmo desenvolvido.

Resultados e Discussão

Através da Tabela 1 é possível verificar as dez instâncias (i1, i2, i3, ..., i10) geradas que foram codificas de acordo com cada configuração. Para cada configuração, tem-se na segunda coluna (N) a quantidade de nós, terceira coluna (A) a quantidade de arcos, quarta coluna a quantidade de nós de demanda (K+) e quinta coluna (K-) a quantidade de nós de oferta. As colunas subsequentes encontram-se os resultados obtidos da implementação: Quantidade de interação, o tempo (s) e os *bounds* (LI e LS) para a Relaxação Lagrangeana.

Devido às limitações computacionais, a quantidade de nós de oferta gerados por configuração foram restringidas quanto ao tempo disponível para compilar os modelos no software e a complexidade das árvores geradas. Para todos os resultados, criou-se gráficos para uma melhor compreensão dos valores obtidos, explorando-os melhor.

Tabela 1 – Resultados gerais obtidos da Relaxão Lagrangeana

Código	N	A	K+	K-	Iterações (número)	Tempo (s)	LI	LS
i1	10	44	5	2	674	3,525	2.318,94	7.420,00
i2	10	44	5	2	647	3,003	1.803,73	6.985,00
i3	20	98	10	3	587	5,157	3.921,12	12.836,00
i4	20	102	10	2	775	7,155	4.605,46	17.778,00
i5	30	160	10	5	677	9,549	2.946,91	16.531,00
i6	30	152	10	5	674	9,212	3.259,67	14.058,00
i7	40	220	15	4	720	13,733	4.729,74	23.227,00
i8	40	218	15	4	641	12,251	3.897,26	18.227,00
i9	50	270	15	3	725	17,741	4.421,96	18.381,00
i10	50	274	15	3	800	19,352	4.291,70	17.232,00

Fonte: autoria própria

O gráfico da Figura 1 evidencia dois resultados, a quantidade de iterações realizadas (n°) e o tempo (s), ambos em relação as dez instâncias. É notório que o tempo representado em vermelho apresenta uma ascendência de execução de acordo a rede fica maior e mais complexa. O R^2 foi bastante satisfatório para tal, chegando a 0,9447. Em relação a quantidade de interações necessárias para obter os limites, apresenta evidências de um aumento mais moderado, entretanto, o R^2 (0,2604) não permite tirar conclusões mais sólidas. Um fator interessante é que a complexidade em resolver o problema não se encontra apenas no tamanho da rede, mas também na quantidade de nós de oferta (Zetina *et al.*, 2019).

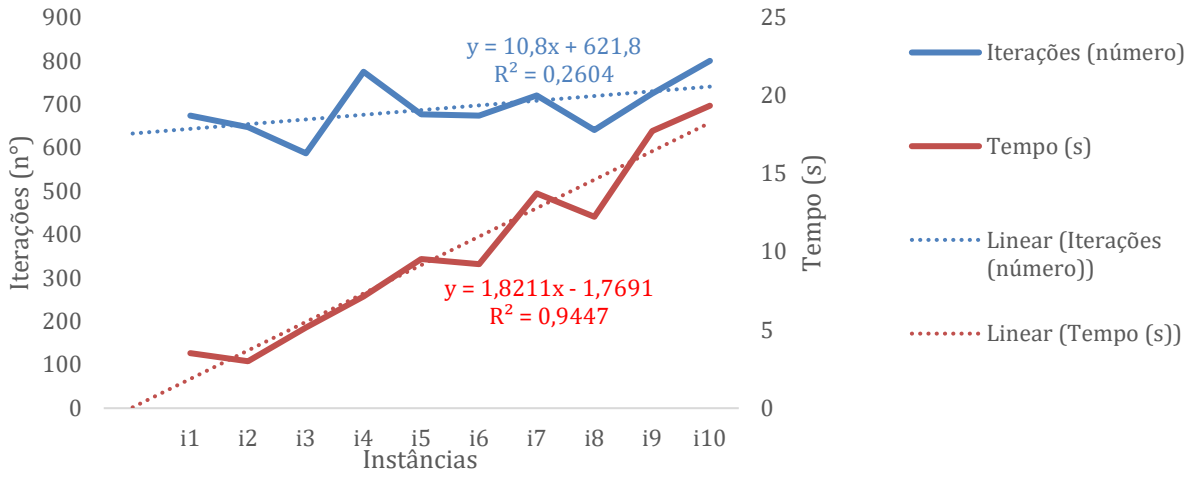

Figura 1 – Apresentação do número de iterações e o tempo de execução para cada instância gerada
Fonte: autoria própria

É sabido que as redes mais densas sugerem um maior tempo para solucionar o problema em virtude da complexidade (Gendron, 2019).

A Figura 2 apresenta os limites inferiores e superiores para o algoritmo gerado. Pode-se notar que a instância de número 7 apresentou uma diferença entre LI e LS. Tal fato pode ser explicado pela complexidade que uma rede pode se tornar com a combinação da quantidade de nós (oferta e demanda) aliada a configuração dos arcos (Ortega; Wolsey, 2003).

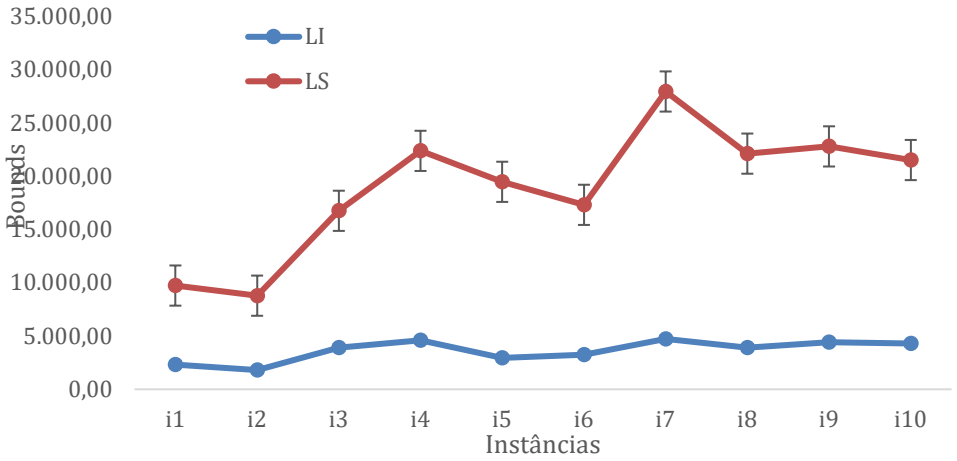

Figura 2 – Limites inferiores e superiores da Relaxação Lagrageana
Fonte: autoria própria

Por fim, apresenta-se a Figura 3 com o algoritmo que proporcionou chegar nos resultados obtidos:

$LS, LI \leftarrow \infty, -\infty$
$h, \mathbf{v^h}, \gamma^h \leftarrow 0, \mathbf{0}, 2$
$PARE, M \leftarrow N, 0$
while $PARE = N$ **do**
  $(inf, \bar{x}, \bar{f}) \leftarrow L(\mathbf{v^h})$
  $(sup, \bar{x}^p, \bar{f}^p) \leftarrow SoluçãoPrimal(\bar{x}, \bar{f})$
  $LS \leftarrow \min(sup, LS)$
  if $LI < inf$ **then**
    $LI, M \leftarrow inf, 0$
  else
    $M \leftarrow M + 1$
  end if
  if $M = M^{MAX}$ **then**
    $\gamma^h, M \leftarrow \frac{\gamma^h}{2}, 0$
  end if
  $g_{ij} \leftarrow (\bar{f}_{ij}^h - D\bar{x}_{ij}^h)^2 \quad \forall (i,j) \in A$
  $norm \leftarrow \sum_{(i,j)\in A} (\bar{f}_{ij}^h - D\bar{x}_{ij}^h)^2$
  $gap \leftarrow 100\frac{(LS-LI)}{LS}$
  if $(norm < \varepsilon^{norm}) \vee (\gamma^h < \varepsilon^{\gamma}) \vee (h = h^{MAX}) \vee (gap < \varepsilon^{gap})$ **then**
    $PARE \leftarrow S$
  else
    $\sigma^h \leftarrow \gamma^h \frac{(LS-LI)}{norm}$
    $v_{ij}^{h+1} \leftarrow \max(0, v_{ij}^h + \sigma^h g_{ij}^h) \quad \forall (i,j) \in A$
  end if
  $h \leftarrow h + 1$
end while

Figura 3 – Algoritmo gerado para propor a Relaxação Lagrangeana
Fonte: autoria própria

Conclusão

Ao longo destas implementações, vimos as dificuldades e facilidades de implementação do algoritmo desenvolvido. Vimos também o desempenho do método na resolução do problema proposto. Os resultados foram bastantes satisfatórios por conseguir chegar em valores plausíveis na implementação quando comparados ao Modelo S (extensão desta pesquisa).

Agradecimentos

A pesquisa apresenta como agradecimento ao apoio financeiro inicial da UniRV-PIBIC e posteriormente pelo incentivo UniRV-PIVIC.

Referências Bibliográficas

BARR, R.; GLOVER, F.; KLINGMAN, D. A new optimization method for large scale fixed charge transportation problems. **Operations Research**, 29:448–463, 1981.

CABOT, A.; ERENGUC, S. Some branch-and-bound procedures for fixed-cost transportation problems. **Naval Research Logistics Quarterly**, 31:145–154, 1984.

CRUZ, F.; SMITH, J.; MATEUS, G. Solving to optimality the uncapacitated fixed-charge network flow problem. **Computers & Operations Research**, 25(1):67–81, 1998.

EVERETT, H. Generalized lagrange multiplier method for solving problems of optimum allocation of resources. **Operations Research**, 11(3):399–417, 1963.

GENDRON, B. Revisiting lagrangian relaxation for network design. **Discrete Applied Mathematics**, 261:203–218, 2019.

GILMORE, P. C.; GOMORY, R. E. A linear programming approach to the cutting stock problem—part ii. **Operations Research**, 11(6):863–888, 1963.

GUISEWITE, G.; PARDALOS, P. Minimum concave-cost network flow problems: applications, complexity, and algorithms. **Annals of Operations Research**, 25:75–100, 1990.

HELD, M.; KARP, R. The traveling-salesman problem and minimum spanning trees: Part ii. **Mathematical Programming**, 1:6–25, 1971.

HELD, M.; KARP, R. M. The traveling-salesman problem and minimum spanning trees. **Operations Research**, 18(6):1138–1162, 1970.

HIRSCH, W.; DANTZIG, G. The fixed charge problem. **Naval Research Logistics Quarterly**, 15:413–424, 1968.

LORIE, J. E SAVAGE, L. J. Three problems in rationing capital. **The Journal of Business**, 28:229–229, 1955.

MAGNANTI, T. L.; MIREAULT, P.; WONG, R. T. **Tailoring Benders decomposition for uncapacitated network design**. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 112–154, 1986.

ORTEGA, F; WOLSEY, L. A. A branch-and-cut algorithm for the single-commodity, uncapacitated, fixed-charge network flow problem. **Networks**, 41(3):143–158, 2003.

PALEKAR, U. S.; KARWAN, M. H.; ZIONTS, S. A branch-and-bound method for the fixed charge transportation problem. **Management Science**, 36(9):1092–1105, 1990.

HIRSCH, W.; DANTZIG, G. The fixed charge problem. **Naval Research Logistics Quarterly**, 15:413–424, 1968.

ZETINA, C. A.; CONTRERAS, I.; CORDEAU, J. F. Exact algorithms based on ben-ders decomposition for multicommodity uncapacitated fixed-charge network design. **Computers & Operations Research**, 111:311–324, 2019.