

Algoritmo de Balas aplicado ao TSP com janelas de tempo e carro elétrico

23 de julho de 2025

Orientador: Álvaro Junio Pereira Franco - `alvaro.junio@ufsc.br`

Estudante: Felipe Lourenço da Silva - `felipe.lourenco@grad.ufsc.br`

Departamento: Informática e Estatística

Centro de ensino: Tecnológico

Título do projeto: Modelos e algoritmos para variações modernas do problema de roteamento de veículos

Fonte financiadora: Conselho Nacional de Desenvolvimento Científico e Tecnológico - Termo de Outorga - Processo: 405247/2023-0 (SIGPEX N. 202104940)

Resumo

Adaptamos o algoritmo de Balas que resolve o TSP com janelas de tempo para resolver o TSP com janelas de tempo e carro elétrico.

O algoritmo de Balas

O algoritmo de Balas foi apresentado no artigo [1]. Uma implementação deste algoritmo foi apresentada em [2]. Vamos usar a mesma notação que aparece nos artigos citados. O grafo de entrada possui n vértices e é completo. As n cidades são indexadas por índices em $N = \{1, 2, \dots, n\}$. Muitas vezes escrevemos i como o índice da cidade i . Uma permutação π de $1, 2, \dots, n$ é uma solução do TSP clássico. A posição da cidade i em uma solução viável do problema é denotado por $\pi(i)$. O grafo possui custos nas arestas t_{ij} que representam tanto a distância quanto o tempo entre os vértices i e j . O algoritmo de Balas resolve o TSP com a seguinte restrição: dado um inteiro positivo k e qualquer par de cidades i e j , se $j \geq i + k$ então em qualquer solução viável do problema, a cidade i deve preceder a cidade j , ou seja, $\pi(i) < \pi(j)$. A este problema, vamos

nos referir como TSP restrito. O algoritmo consiste de uma redução do TSP restrito para o problema de encontrar um caminho de menor custo em um grafo dirigido acíclico.

Existem propriedades importantes que valem destacar. A primeira delas determina as cidades candidatas a ocupar uma determinada posição em uma solução viável, π^{-1} . As cidades candidatas a ocupar a posição i de uma solução viável são aquelas dentro do intervalo de inteiros $\{\max\{1, i - k + 1\}, \dots, \min\{i + k - 1, n\}\}$. Podemos então escrever que uma cidade candidata $\pi^{-1}(i)$ atende a seguinte restrição

$$\max\{1, i - k + 1\} \leq \pi^{-1}(i) \leq \min\{i + k - 1, n\}.$$

Para qualquer cidade j , a posição de j em uma solução viável também está definida dentro do seguinte intervalo de inteiros $\{\max\{1, j - k + 1\}, \dots, \min\{j + k - 1, n\}\}$, ou seja,

$$\max\{1, j - k + 1\} \leq \pi(j) \leq \min\{j + k - 1, n\}.$$

Portanto, dada a $(i - 1)$ -ésima visita do caixeiro viajante a algum vértice do grafo, o próximo vértice j a ser visitado poderá ser obtido através da construção de um grafo auxiliar considerando todas as possibilidades de vértices candidatos para a posição i (ou seja, $\pi^{-1}(i) = \max\{1, i - k + 1\}, \dots, \min\{i + k - 1, n\}$) e considerando todas as posições que uma cidade j pode ocupar (ou seja, $\pi(j) = \max\{1, j - k + 1\}, \dots, \min\{j + k - 1, n\}$). Caso alguma cidade candidata j para a posição i não esteja dentro do intervalo descrito acima, então essa cidade não deve ser considerada.

O grafo auxiliar G^* é construído com $n + 1$ camadas. O conjunto de vértices da camada i do grafo auxiliar é denotado por V_i^* . $V_1^* = \{s\}$ (vértice fonte s). $V_{n+1}^* = \{t = s\}$ (vértice sorvedouro t que é igual a fonte s). O número de vértices de uma cada V_i^* é no máximo $(k + 1)2^{k-2}$ para $i = 2, \dots, n$. O grau de entrada de cada vértice do grafo auxiliar é no máximo k .

Em geral, a seguinte recorrência calcula o custo de um segmento ótimo começando na cidade 1, passando pelas cidades de um subconjunto $W \subset N$ nas posições $2, \dots, i - 1$ e visitando a cidade j na posição i :

$$C(W, i, j) = \min_{l \in W} \{C(W \setminus \{l\}, i - 1, l)\}.$$

No TSP clássico, existem muitas possibilidades para um conjunto W com $i - 2$ elementos, a saber, $\binom{n}{i-2}$. No entanto, no TSP restrito, as possibilidades para W são poucas e descritas por pares de subconjuntos cujo tamanho de cada um é limitado por k . Os pares são:

$$S^-(\pi, i) := \{l \in (1, \dots, n) : l \geq i, \pi(l) \leq i - 1\} \text{ e}$$

$$S^+(\pi, i) := \{h \in (1, \dots, n) : h \leq i - 1, \pi(h) \geq i\}.$$

Descrevendo os conjuntos acima de outra forma, $S^-(\pi, i)$ é o conjunto de cidades de uma solução viável π que possuem índices maiores ou iguais a i e

que foram visitadas em alguma posição $1, \dots, i-1$, enquanto que $S^+(\pi, i)$ é o conjunto de cidades de uma solução viável π que possuem índices menores ou iguais a $i-1$ e serão visitadas em alguma posição i, \dots, n . É demonstrado que $|S^-(\pi, i)| = |S^+(\pi, i)| \leq \lfloor k/2 \rfloor$.

Um caminho de custo mínimo no grafo auxiliar passará por um vértice em cada camada do grafo. Um vértice deste caminho na camada i contém a cidade que é visitada na posição i da solução ótima. Um vértice na camada i do grafo auxiliar é denotado por $(i, j, S_{ij}^-, S_{ij}^+)$ sendo j uma cidade candidata à posição i , e S_{ij}^- e S_{ij}^+ sendo, respectivamente, $S^-(\pi, i)$ e $S^+(\pi, i)$. A construção dos vértices $(i, j, S_{ij}^-, S_{ij}^+)$ da camada V_i^* é realizada considerando todos os candidatos j para a posição i . Dessa forma, é possível obter todos os vértices da camada V_i^* . Os arcos de G^* conectam vértices de camadas consecutivas. Dois vértices, $(i-1, l, S_{i-1,l}^-, S_{i-1,l}^+)$ e $(i, j, S_{ij}^-, S_{ij}^+)$ são adjacentes (*compatíveis*) se cada caminho $T_{i,j}$ em G que vai da cidade 1 até a cidade j na posição i pode ser obtido por um caminho $T_{i-1,l}$ em G que vai da cidade 1 até a cidade l na posição $i-1$. Ou seja, basta adicionar j ao caminho $T_{i-1,l}$ para obter $T_{i,j}$. Os vértices de um caminho $T_{i,j}$ estão no conjunto $N(i, j, S_{ij}^-, S_{ij}^+) = (N_{i-1} \setminus S_{ij}^+) \cup S_{i,j}^-$. N_i é uma notação para o conjunto das cidades $\{1, 2, \dots, i\}$. Com isso, vértices $(i-1, l, S^-, S^+)$ e (i, j, S^-, S^+) são compatíveis se e somente se

$$N(i, j, S_{ij}^-, S_{ij}^+) = N(i-1, l, S_{i-1,l}^-, S_{i-1,l}^+) \cup \{l\}.$$

Para obter os arcos de maneira eficiente, a última equação é expressa em termos dos conjuntos S_{ij}^- , S_{ij}^+ , $S_{i-1,l}^-$ e $S_{i-1,l}^+$.

Proposição de Balas. Os vértices $(i-1, l, S_{i-1,l}^-, S_{i-1,l}^+)$ e $(i, j, S_{ij}^-, S_{ij}^+)$ são adjacentes (compatíveis) se e somente se $j \neq l$ e vale uma das seguintes condições.

- Se $l < i-1$ e $i-1 \in S_{i-1,l}^-$,
então $S_{ij}^- = S_{i-1,l}^- \setminus \{i-1\}$ e $S_{ij}^+ = S_{i-1,l}^+ \setminus \{l\}$.
- Se $l < i-1$ e $i-1 \notin S_{i-1,l}^-$,
então $S_{ij}^- = S_{i-1,l}^-$ e $S_{ij}^+ = S_{i-1,l}^+ \setminus \{l\} \cup \{i-1\}$.
- Se $l = i-1$,
então $S_{ij}^- = S_{i-1,l}^-$ e $S_{ij}^+ = S_{i-1,l}^+$.
- Se $l > i-1$ e $i-1 \in S_{i-1,l}^-$,
então $S_{ij}^- = S_{i-1,l}^- \setminus \{i-1\} \cup \{l\}$ e $S_{ij}^+ = S_{i-1,l}^+$.
- Se $l > i-1$ e $i-1 \notin S_{i-1,l}^-$,
então $S_{ij}^- = S_{i-1,l}^- \cup \{l\}$ e $S_{ij}^+ = S_{i-1,l}^+ \cup \{i-1\}$.

Para finalizar, o custo de um arco entre os vértices $(i-1, l, S_{i-1,l}^-, S_{i-1,l}^+)$ e $(i, j, S_{ij}^-, S_{ij}^+)$ em G^* é o custo do arco entre os vértices l e j de G (c_{lj}).

O tempo de execução para resolver o problema do caminho mínimo no grafo auxiliar construído para o problema do TSP restrito é $O(k^2 2^{k-2} n)$. Ou seja, o

problema está é tratável quando o parâmetro k é fixo e pequeno (FPT - *Fixed Parameter Tractable*).

Com isso, finalizamos a descrição do algoritmo de Balas para o TSP restrito. No entanto, Balas generalizou o algoritmo para o caso onde a restrição de que i preceda a cidade j em uma solução viável sempre que $j \geq i + k(i)$. Ou seja, cada vértice i pode ter um k diferente. Em seguida vamos descrever o algoritmo de Balas restrito e generalizado para este caso.

Para que uma posição de uma cidade j seja igual a i em uma solução viável ($\pi(j) = i$), é necessário e suficiente que i esteja no seguinte intervalo $\max\{1, j - |P_j|\} \leq i \leq \min\{n, j + k(j) - 1\}$, sendo $P_j = \{l \in N : l < j \text{ e } l + k(l) \geq j + 1\}$.

Vamos acompanhar de perto a demonstração de Balas para as desigualdades acima.

Necessidade. Primeiro note que se $i \geq j$, então $\max\{1, j - |P_j|\} \leq i$ é satisfeita. Se $i \leq j$, então $i \leq \min\{n, j + k(j) - 1\}$ é satisfeita. Logo, falta mostrar que a primeira inequação é válida quando $i < j$ e a segunda inequação é válida quando $i > j$.

Quando $i > j$, pelo princípio da casa dos pombos, é fato que teremos algum valor entre 1 e $i - 1$ sendo a posição de alguma cidade $l > i$. Logo, $\pi(j) > \pi(l)$ e assim

$$i < l < j + k(j) - 1,$$

sendo que a segunda desigualdade vale pois precisamos atender ao TSP restrito generalizado onde temos se $\pi(l) \leq \pi(j)$ então $l \leq j + k(j) - 1$.

Quando $i < j$, então teremos $\pi(l) > \pi(j)$ para $j - i$ cidades l tais que $l < j$. Vamos ver o porquê. As cidades de N_{j-1} podem ocupar até $i - 1$ posições k tal que $\pi(k) < \pi(j)$. As outras cidades de N_{j-1} , ou seja $j - 1 - (i - 1) = j - i$ cidades, ocupam posições depois de i . Dessa forma, para uma cidade l dentre essa $j - i$ cidades, temos $\pi(l) > \pi(j)$ como queríamos mostrar. Mas, $\pi(l) > \pi(j)$ implica que $j \leq l + k(l) - 1$ (TSP restrito). Assim, o conjunto de cidades $l < j$ tal que $\pi(l) > \pi(j)$ é exatamente P_j . Portanto, $|P_j| \geq j - i$ ou $i \geq j - |P_j|$.

Suficiência. Considere que para algum i e j vale que $\max\{1, j - |P_j|\} \leq i \leq \min\{n, j + k(j) - 1\}$. A demonstração segue construindo soluções viáveis do TSP restrito e generalizado com $\pi(j) = i$. Se $i = j$, então $\pi(j) = j$ para todo $j \in N$ é uma solução viável.

Se $i > j$, então uma solução viável é a seguinte $\pi(l) = l$ para $l = 1, 2, \dots, j - 1$ e $l = i + 1, i + 2, \dots, n$, $\pi(j) = i$ e $\pi(l) = l - 1$ para $l = j + 1, j + 2, \dots, i$. Veja um exemplo na Figura 1.

Se $i < j$, então

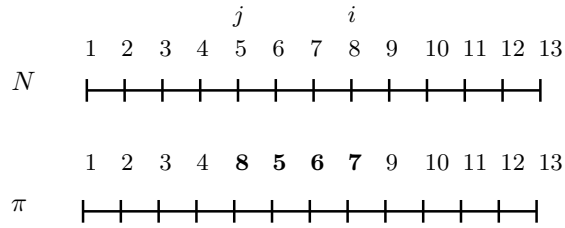


Figura 1: Caso com $i = 8$ e $j = 5$.

Referências

- [1] Egon Balas. «New classes of efficiently solvable generalized traveling salesman problems». Em: *Annals of Operations Research* 86.0 (1999), pp. 529–558.
- [2] Egon Balas e Neil Simonetti. «Linear time dynamic-programming algorithms for new classes of restricted TSPs: A computational study». Em: *INFORMS journal on Computing* 13.1 (2001), pp. 56–75.