

# Uma Aproximação para o Problema de Alocação de Terminais com Capacidade

Lehilton L.C. Pedrosa\*, Vinícius Balbino de Souza†

<sup>1</sup>Instituto de Computação – Universidade Estadual de Campinas  
Campinas, São Paulo

lehilton@ic.unicamp.br, vbalbinods@gmail.com

**Abstract.** We consider the Capacitated  $p$ -Hub Center Problem. An instance comprises a metric space  $V$ , a set of demands  $D \subseteq V^2$ , a number of hubs  $p$ , and a capacity  $L$ . A solution is a multiset  $S$  of locations where to install hubs with  $|S| \leq p$  and an assignment from each demand to a hub such that no hub receives more than  $L$  demands. The objective is to find the solution that minimizes the maximum cost of serving a demand through the assigned hub. In this work, we give the first approximation algorithm for the problem, that achieves factor 7.

**Resumo.** Consideramos o Problema de Alocação de Terminais com Capacidade. Uma instância é composta de um espaço métrico  $V$ , um conjunto de demandas  $D \subseteq V^2$ , um número de terminais  $p$  e uma capacidade  $L$ . Uma solução é um multiconjunto  $S$  de locais para instalar terminais com  $|S| \leq p$  e, para cada demanda, um terminal associado de forma que nenhum deles receba mais de  $L$  demandas. O objetivo é encontrar uma solução que minimiza o maior custo de servir uma demanda através do terminal atribuído. Neste trabalho, obtemos o primeiro algoritmo de aproximação para o problema, com fator 7.

## 1. Introdução

Em diversas aplicações de transportes, logística, etc., necessitamos da instalação de terminais, como aeroportos de conexão ou *switches* de rede. Um terminal é um centro de consolidação, troca e ordenação, que permite remanejamento de conexões diretas, usando uma quantidade menor de ligações indiretas. Uma vez que a maioria de problemas voltados a essas aplicações são NP-difíceis, consideramos alternativas, como algoritmos de aproximação: um algoritmo polinomial para um problema de otimização é uma  $\alpha$ -aproximação se o custo da solução encontrada é no máximo  $\alpha$  vezes mais caro que a melhor solução existente. Enquanto aproximações para problemas de localização, como o problema clássico dos  $k$ -Centros são conhecidas desde os anos 80 [Hochbaum and Shmoys 1986], as primeiras aproximações para problemas de alocação de terminais só apareceram depois, como para o chamado Problema de Alocação de Terminais em Estrela, em 2009 [Iwasa et al. 2009]. Recentemente, Pedrosa et al. [Pedrosa et al. 2016] consideraram o Problema de Centro de Alocação dos  $p$ -Terminais (*Capacitated  $p$ -Hub Center Problem*,  $p$ -HP), mostrando que é NP-difícil encontrar uma aproximação com fator melhor que 2 e obtendo um algoritmo com fator 3.

---

\*Este trabalho foi parcialmente financiado pelo processo nº 2015/11937-9, Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP).

†Bolsista do CNPq - Brasil.

Um exemplo de situação modelada pelo  $p$ -HP ocorre se quisermos conectar pares de computadores instalando um número  $p$  de roteadores. Nesse caso, o objetivo é minimizar a pior latência entre um computador de origem, o roteador associado e o computador de destino. Na prática, um roteador pode ter um limite de conexões a que pode atender, o que significa que uma solução para o  $p$ -HP nem sempre é aplicável. Neste trabalho, consideramos uma generalização do  $p$ -HP para modelar esse tipo de restrição. Especificamente, consideramos o *Capacitated*  $p$ -HP, denotado por  $p$ -CHP, em que uma entrada também contém um número  $L$  e cada um dos  $p$  terminais instalados pode ser associado a no máximo  $L$  demandas.

Nossa contribuição é a primeira aproximação para o  $p$ -CHP, com fator de aproximação 7. O algoritmo utiliza o método do gargalo [Hochbaum and Shmoys 1986], em que se reduz um problema de um espaço métrico para o caso particular em que os pontos estão em um grafo sem pesos. Para lidar com as capacidades, criamos um particionamento de demandas e resolvemos um problema de fluxo máximo, baseando-se na *árvore de monarcas* de [Khuller and Sussmann 2000], utilizada em sua clássica 5-aproximação para o problema dos  $k$ -centros com capacidade.

## 2. Uma 7-Aproximação para o $p$ -CHP

Formalmente, uma instância do  $p$ -CHP é dada por um grafo completo  $G$  com vértices  $V$  e métrica  $d$ , em que  $d(x, y) \geq 0$  é a distância entre os vértices  $x, y \in V$ ; um conjunto  $D$  de pares  $(x, y) \in V \times V$ , em que cada um representa uma demanda entre os vértices  $x$  e  $y$ ; um número inteiro  $p$ , que define o número de terminais a serem instalados; e um número  $L$ , cada um representando a capacidade de um terminal. Uma solução é um multiconjunto  $S$  de elementos de  $V$ , com  $|S| \leq p$ , e uma função  $\phi : D \rightarrow S$ , tal que  $|\phi^{-1}(v)| \leq L$  para todo  $v \in S$  (i.e.,  $\phi$  associa no máximo  $L$  demandas a  $v$ ). O objetivo é encontrar uma solução que minimiza  $\max_{(x,y) \in D} d(x, \phi(x, y)) + d(\phi(x, y), y)$ .

### 2.1. Algoritmo

O algoritmo para o  $p$ -CHP consiste de quatro rotinas,  $p$ -CAPHUB, GETMONARCHS, SETDOMAINS e ASSIGN, que são detalhadas nos Algoritmos 1 a 4. Uma descrição textual é apresentada a seguir.

A rotina  $p$ -CAPHUB obtém uma estimativa do valor de uma solução ótima,  $\tau$ , enumerando cada uma das possibilidades. Dada uma estimativa  $\tau$ , cria-se um grafo  $H$  bipartido entre demandas  $D$  e locais  $V$ , chamado de grafo de gargalo, em que uma aresta significa que uma demanda pode ser servida a partir de um certo local ao custo de no máximo  $\tau$ . Se houver demanda isolada, então pode-se testar o próximo valor de  $\tau$  no conjunto de possibilidades, já que nesse caso não existe terminal que satisfaz essa demanda com custo  $\tau$ . Se toda demanda é incidente a pelo menos uma aresta em  $H$ , então o problema se reduz em encontrar uma solução com o número mínimo de terminais em que cada demanda é satisfeita por um terminal na vizinhança de  $H$ . Iremos mostrar depois que, para um certo  $\tau$ , ou  $p$ -CAPHUB encontra uma solução com até  $p$  terminais em que cada demanda é atribuída a um terminal à distância até 7 em  $H$ , ou em qualquer solução do problema original de custo até  $\tau$ , mais do que  $p$  terminais são necessários. No primeiro caso, o algoritmo devolverá uma 7-aproximação e o algoritmo termina; no último caso, conclui-se que  $\tau$  está subestimando o valor ótimo, e testa-se a próxima possibilidade.

**Algoritmo 1:**  $p$ -CAPHUB( $G, d, D, L, p$ )

1.  $F \leftarrow \{d(x, h) + d(h, y) : (x, y) \in D, h \in V\}$
2. Para cada  $\tau \in F$ , em ordem crescente:
  - (a) Crie o grafo  $H = (D \cup V, E)$ , com arestas  $E = \{((x, y), v) : d(x, v) + d(v, y) \leq \tau\}$ .
  - (b) Se existe  $(x, y)$  isolado em  $H$ , continue em 2.
  - (c)  $M, T, r \leftarrow \text{GETMONARCHS}(H)$
  - (d)  $\text{Dom}, \text{Emp} \leftarrow \text{SETDOMAINS}(H, M, L)$
  - (e) Para  $m \in M$ , faça  $\text{Pass}(m) \leftarrow \emptyset$ .
  - (f)  $S \leftarrow \text{ASSIGN}(H, T, L, r)$ .
  - (g) Se  $|S| \leq p$ , termine e devolva  $S$ .

**Algoritmo 3:** SETDOMAINS( $H, M, L$ )

1. Crie uma rede de fluxo  $N$  com vértices  $D$ , uma cópia de cada demanda de  $M$  e dois novos nós,  $s, t$ . O conjunto de arcos é:
  - Para  $e \in D$ , um arco  $(s, e)$  de capacidade 1.
  - Para  $m \in M$ , um arco  $(m, t)$  de capacidade  $L$ .
  - Para cada  $e \in D$  e cópia  $m \in M$ , um arco não capacitado  $(e, m)$  se  $\text{dist}_H(e, m) \leq 2$ .
2. Obtenha um fluxo  $s$ - $t$  máximo em  $N$ .
3. Para cada  $m \in M$ , seja  $\text{Dom}(m)$  (domínio de  $m$ ) o conjunto de demandas  $e$  por que passa uma unidade de fluxo e cujo fluxo flua por  $m$ .
4. Para cada  $m$ , seja  $\text{Emp}(m)$  (império de  $m$ ) o conjunto de demandas  $e$  com  $d(m, e) \leq 2$  que ainda não estão no domínio ou império de algum monarca. Devolva  $\text{Dom}$  e  $\text{Emp}$ .

**Algoritmo 2:** GETMONARCHS( $H$ )

1. Escolha demanda  $r \in D$  arbitrária.
2. Faça  $M \leftarrow \{r\}$  e torne  $r$  raiz de  $T$ .
3. Enquanto houver  $e \in D \setminus M$ ,  $\text{dist}_H(e, M) \geq 4$ :
  - (a) Escolha  $f \in M$  e  $e \in D \setminus M$ , tais que  $\text{dist}_H(e, f) = 4$ .
  - (b) Adicione  $e$  a  $M$  e faça  $f$  pai de  $e$  em  $T$ .
4. Devolva  $M, T$  e  $r$ .

**Algoritmo 4:** ASSIGN( $H, T, L, m$ )

1. Execute ASSIGN( $H, T, L, f$ ) em cada filho  $f$  de  $m$ .
2. Sejam  $b \geq 0$  e  $0 \leq c < L$  números inteiros tais que  $bL + c = |\text{Pass}(m)| + |\text{Emp}(v)|$ .
3. Escolha um vizinho arbitrário  $v$  de  $m$  no grafo  $H$  e instale  $b + 1$  terminais em  $v$ .
4. Atribua cada demanda  $e \in \text{Pass}(m) \cup \text{Emp}(m)$  aos  $b + 1$  terminais; em seguida, atribua a esses quantos elementos de  $\text{Dom}(v)$  forem possíveis sem exceder a capacidade.
5. Se  $m$  for a raiz:
  - então instale um novo terminal em  $v$  e atribua as demandas excedentes de  $\text{Dom}(m)$ ;
6. Caso contrário:
  - inclua as demandas excedentes em  $\text{Pass}(m')$ , em que  $m'$  é o pai de  $m$ .
7. Devolva o conjunto de terminais instalados.

O laço interno de  $p$ -CAPHUB contém três chamadas a sub-rotinas. A primeira, GETMONARCHS, obtém a chamada *árvore de monarcas*,  $T$ , de maneira gulosa.  $T$  é construída de forma que monarcas pai e filho estejam à distância 4 em  $H$  e que cada demanda  $e \in D$  seja um monarca, ou esteja a distância 2 de algum monarca, i.e,  $\text{dist}_H(e, m) \leq 2$  para  $m \in M$ . Essa propriedade garante que, para cada monarca  $m \in M$ , uma solução ótima  $S^*$  contenha um terminal  $h \in S^*$ , chamado testemunha, que é vizinho de  $m$  em  $H$ .

A segunda sub-rotina é SETDOMAINS. Para cada  $m \in M$ , definem-se conjuntos disjuntos  $\text{Dom}(m)$  e  $\text{Emp}(m)$  de demandas com distância até 2 de  $m$ , chamados de domínio e império de  $m$ . A família de todos pares  $\text{Dom}(m)$  e  $\text{Emp}(m)$  particionam  $D$ . Os domínios são escolhidos de forma que  $|\text{Dom}(m)| \leq L$  para cada  $m$ , mas de forma a maximizar o número de demandas que podem ser servidas se houvesse um terminal em cada monarca. Intuitivamente, a união dos domínios serve pelo menos tantas demandas quantas são servidas na solução ótima  $S^*$  pelas testemunhas de  $M$ . Para maximizar o número de demandas servidas pelos monarcas, é resolvido um problema de fluxo máximo.

Finalmente, ASSIGN instala e atribui demandas a terminais, percorrendo a árvore de monarcas de forma *bottom-up*. Para cada monarca  $m$ , escolhe-se um local  $v$  na vizinhança de  $m$  e instalam-se  $b + 1$  terminais, a fim de servir as demandas de  $\text{Emp}(m)$  e, possivelmente, um conjunto de demandas passadas pelos filhos de  $m$ ,  $\text{Pass}(m)$ . As demandas em  $\text{Dom}(m)$  são atribuídas aos terminais com capacidade livre em  $v$  e as excedentes passadas ao pai de  $m$ . O algoritmo termina quando a raiz é considerada. Nesse caso, não é possível transferir demandas remanescentes e um novo terminal é instalado.

## 2.2. Análise

A correção do algoritmo utiliza dois fatos principais: para um dado  $\tau$ , a rotina  $p$ -CAPHUB não seleciona mais terminais do que uma solução ótima; e, para cada demanda, o terminal associado está à distância no máximo 7. Primeiro precisamos da seguinte definição.

**Definição.** Um monarca leve é um monarca  $m \in M$  com  $|\text{Dom}(m)| < L$ . Um monarca cheio é um monarca  $m \in M$  com  $|\text{Dom}(m)| = L$ .

**Lema 1.** Seja  $k_L$  o número de monarcas leves,  $n_L$  é o número de demandas nos domínios de monarcas leves e  $n$  o número total de demandas. Se  $S^*$  é o conjunto de terminais de uma solução de custo  $\tau$  com um número mínimo de terminais, então  $|S^*| \geq k_L + \lceil \frac{n-n_L}{L} \rceil$ .

O seguinte lema limita a distância entre uma demanda e o terminal atribuído.

**Lema 2.** Seja  $e$  uma demanda. Se  $e$  foi atribuída a um terminal  $v$ , então  $\text{dist}_H(e, v) \leq 7$ .

*Demonstração.* Seja  $e$  uma demanda e seja  $m$  o monarca correspondente à execução de ASSIGN no momento em que  $e$  foi atribuída. Vamos mostrar que  $\text{dist}_H(e, m) \leq 6$ . Consideramos dois casos. Se  $e$  pertence ao domínio ou ao império de  $m$ , então temos  $\text{dist}_H(e, m) \leq 2$ . Do contrário, então  $e \in \text{Pass}(m)$ , i.e.,  $e$  é uma demanda passada por um filho de  $m$  na árvore de monarcas, diga-se  $m'$ . Como  $e$  não foi atribuído a  $m'$ , segue que  $e \in \text{Dom}(m')$ . Obtemos  $\text{dist}_H(m', m) = 4$  e  $\text{dist}_H(e, m') \leq 2$ . Combinando essas equações, obtemos  $\text{dist}_H(e, m) \leq \text{dist}_H(e, m') + \text{dist}_H(m', m) \leq 2 + 4 = 6$ .

Para concluir o lema, basta observar que todas as demandas atribuídas por ASSIGN, ao ser executado com monarca  $m$ , são atribuídas a um mesmo local  $v$  na vizinhança de  $m$ . Daí,  $\text{dist}_H(m, v) = 1$  e, finalmente,  $\text{dist}_H(e, v) \leq \text{dist}_H(e, m) + \text{dist}_H(m, v) \leq 6 + 1 = 7$ .  $\square$

**Lema 3.** Seja  $v \in V$  e  $e = (x, y) \in D$ . Se  $\text{dist}_H(e, v) = \alpha$ , então  $d(x, v) + d(v, y) \leq \alpha\tau$ .

**Teorema 1.** Existe uma 7-aproximação para o  $p$ -CHP.

*Demonstração.* Primeiro note que o algoritmo executa em tempo polinomial. Também, pelo Lema 2, em cada iteração de  $p$ -CAPHUB com valor  $\tau$ , obtemos uma atribuição  $\phi$  tal que para cada  $e \in D$ ,  $\text{dist}_H(e, \phi(e)) \leq 7$ . Pelo Lema 3, obtemos que  $d(e, \phi(e)) \leq 7\tau$ .

Seja OPT o custo da solução ótima  $S^*$ . Note que OPT é considerado pelo algoritmo  $p$ -CAPHUB, i.e.,  $\text{OPT} \in F$ . Pelo Lema 1, na iteração em que  $\tau = \text{OPT}$ , vale  $|S^*| \geq k_L + \lceil \frac{n-n_L}{L} \rceil$ . Como os únicos terminais instalados e não completamente ocupados estão associados a monarcas leves e, possivelmente, um único terminal instalado para a raiz  $r$ , segue que o ASSIGN instala, no máximo,  $k_L + \lceil \frac{n-n_L}{L} \rceil \leq |S^*| \leq p$  terminais. Portanto, para alguma iteração com valor  $\tau' \leq \text{OPT}$ , o algoritmo irá devolver uma solução de custo no máximo  $7\tau'$  e com no máximo  $p$  terminais.  $\square$

## Referências

- [Hochbaum and Shmoys 1986] Hochbaum, D. S. and Shmoys, D. B. (1986). A Unified Approach to Approximation Algorithms for Bottleneck Problems. *J. ACM*, 33(3):533–550.
- [Iwasa et al. 2009] Iwasa, M., Saito, H., and Matsui, T. (2009). Approximation algorithms for the single allocation problem in hub-and-spoke networks and related metric labeling problems. *Discrete Applied Mathematics*, 157(9):2078–2088.
- [Khuller and Sussmann 2000] Khuller, S. and Sussmann, Y. J. (2000). The Capacitated  $K$ -Center Problem. *SIAM Journal on Discrete Mathematics*, 13(3):403–418.
- [Pedrosa et al. 2016] Pedrosa, L. L. C., Santos, V. F., and Schouery, R. C. S. (2016). Uma aproximação para o problema de alocação de terminais. In *Anais do CSBC 2016 (1º ETC - 2016)*.