Simulação e Otimização

Capítulo 2: Problemas de otimização combinatória

Raquel Bernardino

rbernardino@iseg.ulisboa.pt
Gabinete 511 Quelhas 6

Sumário

Motivação

Introdução

O problema do caixeiro viajante

O problema de roteamento de veículos

Sumário

Motivação

Introdução

O problema do caixeiro viajante

O problema de roteamento de veículos

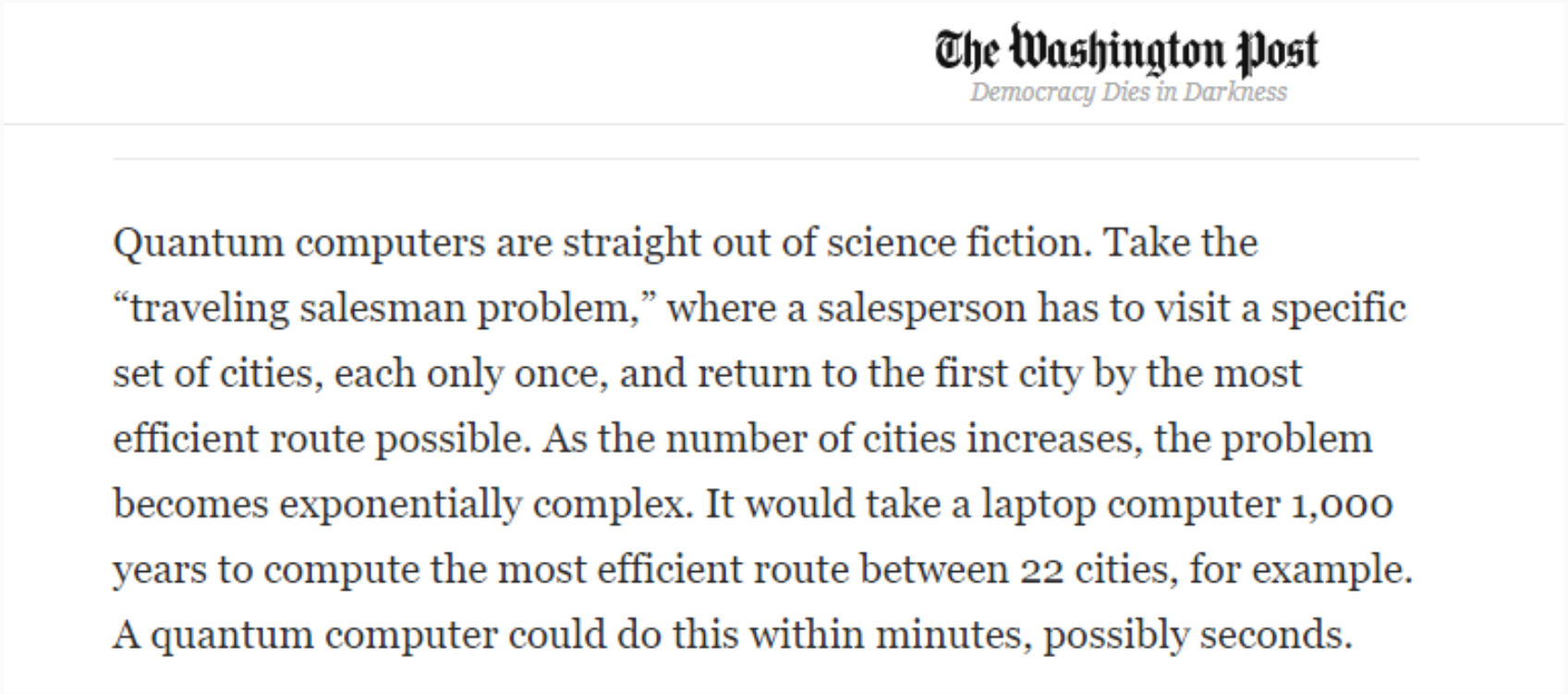
Motivação

Dado:

- um conjunto de cidades N; e
- custos de deslocação entre cada par de cidades c_{ij}, $i, j \in N : i \neq j$,

o Problema do Caixeiro Viajante consiste em determinar um circuito (ou ciclo) de menor custo que passe por todas as cidades uma e uma só vez.

Motivação



The Washington Post
Democracy Dies in Darkness

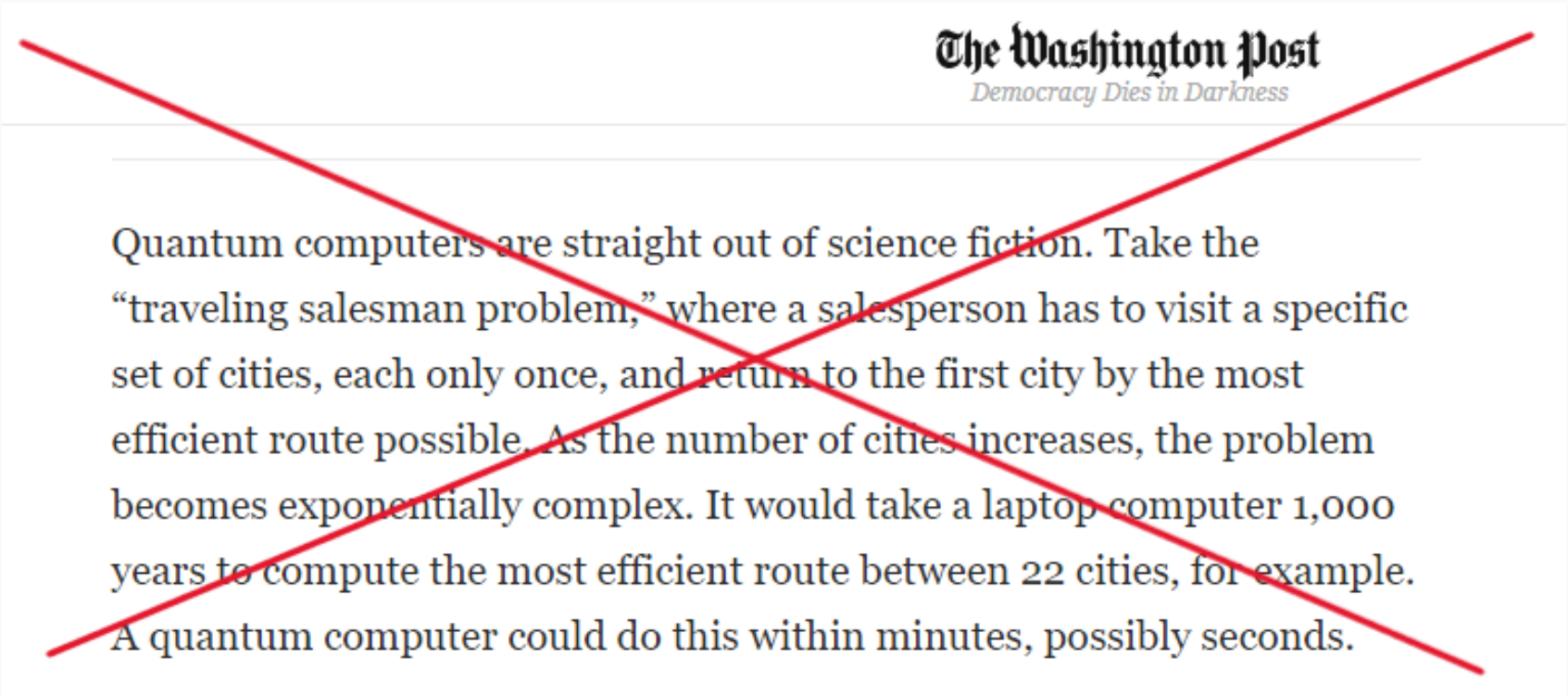
Quantum computers are straight out of science fiction. Take the "traveling salesman problem," where a salesperson has to visit a specific set of cities, each only once, and return to the first city by the most efficient route possible. As the number of cities increases, the problem becomes exponentially complex. It would take a laptop computer 1,000 years to compute the most efficient route between 22 cities, for example. A quantum computer could do this within minutes, possibly seconds.

Figura 1: Notícia errada sobre o problema caixeiro viajante no "The Washington Post".

Palestra de William Cook sobre o problema do caixeiro viajante:
`https://www.youtube.com/watch?v=5VjphFYQKj8&t=55s` [Euro 2019, Dublin]

Site com um *solver* para o caixeiro viajante:
`https://www.math.uwaterloo.ca/tsp/index.html`

Motivação



The Washington Post
Democracy Dies in Darkness

Quantum computers are straight out of science fiction. Take the "traveling salesman problem," where a salesperson has to visit a specific set of cities, each only once, and return to the first city by the most efficient route possible. As the number of cities increases, the problem becomes exponentially complex. It would take a laptop computer 1,000 years to compute the most efficient route between 22 cities, for example. A quantum computer could do this within minutes, possibly seconds.

Figura 2: Notícia errada sobre o problema caixeiro viajante no "The Washington Post".

Palestra de William Cook sobre o problema do caixeiro viajante:
`https://www.youtube.com/watch?v=5VjphFYQKj8&t=55s` [Euro 2019, Dublin]

Site com um *solver* para o caixeiro viajante:
`https://www.math.uwaterloo.ca/tsp/index.html`

Motivação

- 1954 Dantzig, Ford & Fulkerson [1] 49 cidades

Figura 3: O problema do caixeiro viajante com 49 cidades.

[1] Dantzig, G., Fulkerson, R., & Johnson, S. (1954). *Solution of a large-scale traveling-salesman problem*. Journal of the Operations Research Society of America, 2(4), 393-410.

Sumário

Motivação

Introdução

O problema do caixeiro viajante

O problema de roteamento de veículos

Introdução

- Os problemas de roteamento consistem em estabelecer rotas com determinadas características de forma a otimizar um determinado critério.

- Metodologia:
  - Métodos exatos
    - Modelos de programação linear inteira (mista). → Foco do capítulo.
  - Métodos aproximados
    - Relaxações.
    - Heurísticas.

- Os métodos exatos serão implementados utilizando software específico.
  - O pacote PuLP do Python com o *solver* da COIN-OR resolve os modelos de PLIM.
    - Exemplos de outros *solvers* são: CPLEX, Gurobi, Xpress, OpenSolver.

Introdução

- Definição genérica de um problema de roteamento

Queremos determinar uma rota que:

- comece e termine num ponto,
- passe por um subconjunto de clientes satisfazendo as suas necessidades, e
- minimize o "custo".

- Uma rota é uma sequência de vértices que começa e acaba num depósito.

Características dos problemas de roteamento

- Rotas:
  - Rotas fechadas ou abertas (começam e acabam em depósitos diferentes).
  - Rotas equilibradas - visitam o mesmo número de clientes.
  - Rotas "giras" - compactas, não se intersetam.

- Clientes:
  - Apenas alguns clientes requerem serviço.
  - Clientes têm que ser servidos num determinado período de tempo - janela temporal.
  - Um tipo de clientes pode ter prioridade sobre outros tipos de clientes - restrições de precedência.
  - Clientes têm oferta que é usada para satisfazer a procura de outros clientes - *pick-up and delivery*.
  - Clientes requerem visitas periódicas ou consistentes.

Características dos problemas de roteamento

- Frota:
  - Um ou vários veículos.
    - Com vários veículos: exatamente, pelo menos ou no máximo.
  - É homogénea ou heterogénea.
  - Veículos têm limites à capacidade do que podem transportar ou à distância percorrida - veículos elétricos.
  - Cada veículo pode fazer uma ou várias rotas.

- Objetivos:
  - Minimizar o custo.
  - Minimizar o número de veículos utilizados.
  - Minimizar a emissão de gases.
  - Maximizar o lucro recolhido.

Aplicações práticas

- Exemplos de aplicações práticas de problemas de roteamento são:
  - Problemas de distribuição,
  - Problemas de recolha,
  - Sequenciamento de ADN,
  - Escalonamento de cirurgias por médico,
  - Transporte de cavalos da GNR,
  - Operações num porto marítimo - escalonamento da grua,
  - Cuidados de saúde em casa,
  - Roteamento de drones,
  - Recolha de lixo,
  - Remoção de neve.
- Os últimos problemas são modelados como problemas de roteamento nos arcos.

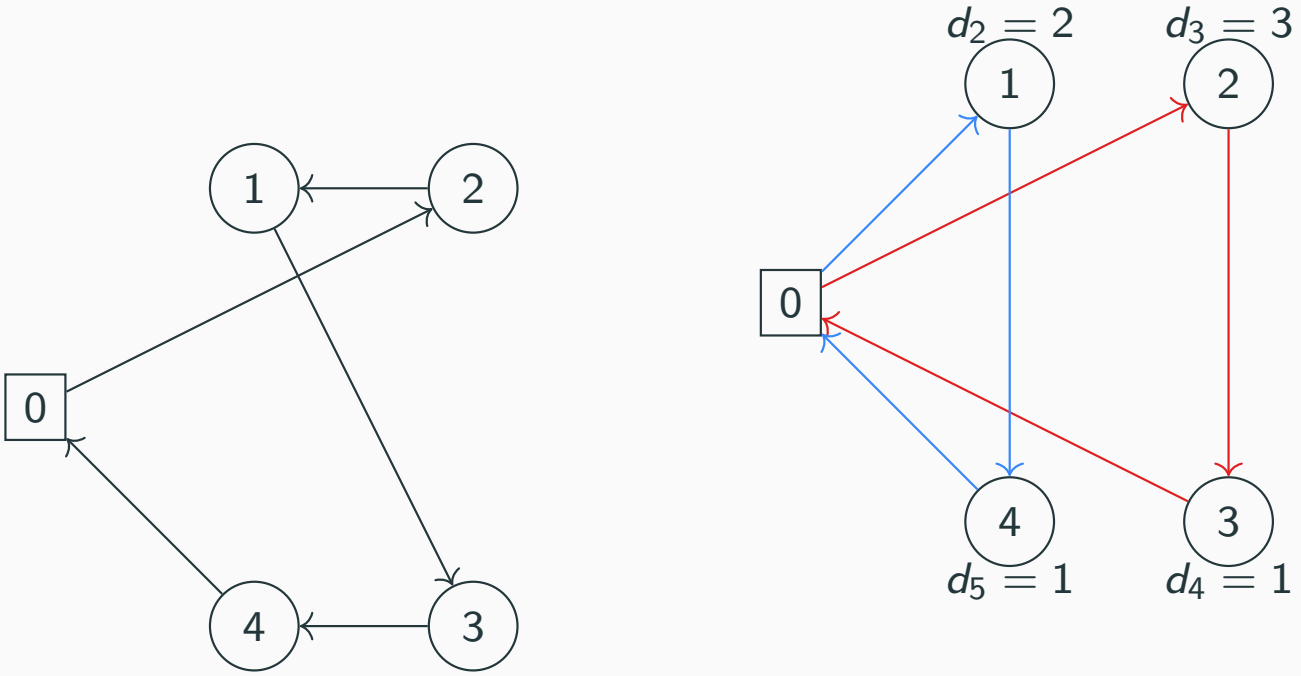
Problemas de roteamento

- Existem inúmeras variantes de problemas de roteamento. $\Longrightarrow$ Foco nas variantes base.

- Problema do caixeiro viajante (*traveling salesman problem - TSP*)
  - uma rota;
  - visita todos os clientes uma e uma só vez; e
  - o objetivo é minimizar o custo total de deslocação.

- Problema do roteamento de veículos (*vehicle routing problem - VRP*)
  - cada um dos K veículos homogéneos disponíveis executa uma rota;
  - cada veículo tem capacidade Q;
  - cada cliente tem procura que deve ser toda satisfeita; e
  - o objetivo é minimizar o custo total de deslocação.

Problemas de roteamento



(a) TSP. **(b)** VRP ($K = 2$ e $Q = 4$).

Figura 4: Soluções do TSP e do VRP.

Sumário

Motivação

Introdução

O problema do caixeiro viajante

- Formulações em programação linear inteira (mista)
- Relaxações
- Heurísticas

O problema de roteamento de veículos

O problema do caixeiro viajante

- O problema do caixeiro viajante pode ser modelado num grafo orientado $G = (V, A)$, onde:
  - V é o conjunto das cidades que devem ser visitadas uma e uma só vez;
    - Uma das cidades $0 \in V$ é o depósito fictício - indica o início e o fim do percurso.
  - A é o conjunto dos arcos, que representam as ligações entre as cidades.
    - $A = \{(i, j) : i, j \in V \text{ e } i \neq j\}$
  - c_{ij} é o custo de atravessar o arco $(i, j) \in A$.

- Queremos determinar o circuito de custo total mínimo que passa por todos os nodos de V uma e uma só vez (circuito Hamiltoniano).

Formulação genérica

$$x_{ij} = \begin{cases} 1, & \text{se o arco } (i,j) \in A \text{ é atravessado pelo caixeiro viajante} \\ 0, & \text{caso contrário} \end{cases}$$

$$\min \sum_{(i,j)\in A} c_{ij} x_{ij}$$

sujeito a:

$$\sum_{i \in V} x_{ji} = 1 \qquad \forall j \in V \quad \text{(um arco sai de j)}$$

$$\sum_{i \in V} x_{ij} = 1 \qquad \forall j \in V \quad \text{(um arco entra j)}$$

Não contém subcircuitos

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

Nota: Subcircuitos (ou subrotas) são circuitos que não contêm o depósito.

Problemas de roteamento

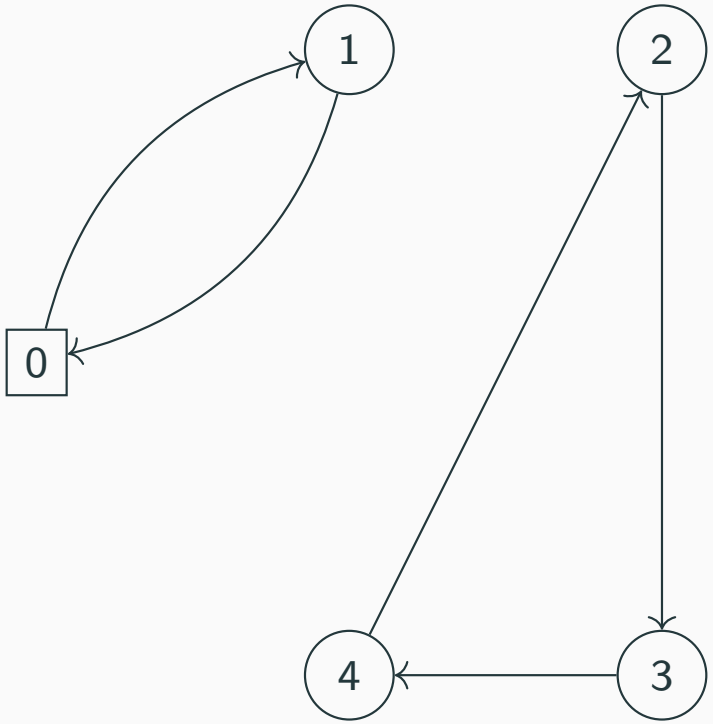


Figura 5: Solução não admissível para o TSP.

Como garantir que não temos subcircuitos?

Formulação MTZ

Formulação de Miller-Tucker-Zemlin (MTZ) [2]

u_i - posição que o nodo $i \in V \setminus \{0\}$ ocupa na rota.

$$\min \sum_{(i,j)\in A} c_{ij} x_{ij}$$

sujeito a:

$$\sum_{i\in V} x_{ji} = 1 \qquad \forall j \in V \qquad \text{(um arco sai de j)}$$

$$\sum_{i\in V} x_{ij} = 1 \qquad \forall j \in V \qquad \text{(um arco entra de j)}$$

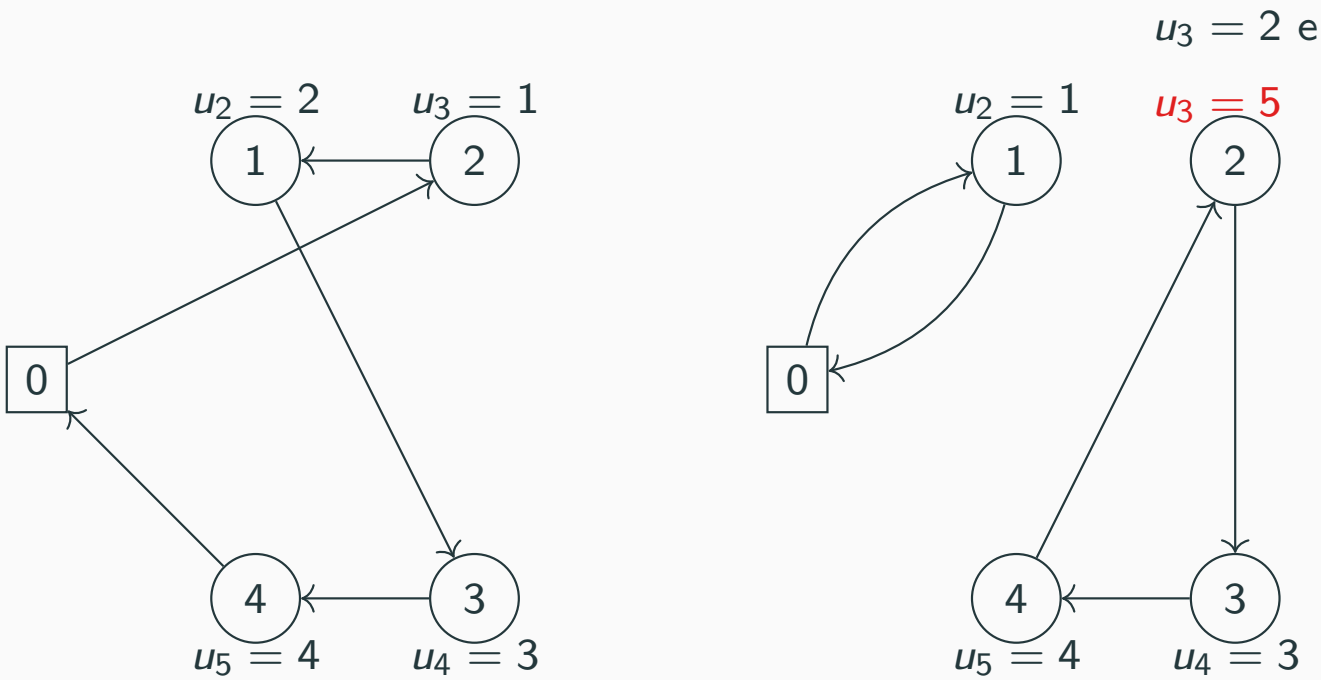
$$u_i + (|N| - 1)x_{ij} \leq u_j + |N| - 2 \qquad \forall (i,j) \in A : i,j \neq 0 \qquad \text{(posição j é a do i mais 1, se vamos de i para j)}$$

$$1 \leq u_i \leq |N| - 1 \qquad \forall i \in V \setminus \{0\} \qquad \text{(limites variáveis u)}$$

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

[2] Miller, C. E., Tucker, A. W., & Zemlin, R. A. (1960). *Integer programming formulation of traveling salesman problems*. Journal of the ACM (JACM), 7(4), 326-329.

Formulação MTZ



(a) Solução admissível TSP.

(b) Solução não admissível TSP.

Figura 6: Como funciona o MTZ.

Formulação MTZ

- Consideremos $V = \{0, 1, 2\}$.

$$\min \; c_{01}x_{01} + c_{02}x_{02} + c_{10}x_{10} + c_{12}x_{12} + c_{20}x_{20} + c_{21}x_{21}$$

sujeito a:

$$x_{01} + x_{02} = 1 \qquad \text{(sai 0)}$$

$$x_{10} + x_{12} = 1 \qquad \text{(sai 1)}$$

$$x_{20} + x_{21} = 1 \qquad \text{(sai 2)}$$

$$x_{10} + x_{20} = 1 \qquad \text{(entra 0)}$$

$$x_{01} + x_{21} = 1 \qquad \text{(entra 1)}$$

$$x_{02} + x_{12} = 1 \qquad \text{(entra 2)}$$

$$u_1 + 2x_{12} \leq u_2 + 1 \qquad \text{(posição do 2 é a do 1 mais 1, se vamos de 1 para 2)}$$

$$u_2 + 2x_{21} \leq u_1 + 1 \qquad \text{(posição do 1 é a do 2 mais 1, se vamos de 2 para 1)}$$

$$1 \leq u_1, u_2 \leq 2 \qquad \text{(limites u)}$$

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

Formulação SCF

Formulação de fluxo único - Single Commodity Flow (SCF) [3]

f_{ij} - fluxo que atravessa o arco $(i,j) \in A$, que corresponde ao número de nodos que ainda têm que ser visitados.

$$\min \sum_{(i,j)\in A} c_{ij} x_{ij}$$

sujeito a:

$$\sum_{i\in V} x_{ji} = 1 \qquad \forall j \in V \qquad \text{(um arco sai de j)}$$

$$\sum_{i\in V} x_{ij} = 1 \qquad \forall j \in V \qquad \text{(um arco entra de j)}$$

$$\sum_{i\in V\setminus\{0\}} f_{0i} = |N| - 1 \qquad \text{(sai } |N| - 1 \text{ fluxo de 0)}$$

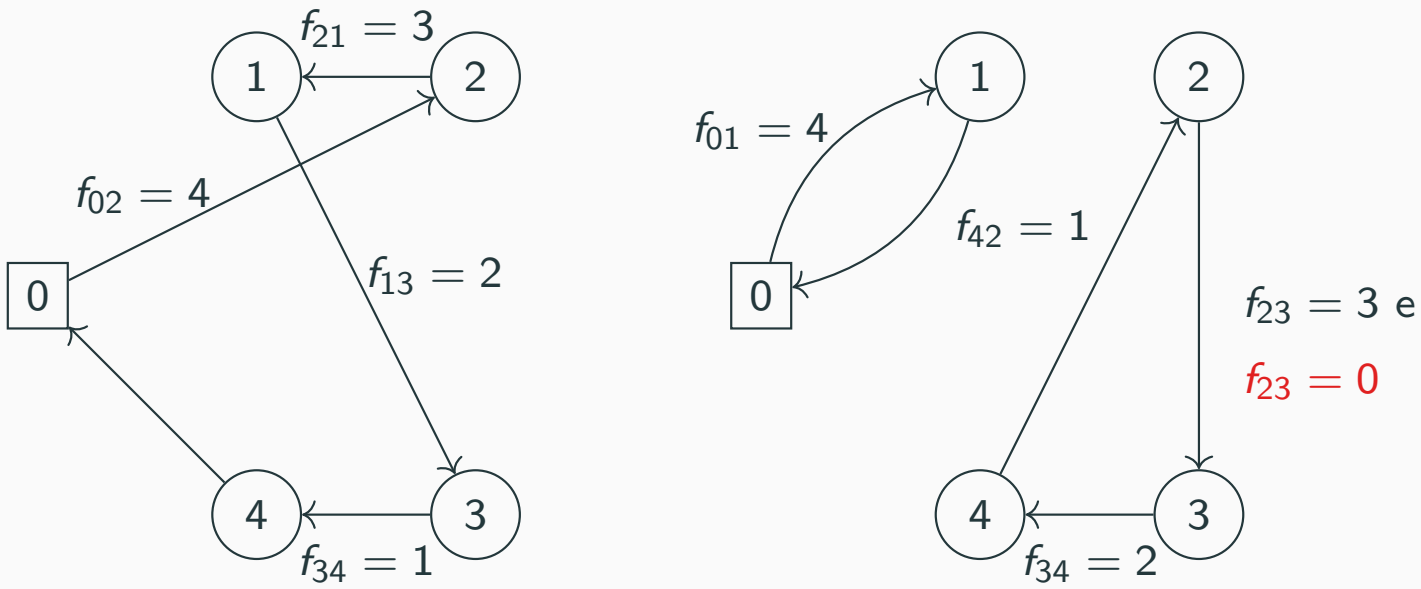
$$\sum_{j\in V:j\neq i} f_{ji} = \sum_{j\in V:j\neq i} f_{ij} + 1 \qquad \forall i \in V \setminus \{0\} \qquad \text{(fluxo entra = fluxo sai + fluxo fica)}$$

$$0 \leq f_{ij} \leq (|N| - 1)x_{ij} \qquad \forall (i,j) \in N \qquad \text{(limites var. fluxo)}$$

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

[3] Gavish, B., & Graves, S. C. (1978). *The travelling salesman problem and related problems*. Working paper GR-078-78, Operations Research Center, Massachusetts Institute of Technology.

Formulação SCF



(a) Solução admissível TSP.

(b) Solução não admissível TSP.

Figura 7: Como funciona o SCF.

Formulação SCF

- Consideremos $V = \{0, 1, 2\}$.

$$\min \; c_{01}x_{01} + c_{02}x_{02} + c_{10}x_{10} + c_{12}x_{12} + c_{20}x_{20} + c_{21}x_{21}$$

sujeito a: (...)

$$f_{01} + f_{02} = 2 \qquad \text{(sai 2 de fluxo do depósito)}$$

$$f_{01} + f_{21} = f_{10} + f_{12} + 1 \qquad \text{(cons. fluxo 1)}$$

$$f_{02} + f_{12} = f_{20} + f_{21} + 1 \qquad \text{(cons. fluxo 2)}$$

$$0 \leq f_{01} \leq 2x_{01} \qquad \text{(rel. f e x para (0,1))}$$

$$0 \leq f_{02} \leq 2x_{02} \qquad \text{(rel. f e x para (0,2))}$$

$$0 \leq f_{10} \leq 2x_{10} \qquad \text{(rel. f e x para (1,0))}$$

$$0 \leq f_{12} \leq 2x_{12} \qquad \text{(rel. f e x para (1,2))}$$

$$0 \leq f_{20} \leq 2x_{20} \qquad \text{(rel. f e x para (2,0))}$$

$$0 \leq f_{21} \leq 2x_{21} \qquad \text{(rel. f e x para (2,1))}$$

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

Formulação CC

Formulação de cortes de conexidade - *Connectivity cuts (CC)*[4]

$$\min \sum_{(i,j)\in A} c_{ij} x_{ij}$$

sujeito a:

$$\sum_{i\in V} x_{ji} = 1 \qquad \forall j \in V \qquad \text{(um arco sai de j)}$$

$$\sum_{i\in V} x_{ij} = 1 \qquad \forall j \in V \qquad \text{(um arco entra em j)}$$

$$\sum_{i\in V\setminus S} \sum_{j\in S} x_{ij} \geq 1 \qquad \forall S \subsetneq V : S \neq \emptyset \qquad \text{(um arco com início em S e fim em } V \setminus S \text{ tem que ser usado na solução)}$$

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

[4] Dantzig, G., Fulkerson, R., & Johnson, S. (1954). *Solution of a large-scale traveling-salesman problem*. Journal of the Operations Research Society of America, 2(4), 393-410.

Formulação CC

- As restrições $\sum_{i \in V \setminus S} \sum_{j \in S} x_{ij} \geq 1$ são em número exponencial. $\Longrightarrow$ A formulação CC é não compacta.
  - Esta formulação deve ser resolvida com recurso a um algoritmo de branch-and-cut.
    - As restrições em número exponencial podem ser utilizadas como desigualdades válidas nas outras formulações apresentadas.

Formulação CC

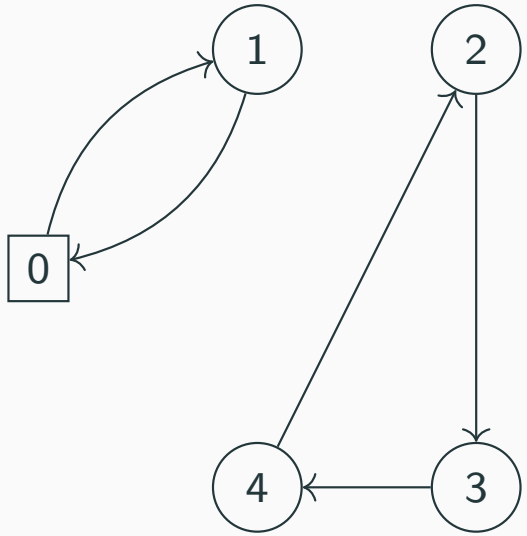


Figura 8: Solução não admissível para o TSP.

Considerando $S = \{3, 4, 5\}$, não existe nenhum arco com inicio em $V \setminus S = \{1, 2\}$ e com fim em S.

- Um corte para esta solução é $x_{13} + x_{14} + x_{15} + x_{23} + x_{24} + x_{25} \geq 1$.

Formulação CC

- Consideremos $V = \{0, 1, 2, 3\}$.

$$\begin{aligned}
\min\ & c_{01}x_{01} + c_{02}x_{02} + c_{03}x_{03} + c_{10}x_{10} + c_{12}x_{12} + c_{13}x_{13} + c_{20}x_{20} + \\
& c_{21}x_{21} + c_{23}x_{23} + c_{30}x_{30} + c_{31}x_{31} + c_{32}x_{32} \\
\text{sujeito a:}\ & (\ldots) \\
& x_{02} + x_{03} + x_{12} + x_{13} \geq 1 \\
& x_{01} + x_{03} + x_{21} + x_{23} \geq 1 \\
& x_{01} + x_{02} + x_{31} + x_{32} \geq 1 \\
& x_{10} + x_{13} + x_{20} + x_{23} \geq 1 \\
& x_{10} + x_{12} + x_{30} + x_{32} \geq 1 \\
& x_{20} + x_{21} + x_{30} + x_{31} \geq 1 \\
& x_{ij} \in \{0, 1\} \qquad \forall (i, j) \in A
\end{aligned}$$

- Que conjuntos S faltam?

Na formulação não estão presentes os conjuntos S de cardinalidade 1 e 3 porque são implicados pelas restrições de afetação.

Comparação da dimensão das formulações apresentadas

Tabela 1: Comparação de formulações em termos de número de variáveis e restrições.

Formulação	#Variáveis	#Restrições
MTZ	$\|A\| + \|V\| - 1$	$\|A\| + \|V\| - 1$
SCF	$2\|A\|$	$3\|V\| + \|A\|$
CC	$\|A\|$	$2^{\|V\|-1}$

- Qual das formulações é melhor?

Relaxações do problema do caixeiro viajante

- Qualquer formulação válida para o problema do caixeiro garante que
  1. cada nodo tem um arco a entrar e um arco a sair; e
  2. não existem subcircuitos.

- Consideremos a formulação genérica:

$$\min \sum_{(i,j)\in A} c_{ij} x_{ij}$$

$$\text{sujeito a: } \sum_{i\in V} x_{ji} = 1 \qquad \forall j \in V \quad \text{(um arco sai de j)}$$

$$\sum_{i\in V} x_{ij} = 1 \qquad \forall j \in V \quad \text{(um arco entra j)}$$

Não contém subcircuitos

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

- Podemos obter relaxações removendo cada um dos conjuntos de restrições da formulação genérica.

Relaxações do problema do caixeiro viajante

- Relaxando as restrições de eliminação de subcircuitos obtemos:

$$(\text{REL1}) \equiv \min \sum_{(i,j)\in A} c_{ij}x_{ij}$$

$$\text{sujeito a: } \sum_{i\in V} x_{ji} = 1 \qquad \forall j \in V \quad \text{(um arco sai de j)}$$

$$\sum_{i\in V} x_{ij} = 1 \qquad \forall j \in V \quad \text{(um arco entra j)}$$

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

- Que problema é este?

É o problema da afetação.

Relaxações do problema do caixeiro viajante

- Relaxando as restrições saída de um nodo obtemos:

$$(\text{REL2}) \equiv \min \sum_{(i,j)\in A} c_{ij}x_{ij}$$

$$\text{sujeito a: } \sum_{i\in V} x_{ij} = 1 \qquad \forall j \in V \quad \text{(um arco entra j)}$$

$$\text{Não contém subcircuitos}$$

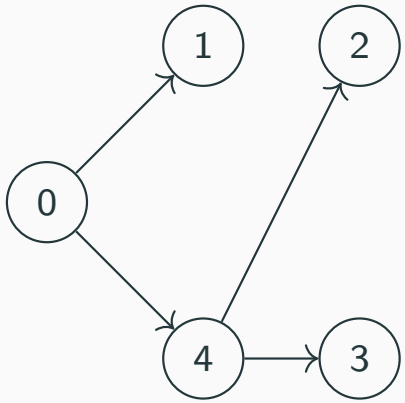
$$x_{ij} \in \{0,1\} \qquad \forall (i,j) \in A$$

- Que problema é este?

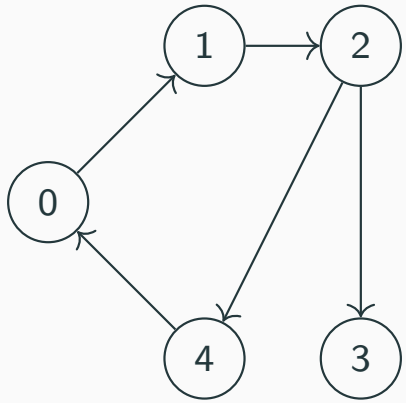
Uma SA para a formulação apresentada anteriormente tem exatamente um arco a entrar em cada nodo e é conexa.

Relaxações do problema do caixeiro viajante

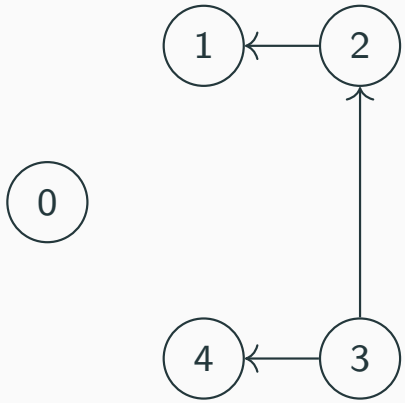
- Uma árvore é um grafo conexo e sem ciclos. Caso a árvore contenha todos os nodos do grafo original é uma árvore de suporte (ou árvore geradora).
- Uma arborescência é uma árvore com uma raiz - nodo sem "arcos a entrar" - e em que existe um único caminho da raiz para cada um dos restantes vértices da árvore. $\Longrightarrow$ Uma arborescência que contenha todos os nodos do grafo é uma arborescência geradora de G.
- Exemplos:



(a) Arborescência geradora de G.



(b) Não arborescência.



(c) Arborescência de G.

Relaxações do problema do caixeiro viajante

- Uma solução admissível da formulação REL2:

  1. contém todos os nodos do grafo,
  2. é conexa, e
  3. cada nodo tem exatamente um arco a entrar

- Se a uma arborescência geradora de G juntarmos um arco a entrar na raiz x, obtemos uma solução admissível para a REL2.

- Algoritmo para determinar minorantes baseado na REL2:

  Passo 1: Identificar em $G = (V, A)$ a arborescência de custo mínimo T como raiz em $x \in V$.

  Passo 2: Juntar a T o arco de menor custo a entrar em x. O valor de T é um minorante para o valor do TSP.

Relaxações do problema do caixeiro viajante

- O problema da afetação e o problema da arborescência de custo mínimo são problemas com um algoritmo polinomial conhecido. $\Longrightarrow$ "Problemas fáceis".

- Na prática, devem ser aplicados os algoritmos polinomiais para determinar a solução ótima dos problemas referidos.

- Como os algoritmos não estão no âmbito da disciplina, vamos resolver o problema da afetação e o problema da arborescência de custo mínimo utilizando modelos de programação linear inteira mista.

Heurística do vizinho mais próximo

Definição: Heurística do vizinho mais próximo

Começar numa cidade i^* e sucessivamente visitar a cidade mais próxima da última cidade visitada, terminado o circuito em i^*.

Heurística do vizinho mais próximo

1: Começar a rota R com uma cidade i^*.
2: Fazer $i = i^*$.
3: **while** Há cidades por visitar **do**
4: Determinar a cidade j que ainda não está na rota que minimiza c_{ij}.
5: Adicionar j ao fim da rota.
6: Fazer $i = j$.
7: **end while**

- Seja $v(V + P)$ o valor da solução obtida com a heurística do vizinho mais próximo.

Exemplo

- Seja $R = \{A\}$ e $i = A$.
- Há cidades por visitar? Sim. Cidade j que minimiza $c_{Aj} \rightarrow$ Cidade E.
  - $R = (A, E)$ e $i = E$.

	A	B	C	D	E
A	-	3	5	19	2
B	3	-	14	15	11
C	5	14	-	20	1
D	19	15	20	-	9
E	2	11	1	9	-

- Há cidades por visitar? Sim. Cidade j que minimiza $c_{Ej} \rightarrow$ Cidade C.
  - $R = (A, E, C)$ e $i = C$.
- Há cidades por visitar? Sim. Cidade j que minimiza $c_{Cj} \rightarrow$ Cidade B.
  - $R = (A, E, C, B)$ e $i = B$.
- Há cidades por visitar? Sim. Cidade j que minimiza $c_{Bj} \rightarrow$ Cidade D.
  - $R = (A, E, C, B, D)$ e $i = D$.
- Há cidades por visitar? Não. Fim.

A rota obtida com a heurística do vizinho mais próximo é (A, E, C, B, D, A) e $v(V+P) = 2 + 1 + 14 + 15 + 19 = 51 \rightarrow gap = 54,55\%$.

Heurística de inserção de menor custo

Heurística de inserção de menor custo

1: Construir a rota R que contém as duas cidades i^* e j^*, tais que, $c_{i^*j^*} = min_{ij} c_{ij}$.
2: **while** Há cidades por visitar **do**
3: Determinar para cada cidade por visitar k e para cada par de cidades visitadas consecutivamente o menor custo de inserção, isto é, $c_{ik} + c_{jk} - c_{ij}$.
4: Inserir a cidade k entre as cidades i e j que minimize o custo de inserção.
5: **end while**

- Seja $v(IMC)$ o valor da solução obtida com a heurística de inserção mais próxima.

Exemplo

	A	B	C	D	E
A	-	3	5	19	2
B	3	-	14	15	11
C	5	14	-	20	1
D	19	15	20	-	9
E	2	11	1	9	-

- Seja $R = \{\{C, E\}\}$.
- Há cidades por visitar?

  Sim. $R = \{\{C, A\}, \{A, E\}, \{E, C\}\}$
  - Cidade A: $c_{CA} + c_{AE} - c_{CE} = 5 + 2 - 1 = 4.$ $\leftarrow$ min
  - Cidade B: $c_{CB} + c_{BE} - c_{CE} = 14 + 11 - 1 = 24.$
  - Cidade D: $c_{CD} + c_{DE} - c_{CE} = 20 + 9 - 1 = 28.$

- Há cidades por visitar? Sim. $R = \{\{C, B\}, \{B, A\}, \{A, E\}, \{E, C\}\}$
  - Cidade B:
    - Aresta $\{C, A\}$: $c_{CB} + c_{BA} - c_{CA} = 14 + 3 - 5 = 12.$ $\leftarrow$ min
    - Aresta $\{A, E\}$: $c_{AB} + c_{BE} - c_{AE} = 3 + 11 - 2 = 12.$ $\leftarrow$ min
    - Aresta $\{E, C\}$: $c_{EB} + c_{BC} - c_{EC} = 11 + 14 - 1 = 24.$
  - Cidade D:
    - Aresta $\{C, A\}$: $c_{CD} + c_{DA} - c_{CA} = 20 + 19 - 5 = 34.$
    - Aresta $\{A, E\}$: $c_{AD} + c_{DE} - c_{AE} = 19 + 9 - 2 = 26.$
    - Aresta $\{E, C\}$: $c_{ED} + c_{DC} - c_{EC} = 9 + 20 - 1 = 28.$

Exemplo (continuação)

- Há cidades por visitar? Sim. $R = \{\{C, D\}, \{D, B\}, \{B, A\}, \{A, E\}, \{E, C\}\}$
  - Cidade D:
    - Aresta $\{C, B\}$: $c_{CD} + c_{DB} - c_{CB} = 20 + 15 - 14 = 21$. $\leftarrow$ min
    - Aresta $\{B, A\}$: $c_{BD} + c_{DA} - c_{BA} = 15 + 19 - 3 = 31$.
    - Aresta $\{A, E\}$: $c_{AD} + c_{DE} - c_{AE} = 19 + 9 - 2 = 26$.
    - Aresta $\{E, C\}$: $c_{ED} + c_{DC} - c_{EC} = 9 + 20 - 1 = 28$.
- Há cidades por visitar? Não. Fim.

A rota obtida com a heurística de inserção de menor custo é (C, D, B, A, E, C) e $v(IMC) = 20 + 15 + 3 + 2 + 1 = 41 \rightarrow gap = 24,24\%$.

Notas sobre o problema do caixeiro viajante

- O problema do caixeiro viajante é muito estudado na comunidade científica.
- Tem inúmeras variantes, como por exemplo:
  - Problema do caixeiro viajante com precedências.
  - Problema do caixeiro viajante com múltiplos depósitos.
  - Problema do caixeiro viajante com lucros.
  - Problema do comprador viajante.
  - Problema do caixeiro viajante com famílias.