Otimização: O melhor

Modelos Matemáticos para Tomada de Decisões

Luidi Gelabert Simonetti
luidi@cos.ufrj.br

PESC - COPPE - UFRJ

2017

Otimização

Otimização

Dicionário: otimização

1. criação de condições mais favoráveis para o desenvolvimento de algo.

2. processo através do qual se obtém o melhor valor de uma grandeza.

3. reestruturação efetuada com o objetivo de obter o maior rendimento possível

Otimização - Pesquisa Operacional

- É um ramo interdisciplinar da matemática aplicada que faz uso de modelos matemáticos, estatísticos e de algoritmos na ajuda à tomada de decisões

- É usada sobretudo para analisar sistemas complexos do mundo real, tipicamente com o objetivo de melhorar ou otimizar a performance

Origem

- A investigação operacional nasceu durante a II guerra mundial, quando os Aliados se viram confrontados com problemas (logística e militar) de grande dimensão e complexidade.
- Aplicaram o método científico aos problemas que lhes foram sendo colocados e criaram modelos matemáticos que lhes permitissem avaliar o resultado hipotético de estratégias ou decisões alternativas.
- Com o fim do conflito e sucesso obtido, os grupos de cientistas transferiram a nova metodologia na abordagem de problemas para as empresas.
- Com a evolução observada na informática criaram-se condições de concretização algorítmica e velocidade de processamento adaptados à imaginação dos profissionais da pesquisa operacional.

Otimização

Exemplo: Otimizar a produtividade

Nessa posição corrijo 2 trabalhos por hora

Otimização

Exemplo: Otimizar a produtividade

Nessa posição corrijo 3 trabalhos por hora

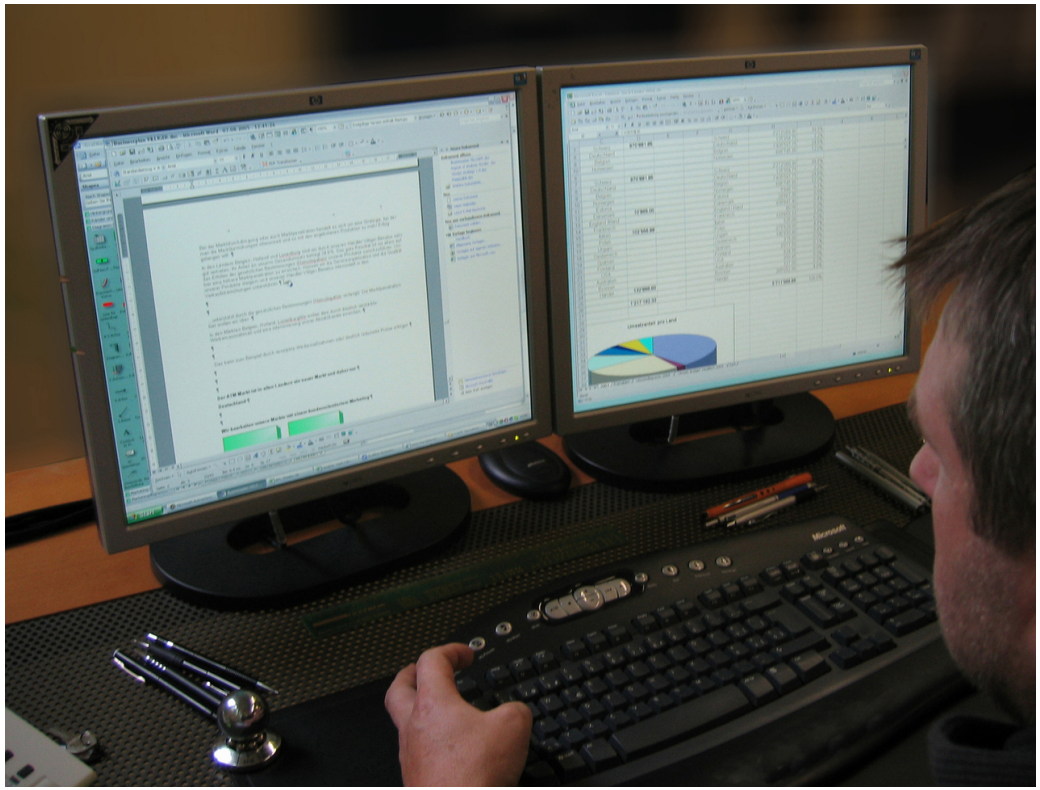
Otimização

Exemplo: Otimizar a produtividade

Nessa posição corrijo 3,8 trabalhos por hora

Otimização

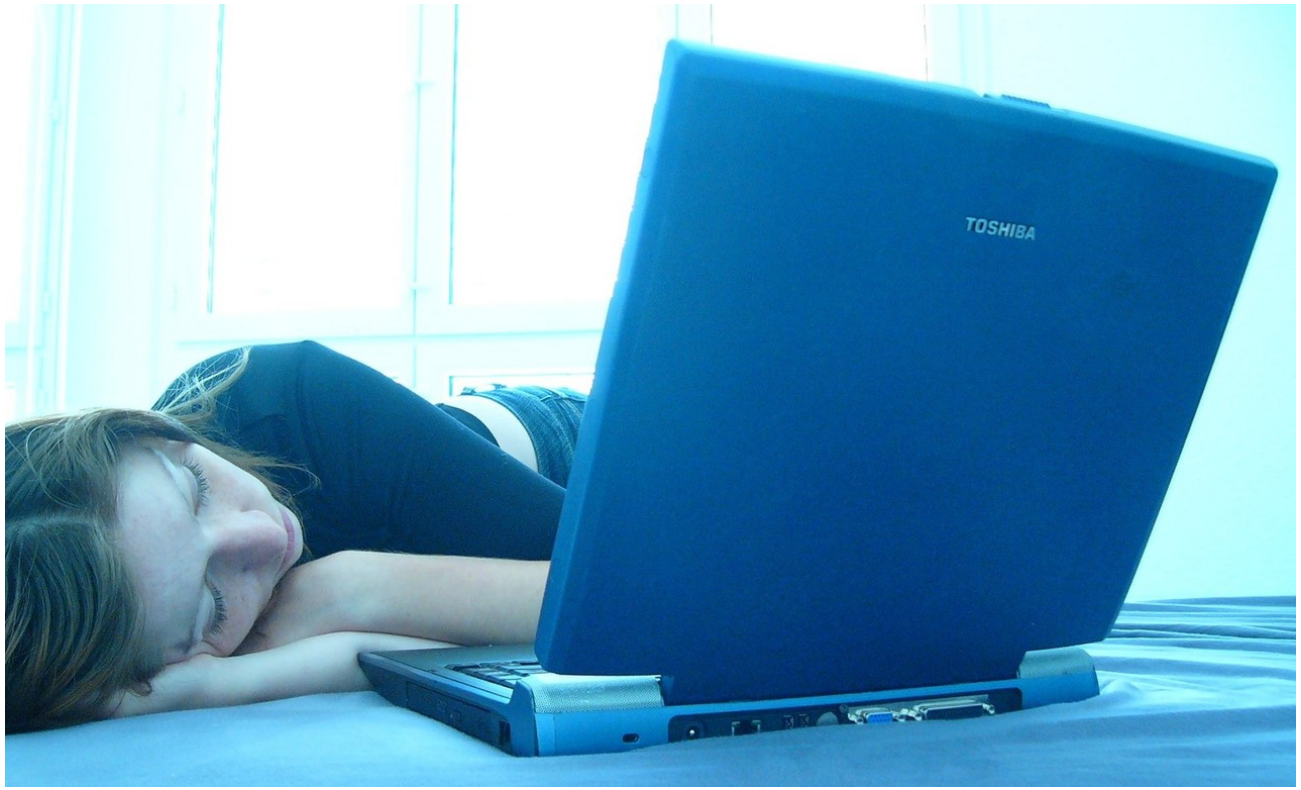
Exemplo: Otimizar a produtividade



Agora corrijo 15 trabalhos por hora!

Otimização

Exemplo: Otimizar a produtividade



Melhor!!! Corrijo tudo automaticamente!!!

Otimização

O Melhor x Melhorar

- Tentar melhorar uma solução já conhecida também é otimizar.
- Num mundo capitalista onde sempre queremos ser o melhor e sempre queremos mais, por que não tentar achar o melhor?
- Se sabemos que temos a melhor solução não temos como melhorá-la.

Otimização

Exemplos

- **Maximizar** PAZ ????
- **Maximizar** Minha produtividade ????
- **Minimizar** Pobreza ????
- Tem que existir as opções ou regras.
- Se não o problema não está bem definido.

Otimização

Exemplos

- **Maximizar** PAZ ????
- **Maximizar** Minha produtividade ????
- **Minimizar** Pobreza ????
- Tem que existir as opções ou regras.
- Se não o problema não está bem definido.

Otimização

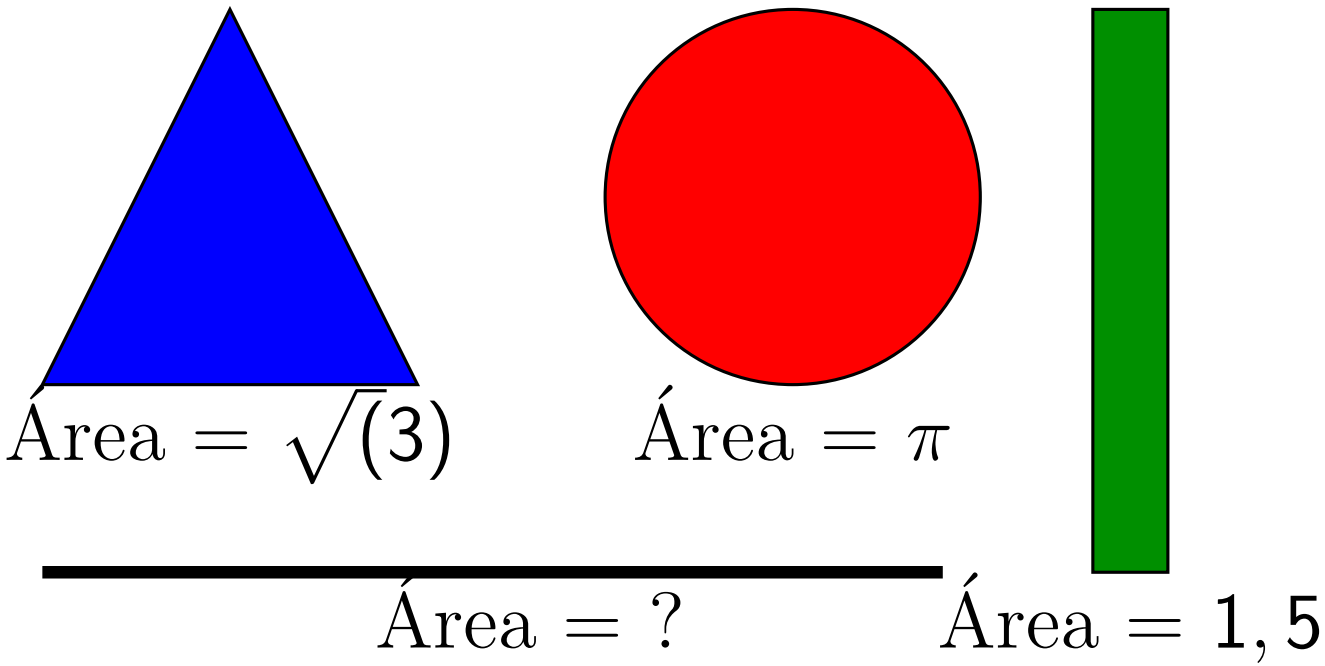
Estudo de opções

- Uma possível estratégia quando o problema é muito difícil ou não tem limites.
- Exemplo: aumentar a minha produtividade.
- A cada dia inventam novas técnicas e tecnologias para aumentar a produtividade.
- Como modelar algo que não sabemos ou que não existe.
- Podemos criar opções, testá-las e usar técnicas para avaliar quais as melhores.

Melhor?

Função objetivo

- Escolher o Maior



Melhor?

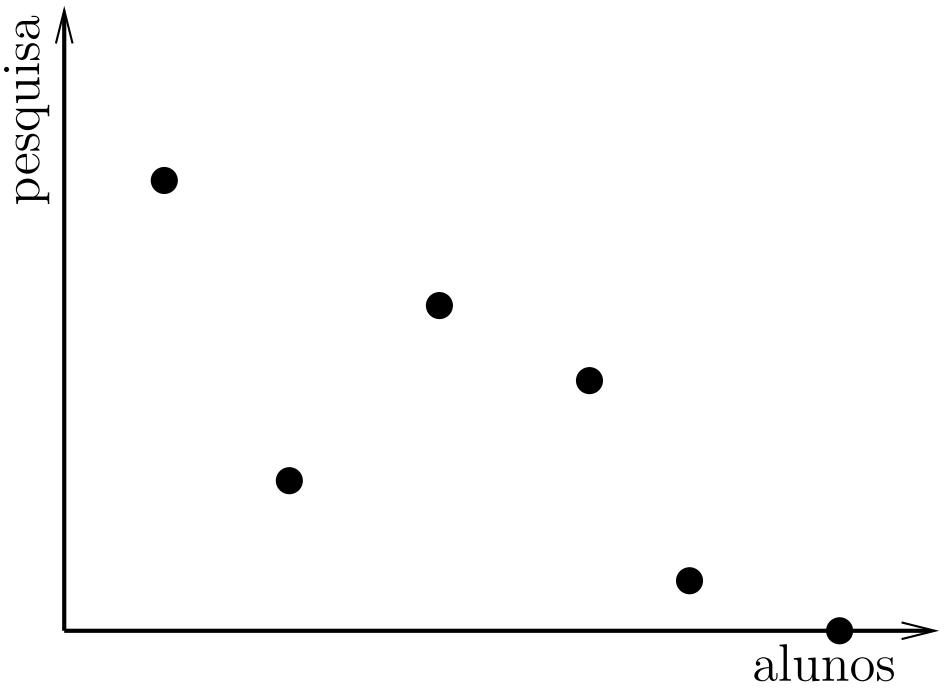
Função objetivo

- Pode ser simples.
  - max Lucro
  - min Desperdício
- Combinação de funções
  - max α(Lucro) - β(Impacto Ambiental)
  - Depende dos valores de α e β
  - $\alpha \gg \beta$ - Maximizar Lucro e Impacto Ambiental é um critério de desempate
  - $\alpha \ll \beta$ - Minimizar Impacto Ambiental e Lucro é um critério de desempate
- Multiobjetivo - não tem uma hierarquia entre as funções objetivas.
  - max Anos de vida, max Qualidade, max Felicidade, min max Triteza

Melhor?

Exemplo: DEA (Data envelopment analysis)

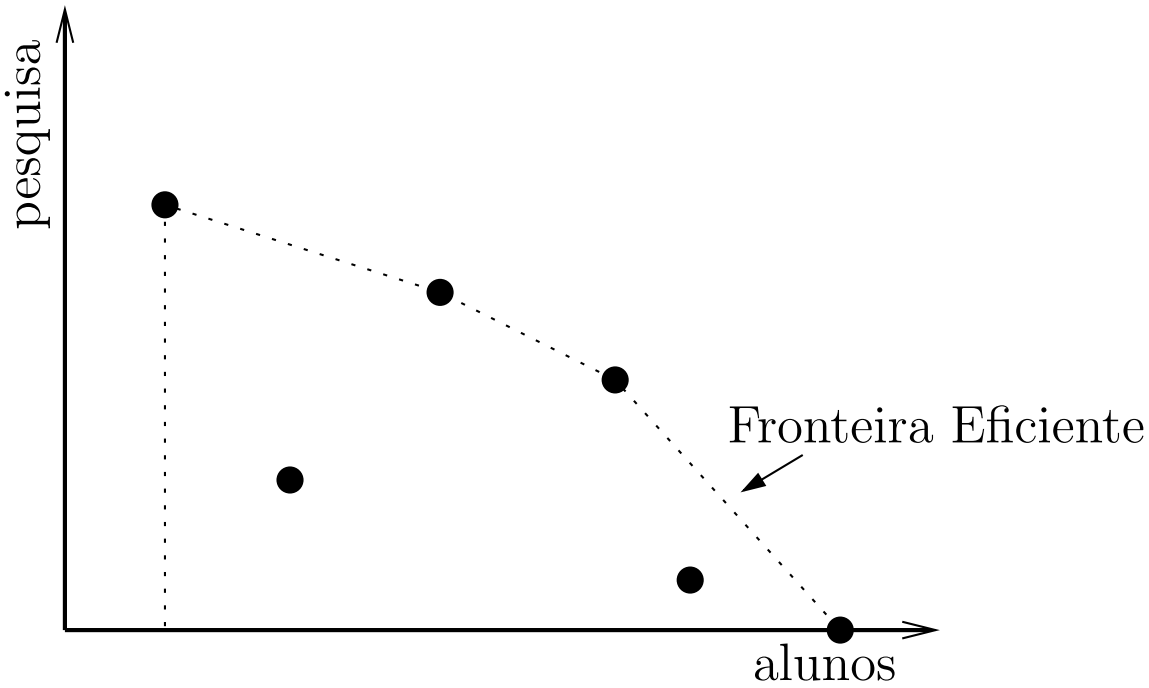
- Queremos avaliar 6 universidades
- Vamos usar os critérios: alunos formados e pesquisa (publicações, patentes, etc) per capito



Melhor?

Exemplo: DEA (Data envelopment analysis)

- Qual o melhor peso para cada critério?
- Por que não deixar cada universidade escolher o melhor peso para ela?



Otimização

Exemplo

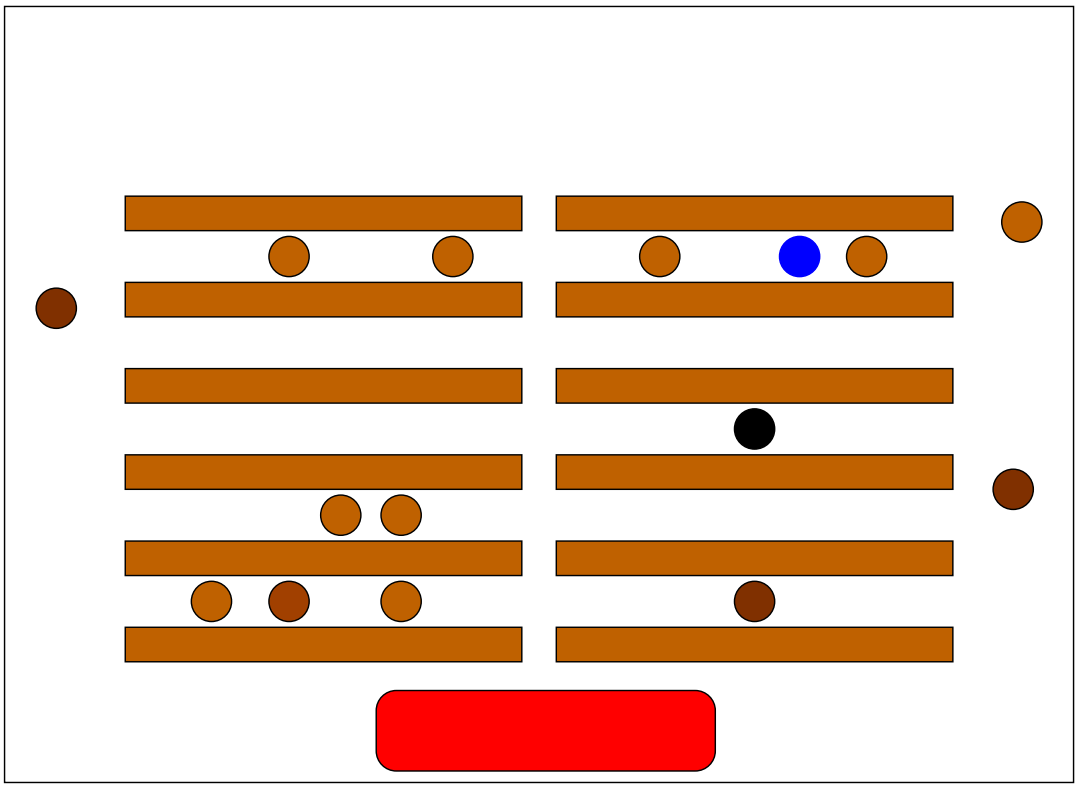
- **Minimizar** Tempo de deslocamento de um assento até o palco

Otimização

Exemplo - Caminho Mínimo

- No espaço euclidiano o caminho mais curto não é uma reta?
- Posso passar por cima das cadeiras e/ou outras pessoas sentadas?
- Consigo passar entre as cadeiras ou obstáculos?
- As pessoas vão ter que se levantar ou mudar de posição para eu conseguir passar?
- A velocidade de deslocamento é constante? Não importa o lugar?
- ...

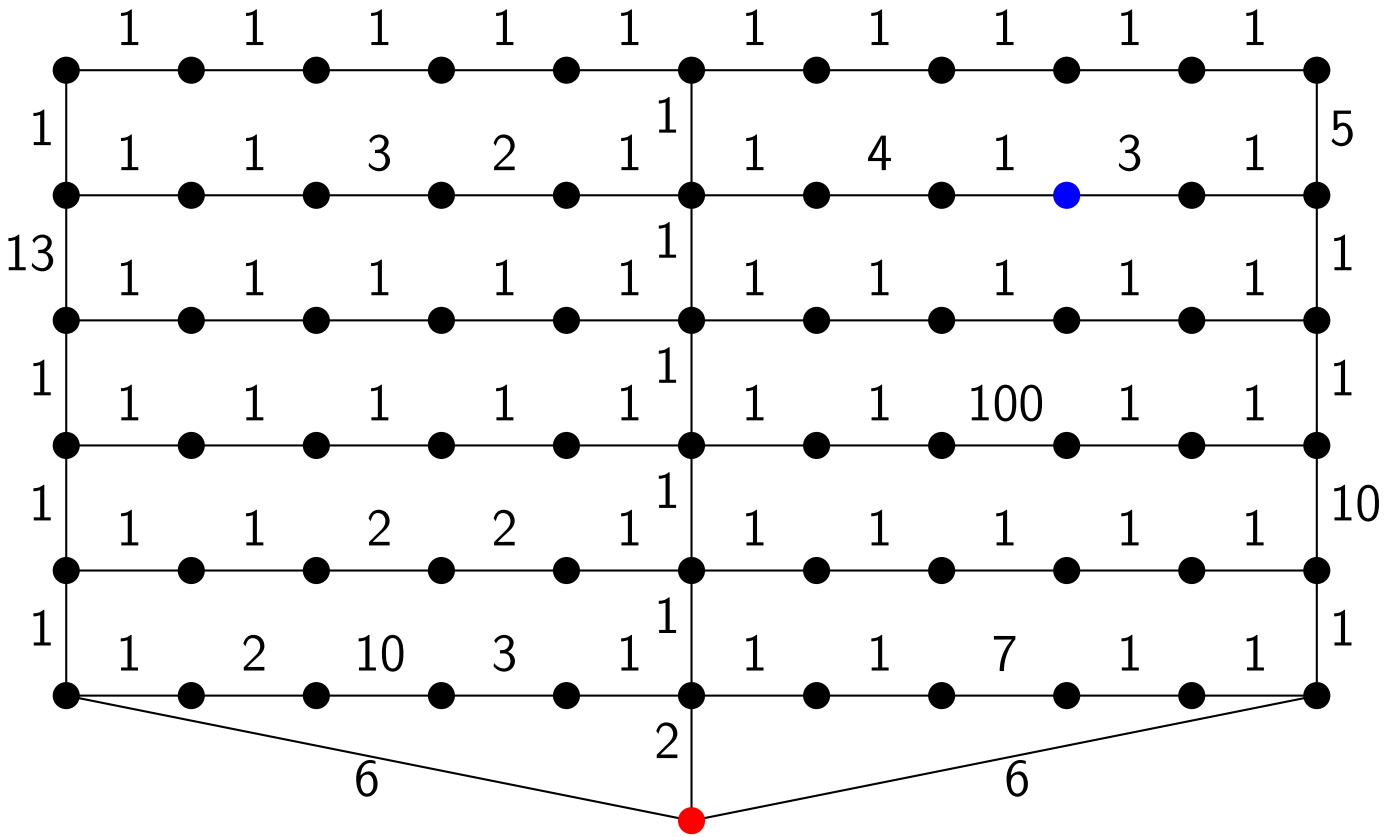
Exemplo - Caminho Mínimo - 1ª alternativa



Otimização

Exemplo - Caminho Mínimo - 2ª alternativa

- Preprocessar o problema e construir um grafo que só leva em consideração a melhor alternativa entre pontos adjacentes.



Exemplo - Caminho Mínimo - Estocástico

- E se no meio do caminho acontece um imprevisto (alguém levanta engarrafando o corredor)?
- Se o grafo for dinâmico, podemos querer
  - maximizar a probabilidade de fazermos o melhor caminho
  - minimizar a chance de fazermos o pior caminho
  - minimizar o pior caso
  - ...
- Mas para isso teríamos que fazer um estudo probabilístico.

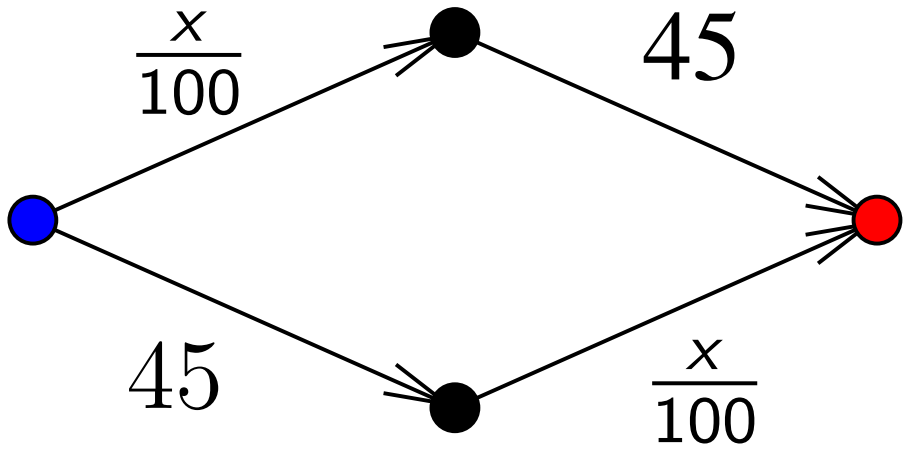
Otimização

Exemplo - Caminho Mínimo - Teoria dos jogos

- E se mais de uma pessoa está caminhando no grafo? Uma pode atrapalhar a outra.
- Alguns sites de mapas já levam em consideração o número de buscas de rotas.
- A melhor solução pode levar a um sistema pior. Temos que tomar cuidado com as decisões tomadas.

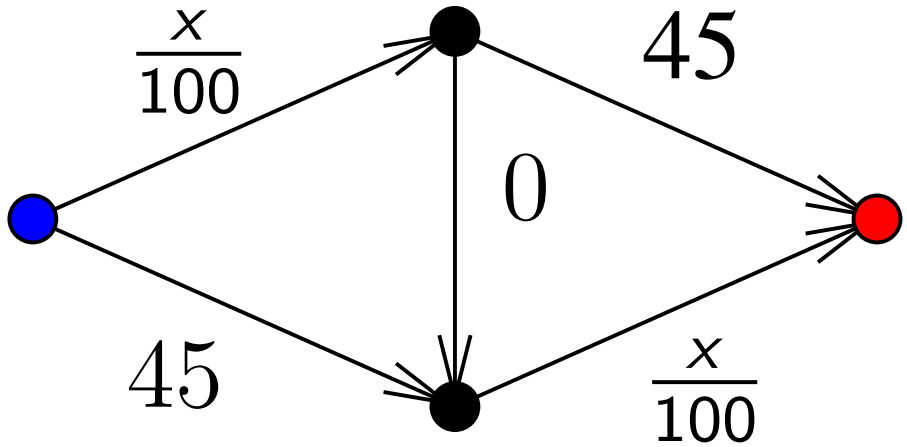
Exemplo - Caminho Mínimo - Teoria dos jogos

- Veículos - $x = 4000$
- Melhor opção metade em cada rota, tempo de cada rota 65 min. (Equilíbrio de Nash)



Exemplo - Caminho Mínimo - Teoria dos jogos

- O governo decide melhorar a rede e cria uma via super rápida.
- Agora a melhor opção é todos usaram essa nova via levando a um tempo de 80 min.
- Piorando a solução do sistema!!!



Modelo Matemático

Exemplo muito simples

- Dado um vetor $v = [v_1, v_2, \ldots, v_n]$.
- Queremos o elemento do vetor com o maior valor.
- $\max_{i=1,\ldots,n} (v_i)$
- $\max \sum_{i=1}^{n} v_i y_i$, onde $\sum_{i=1}^{n} y_i = 1$ e $y \geq 0$.
- $\min z$, onde $z \geq v_i$.

Modelo Matemático

Exemplo muito simples

- Dado um vetor $v = [v_1, v_2, \ldots, v_n]$.
- Queremos o elemento do vetor com o maior valor.
- $\max_{i=1,\ldots,n} (v_i)$
- $\max \sum_{i=1}^{n} v_i y_i$, onde $\sum_{i=1}^{n} y_i = 1$ e $y \geq 0$.
- $\min z$, onde $z \geq v_i$.

Modelo Matemático

Exemplo muito simples

- Dado um vetor $v = [v_1, v_2, \ldots, v_n]$.
- Queremos o elemento do vetor com o maior valor.
- $\max_{i=1,\ldots,n} (v_i)$
- $\max \sum_{i=1}^{n} v_i y_i$, onde $\sum_{i=1}^{n} y_i = 1$ e $y \geq 0$.
- $\min z$, onde $z \geq v_i$.

Modelo Matemático

Exemplo muito simples

- Dado um vetor $v = [v_1, v_2, \ldots, v_n]$.
- Queremos o elemento do vetor com o maior valor.
- $\max_{i=1,\ldots,n} (v_i)$
- $\max \sum_{i=1}^{n} v_i y_i$, onde $\sum_{i=1}^{n} y_i = 1$ e $y \geq 0$.
- $\min z$, onde $z \geq v_i$.

Modelo Matemático

Vantagens

- Não precisa traduzir para outras línguas.
- O problema fica bem definido. Não dá margem à dúvidas.
- Podemos ganhar um algoritmo de solução **caixa preta**.
- Normalmente ajuda na solução do problema.
- ...

Otimização

Problema real x Problema que resolvemos

- Normalmente é muito difícil descrever o mundo real.
- Quando estamos planejando as decisões não levamos em consideração o jeitinho do dia a dia.
- Além disso, o mundo real tem imprevistos.
- Podemos até usar um modelo estocástico, mas não temos como prever todos os cenários.

Otimização

Problema real x Problema que resolvemos

- Acabamos achando a solução ótima de um problema simplificado. Dependendo das escolhas e do problema essa solução pode ser muito diferente da solução ótima real.
- Esse é um problema na relação da academia com a industria.
- Quem sabe resolver o problema simplificado normalmente não entende do problema real e vice-versa.
- Saber o que é relevante ou não é bastante difícil. Essas escolhas podem afetar a resolução e/ou na qualidade da solução.
- Normalmente, é criado uma ferramente que ajuda na tomada de decisão.

Conclusão

- Praticamente tudo pode ser formulado com otimização.
- Ex: P = NP? $\rightarrow \max |P|$ ou $\min |NP| - |P|$
- Essa conclusão não ajuda a entender o que é Otimização!
- A interseção é grande com outras áreas.
- Qual é a diferença?
  - Não é algoritmo exatos.
  - Não é modelagem matemática.
  - Não é dificuldade dos problemas
  - Acho que é o compromisso com a função objetivo.

Principais Técnicas

- Solução Exata
  - Programação Matemática
    - Programação Linear Inteira Mista
    - Programação Não Linear
  - Programação Estocástica
  - Programação Dinâmica
  - Programação por Restrição
- Solução Não Exatas
  - Heurísticas
    - Gulosa
    - Aleatória
    - Busca Local
    - Lagrangeana
  - Meta-heurísticas
    - Grasp
    - Busca Tabu
    - Recozimento Simulado (simulated annealing)
    - Algoritmos Genéticos (genetic algorithms)
    - Otimização Colônia de Formigas (ant colony optimization)
  - Algoritmos Aproximados

Principais Áreas de Aplicação

- Área Militar
- Indústria Petrolífera
- Indústria de Alimentos
- Indústria Siderúrgica
- Telecomunicações
- Distribuição de Energia Elétrica
- Transporte
- Mercado Financeiro
- ...

Exemplos de Problemas

- Plano de produção
  - Projetar o layout dos equipamentos numa fabrica ou componentes de um chip para reduzir o tempo de fabricação
- Otimização de redes
  - Projetar rede de telecomunicação para manter a qualidade de serviço durante falhas
- Gestão da Cadeia de Abastecimento
  - Gerenciamento do fluxo de matérias-primas e produtos baseados na demanda incerta para os produtos finais
- Roteamento
  - Determinar as rotas dos veículos na qual o menor número de veículos é usado
- Transporte
  - Gerenciamento de sistemas de transporte de cargas e entrega

Força Bruta

- Tentando resolver por enumeração
  - Normalmente o número de soluções viáveis são $n!$ ou 2^n.

n	$\log n$	n^2	2^n	$n!$
10	3.32	10^2	$1.02x10^3$	$3.6x10^6$
100	6.64	10^4	$1.27x10^{30}$	$9.33x10^{157}$
1000	9.97	10^6	$1.07x10^{301}$	$4.02x10^{2567}$

  - Num supercomputador atual (10^{16} flops) seria necessário $2.95x10^{134}$ anos (100!) e $1.27x10^{2544}$ anos (1000!).
    Obs: o universo tem $13.82x10^9$ anos e o número de átomos no Universo é estimado entre $4x10^{78}$ e $6x10^{79}$
  - Usando enumeração completa só revolvemos problemas para n muito pequeno!!

Exemplo: Problema da Mochila 0-1

- Versão 1:
  - Um ladrão tem n possíveis itens para roubar
  - Cada item i tem um valor c_i e peso a_i
  - O ladrão tem um limite máximo de peso que consegue levar b
  - Objetivo: selecionar os itens com maior valor
- Versão 2:
  - Uma empresa tem n possíveis projetos para investir
  - Cada projeto tem um custo a_i e um retorno c_i
  - O orçamento da empresa para investir é limitado b
  - Objetivo: selecionar um portfólio de projeto de máximo retorno
- Um problema pode ser "adaptado" para outros problemas "reais"

Exemplo: Problema da Mochila 0-1

- Uma empresa tem n possíveis projetos para investir
- Cada projeto tem um custo a_i e um retorno c_i
- O orçamento da empresa para investir é limitado b
- Objetivo: selecionar um portfólio de projeto de máximo retorno
- Formulação Matemática

$$\max \left\{ \sum_{i=1}^{n} c_i x_i : x \in P \cap \mathbb{B}^n \right\}$$

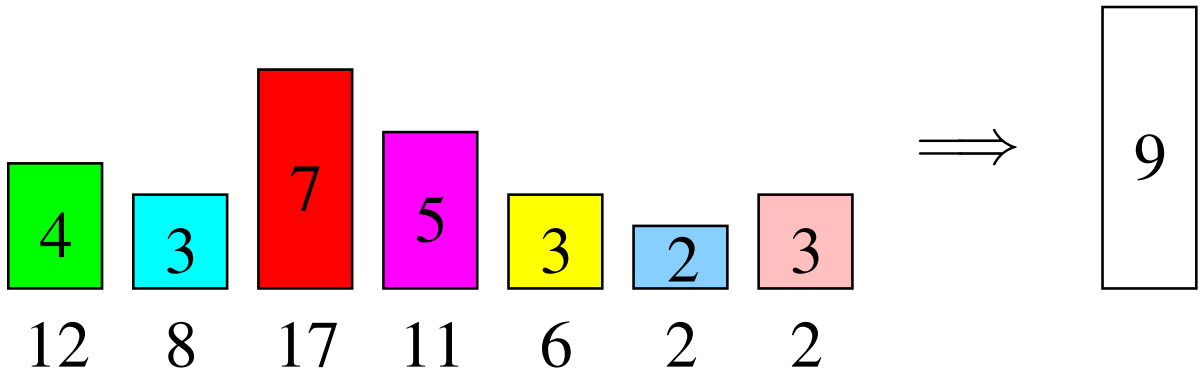
Onde P é dado por:

$$\sum_{i=1}^{n} a_i x_i \leq b$$

$$0 \leq x_i \leq 1, \ \forall i \in \{1, \ldots, n\}$$

Exemplo: Problema da Mochila 0-1

- Uma empresa tem n possíveis projetos para investir
- Cada projeto tem um custo a_i e um retorno c_i
- O orçamento da empresa para investir é limitado b
- Objetivo: selecionar um portfólio de projeto de máximo retorno
- Exemplo de Heurística Gulosa

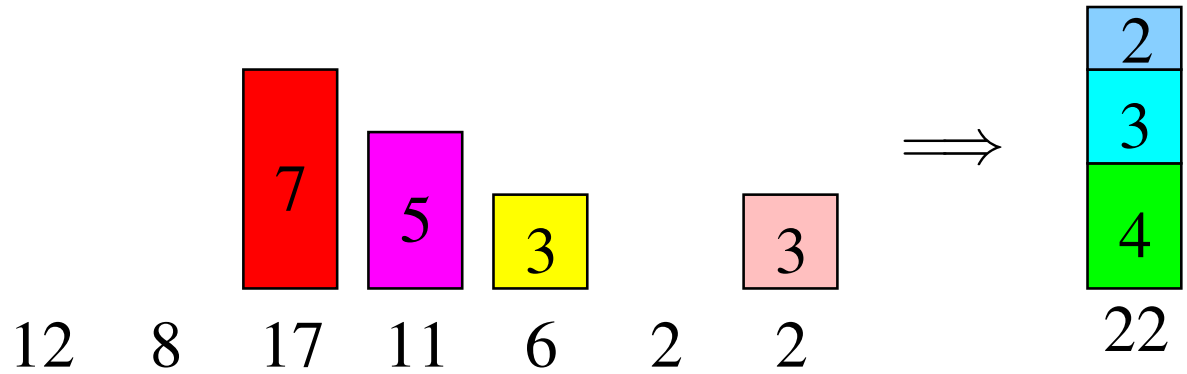


1. ordenar os n elementos em ordem decrescente da razão $\frac{c_j}{a_j}$
2. adicionar os elementos nessa ordem até o limite ser estourado

- Solução heurística - custo 22
- Solução ótima - custo 23

Exemplo: Problema da Mochila 0-1

- Uma empresa tem n possíveis projetos para investir
- Cada projeto tem um custo a_i e um retorno c_i
- O orçamento da empresa para investir é limitado b
- Objetivo: selecionar um portfólio de projeto de máximo retorno
- Exemplo de Heurística Gulosa

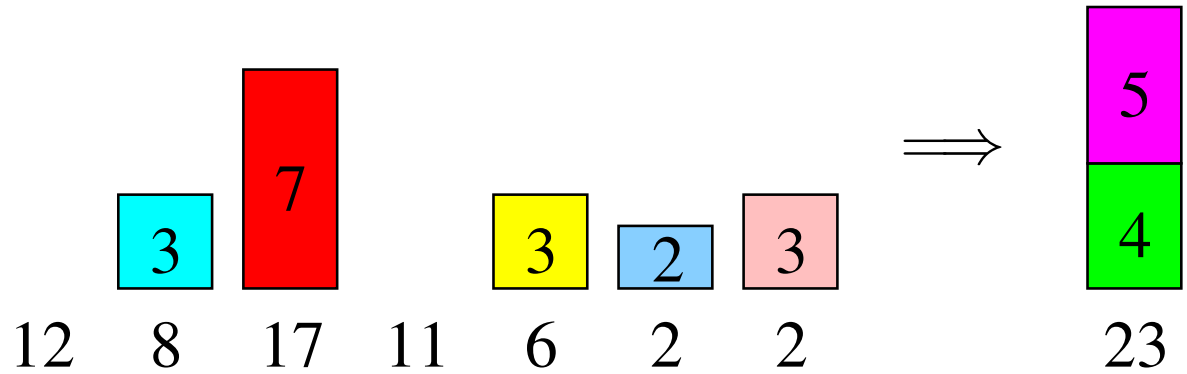


1. ordenar os n elementos em ordem decrescente da razão $\frac{c_j}{a_j}$
2. adicionar os elementos nessa ordem até o limite ser estourado

- Solução heurística - custo 22
- Solução ótima - custo 23

Exemplo: Problema da Mochila 0-1

- Uma empresa tem n possíveis projetos para investir
- Cada projeto tem um custo a_i e um retorno c_i
- O orçamento da empresa para investir é limitado b
- Objetivo: selecionar um portfólio de projeto de máximo retorno
- Exemplo de Heurística Gulosa



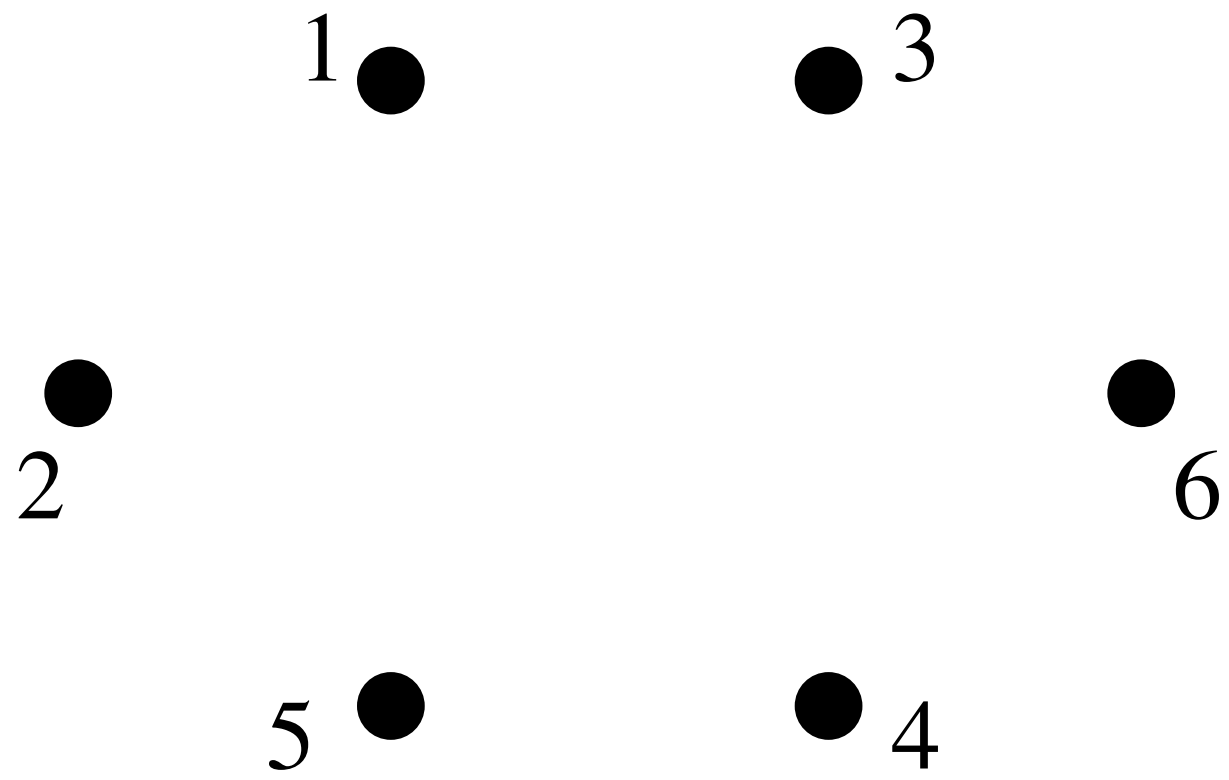
1. ordenar os n elementos em ordem decrescente da razão $\frac{c_j}{a_j}$
2. adicionar os elementos nessa ordem até o limite ser estourado

- Solução heurística - custo 22
- Solução ótima - custo 23

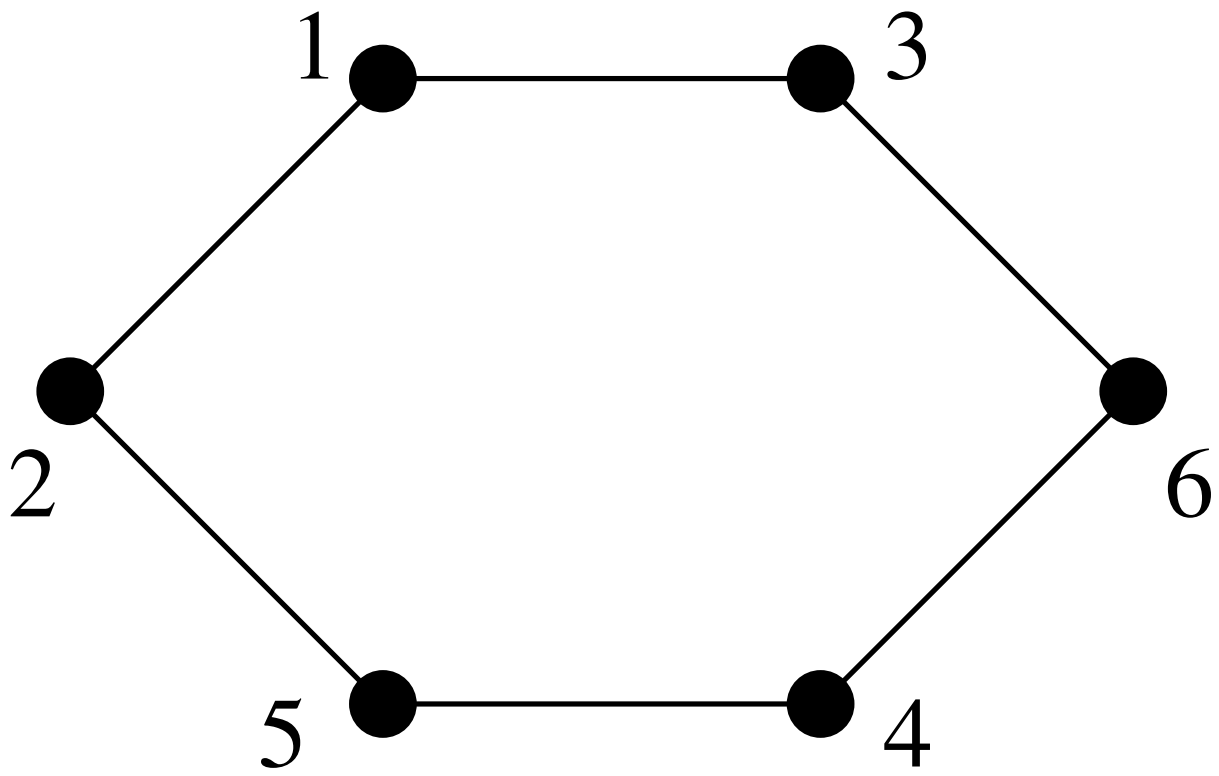
Exemplo: Problema do Caixeiro Viajante (Traveling Salesman Problem, TSP)

- É o mais famoso problema em Otimização Inteira e Combinatória!
- Um conjunto de cidades $N = \{1, \ldots, n\}$ é dado.
- O tempo de viagem entre quaisquer duas cidades é conhecido.
- Um caixeiro viajante deve visitar cada cidade uma única vez e retornar ao ponto de partida.
- Objetivo: Definir uma ordem de visita de forma a retornar ao ponto de partida o mais rápido possível.
- uma solução viável do TSP é chamado de um tour e, no grafo, corresponde a um ciclo hamiltoniano.

Exemplo: Problema do Caixeiro Viajante (Traveling Salesman Problem, TSP)



Exemplo: Problema do Caixeiro Viajante (Traveling Salesman Problem, TSP)

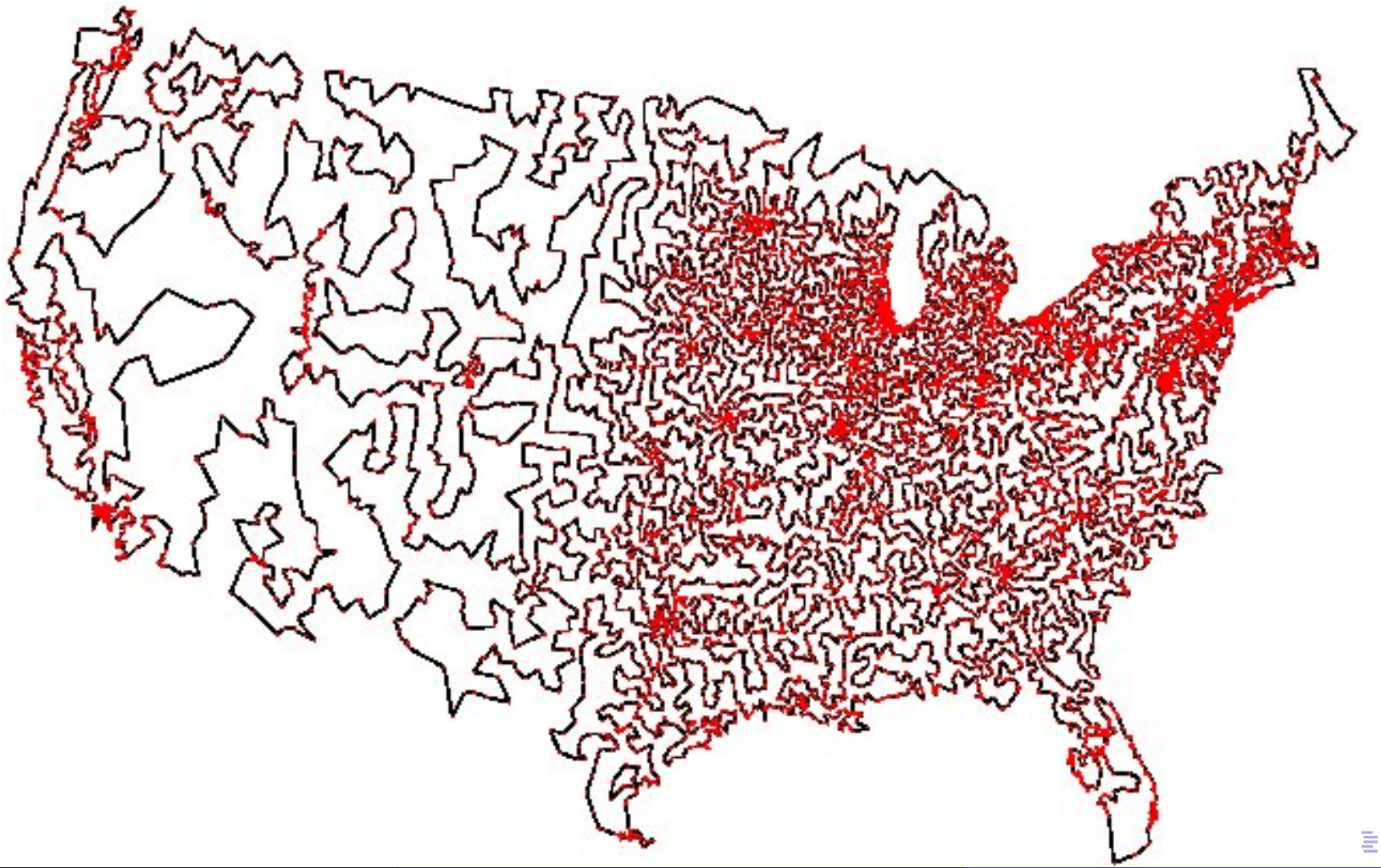


Exemplo: Problema do Caixeiro Viajante (Traveling Salesman Problem, TSP)

USA - 13509 cidades

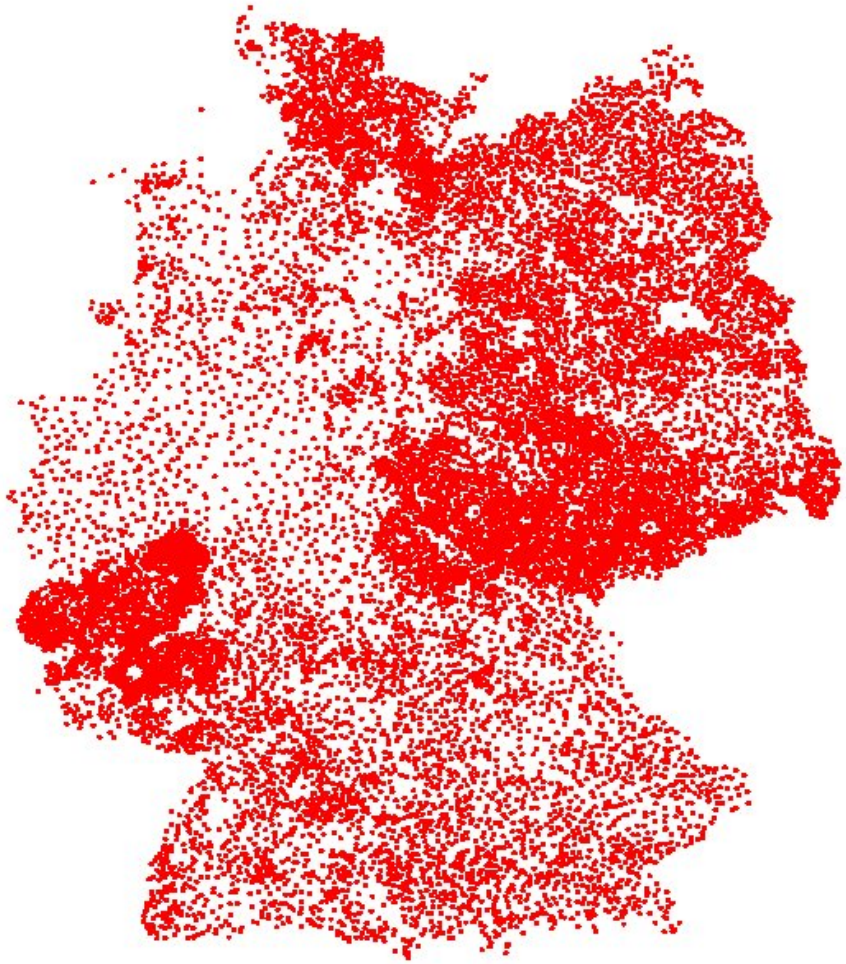
Exemplo: Problema do Caixeiro Viajante (Traveling Salesman Problem, TSP)

USA - 13509 cidades



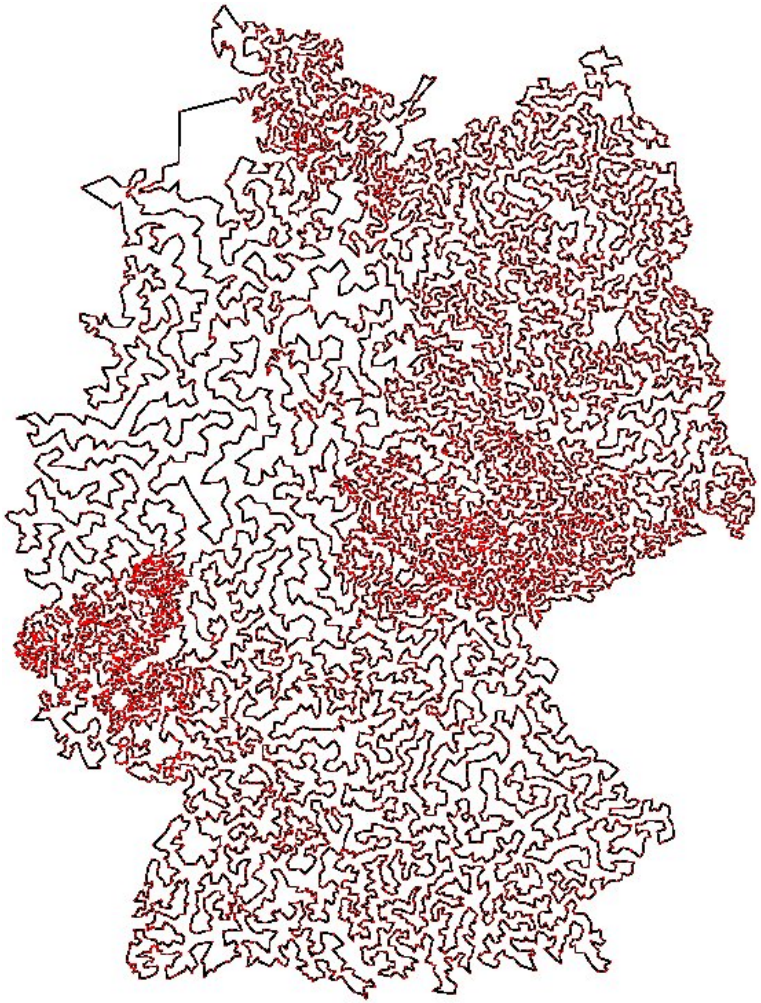
Exemplo: Problema do Caixeiro Viajante (Traveling Salesman Problem, TSP)

Alemanha - 15112 cidades



Exemplo: Problema do Caixeiro Viajante (Traveling Salesman Problem, TSP)

Alemanha - 15112 cidades

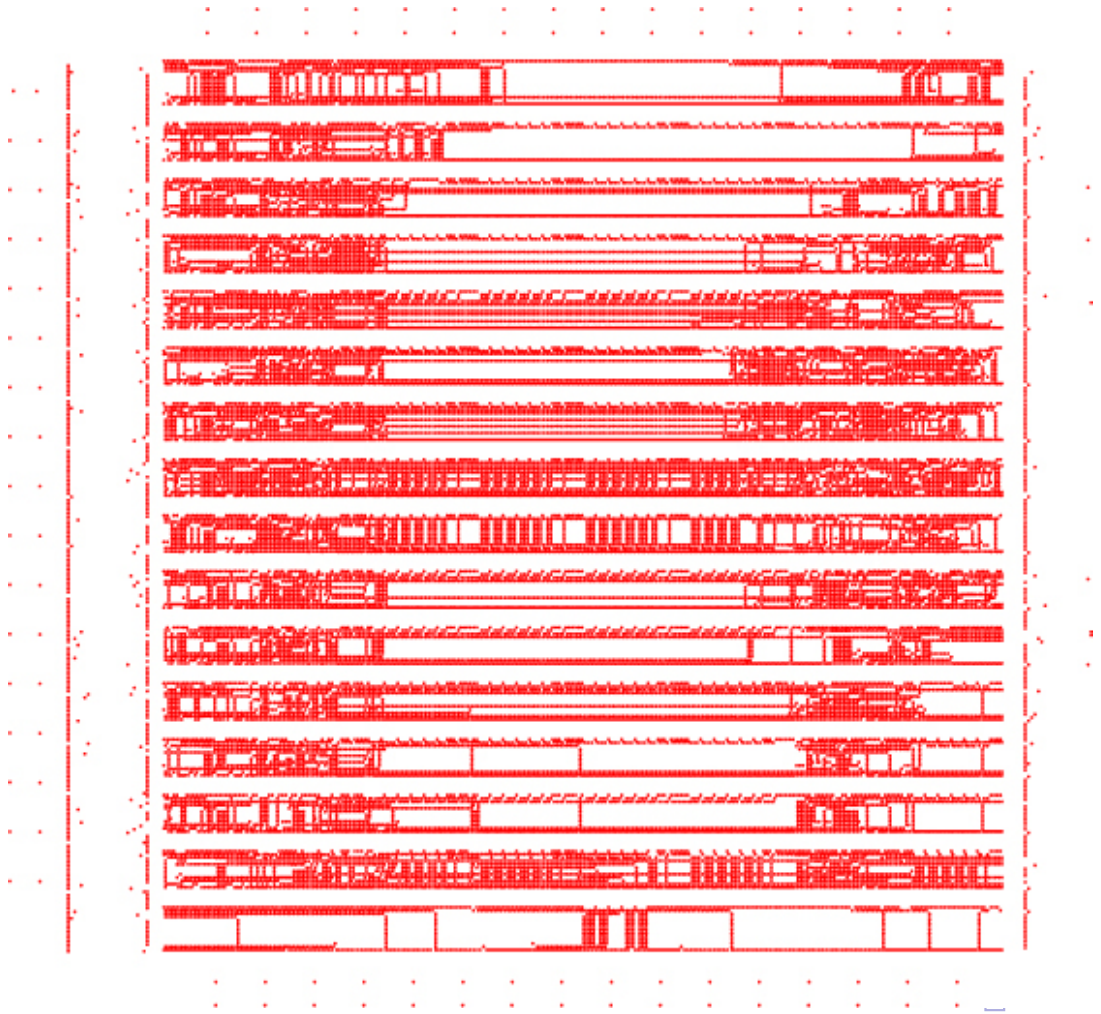


Exemplo: Problema do Caixeiro Viajante (Traveling Salesman Problem, TSP)

- Por que estudar um problema que não tem mais tanta aplicação prática?
- Um método desenvolvido para um problema "normalmente" pode ser adaptado para outros
- Na verdade o TSP tem diversas aplicações atuais
  - A principal aplicação vem de subproblemas em várias aplicações de transporte e logística (Ônibus escolar, equipamento agrícola para teste de solo, visita agendada em firma de cabo, delivery de refeição para pessoas "restritas" em casa, planejamento de retroescavadeira em armazéns, etc)
  - planejamento de uma máquina de furar uma placa de circuito
  - Otimizar a sequencia de imagens de objetos celestes
  - Coleta de moedas em telefones públicos
  - ...

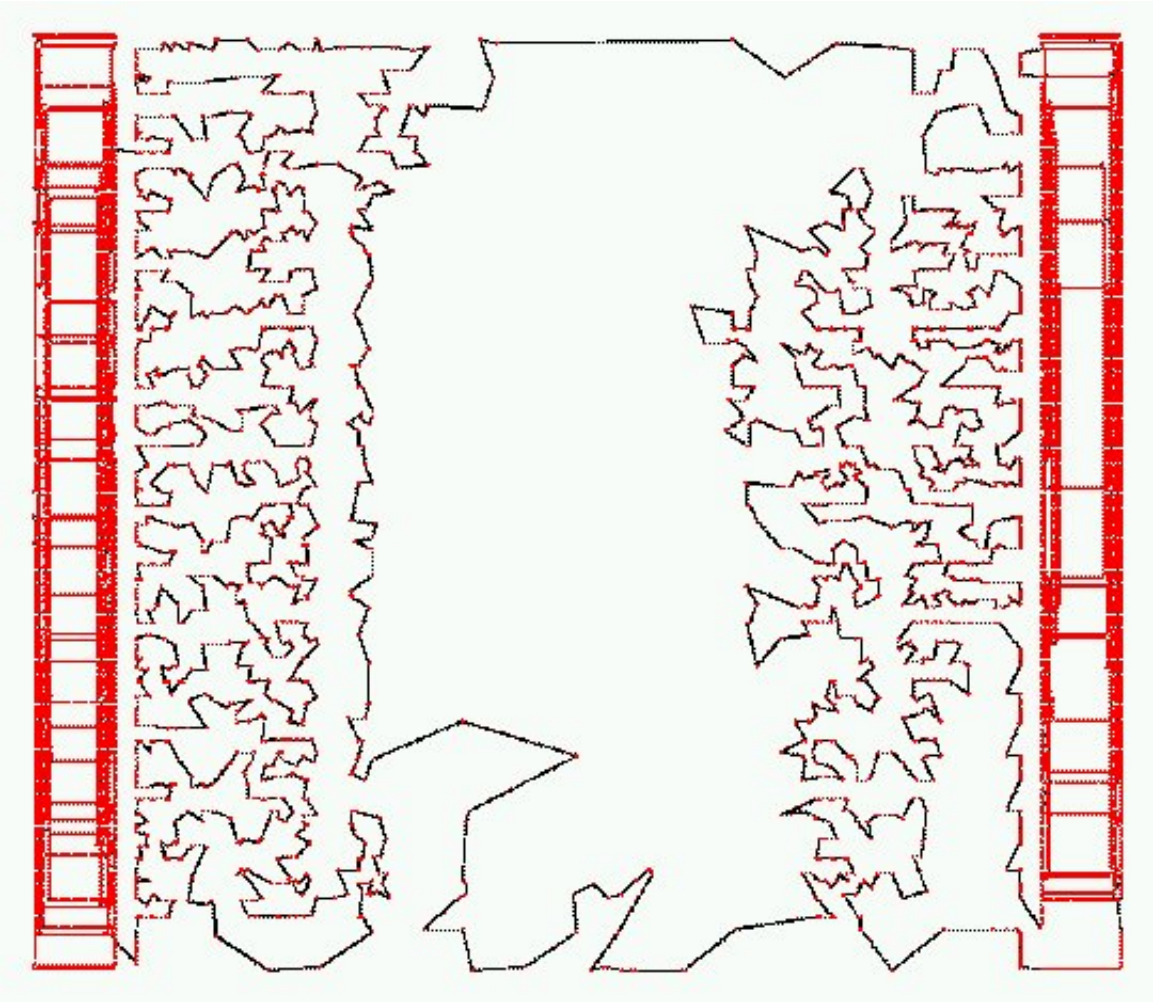
Exemplo: Problema do Caixeiro Viajante (Traveling Salesman Problem, TSP)

Placa de circuito - 7397 pontos



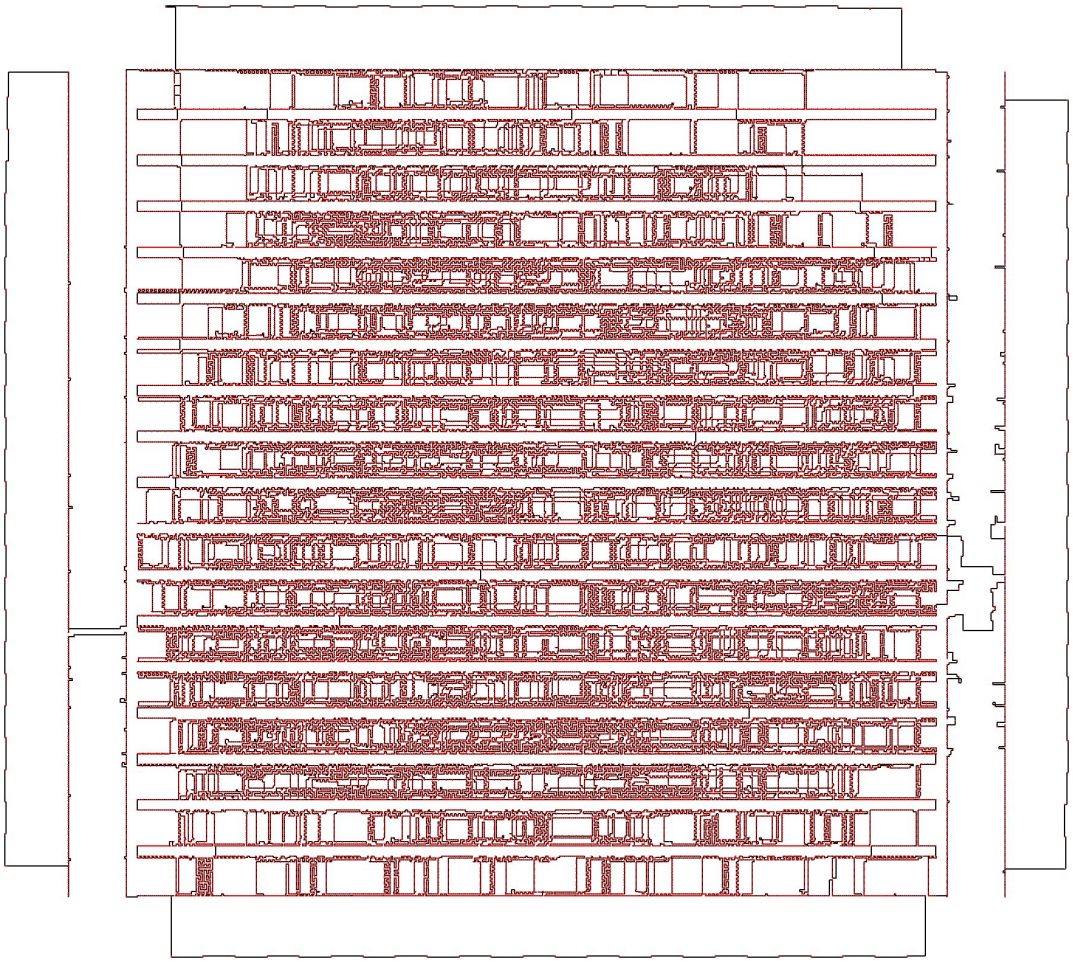
Exemplo: Problema do Caixeiro Viajante (Traveling Salesman Problem, TSP)

Placa de circuito - 7397 pontos



Exemplo: Problema do Caixeiro Viajante (Traveling Salesman Problem, TSP)

Placa de circuito - 85900 pontos



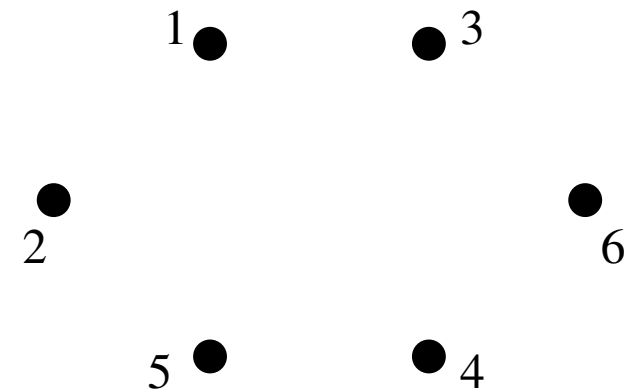
Exemplo: Problema do Caixeiro Viajante (Traveling Salesman Problem, TSP)

Limite Primal - Exemplo - TSP

$$c = \begin{bmatrix} - & 9 & 2 & 8 & 12 & 11 \\ & - & 7 & 19 & 10 & 32 \\ & & - & 29 & 18 & 6 \\ & & & - & 24 & 3 \\ & & & & - & 19 \\ & & & & & - \end{bmatrix}$$

- Algoritmo guloso
  1. ordenar as arestas em ordem crescente de custos
  2. adicionar as arestas nessa ordem até ter adicionado n arestas
     - o grau dos vértices não pode ser maior que 2
     - não pode formar subciclo

Lista ordenada das arestas:



Custo:

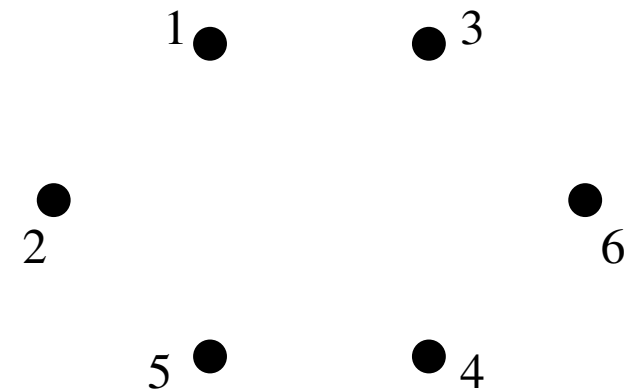
Exemplo: Problema do Caixeiro Viajante (Traveling Salesman Problem, TSP)

Limite Primal - Exemplo - TSP

$$c = \begin{bmatrix} - & 9 & 2 & 8 & 12 & 11 \\ & - & 7 & 19 & 10 & 32 \\ & & - & 29 & 18 & 6 \\ & & & - & 24 & 3 \\ & & & & - & 19 \\ & & & & & - \end{bmatrix}$$

- Algoritmo guloso
  1. ordenar as arestas em ordem crescente de custos
  2. adicionar as arestas nessa ordem até ter adicionado n arestas
     - o grau dos vértices não pode ser maior que 2
     - não pode formar subciclo

Lista ordenada das arestas:
$\{(1,3),(4,6),(3,6),(2,3),(1,4),$
$(1,2),(2,5),(1,6),(1,5),(3,5),$
$(2,4),(5,6),(4,5),(3,4),(2,6)\}$



Custo:

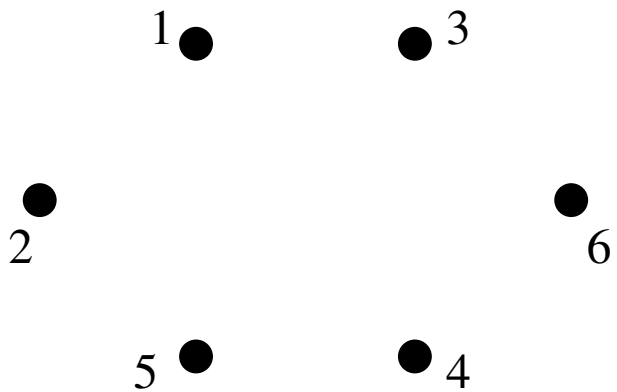
Exemplo: Problema do Caixeiro Viajante (Traveling Salesman Problem, TSP)

Limite Primal - Exemplo - TSP

$$c = \begin{bmatrix} - & 9 & 2 & 8 & 12 & 11 \\ & - & 7 & 19 & 10 & 32 \\ & & - & 29 & 18 & 6 \\ & & & - & 24 & 3 \\ & & & & - & 19 \\ & & & & & - \end{bmatrix}$$

- Algoritmo guloso
  1. ordenar as arestas em ordem crescente de custos
  2. adicionar as arestas nessa ordem até ter adicionado n arestas
     - o grau dos vértices não pode ser maior que 2
     - não pode formar subciclo

Lista ordenada das arestas:
$\{(1,3),(4,6),(3,6),(2,3),(1,4),$
$(1,2),(2,5),(1,6),(1,5),(3,5),$
$(2,4),(5,6),(4,5),(3,4),(2,6)\}$



Custo:

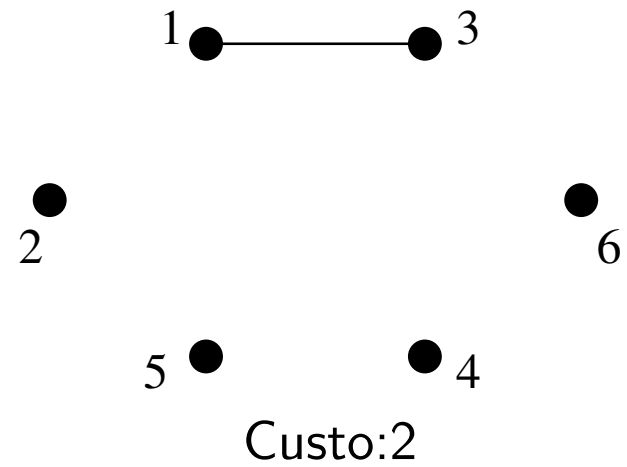
Exemplo: Problema do Caixeiro Viajante (Traveling Salesman Problem, TSP)

Limite Primal - Exemplo - TSP

$$c = \begin{bmatrix} - & 9 & 2 & 8 & 12 & 11 \\ & - & 7 & 19 & 10 & 32 \\ & & - & 29 & 18 & 6 \\ & & & - & 24 & 3 \\ & & & & - & 19 \\ & & & & & - \end{bmatrix}$$

- Algoritmo guloso
  1. ordenar as arestas em ordem crescente de custos
  2. adicionar as arestas nessa ordem até ter adicionado n arestas
     - o grau dos vértices não pode ser maior que 2
     - não pode formar subciclo

Lista ordenada das arestas:
$\{(1,3),(4,6),(3,6),(2,3),(1,4),$
$(1,2),(2,5),(1,6),(1,5),(3,5),$
$(2,4),(5,6),(4,5),(3,4),(2,6)\}$



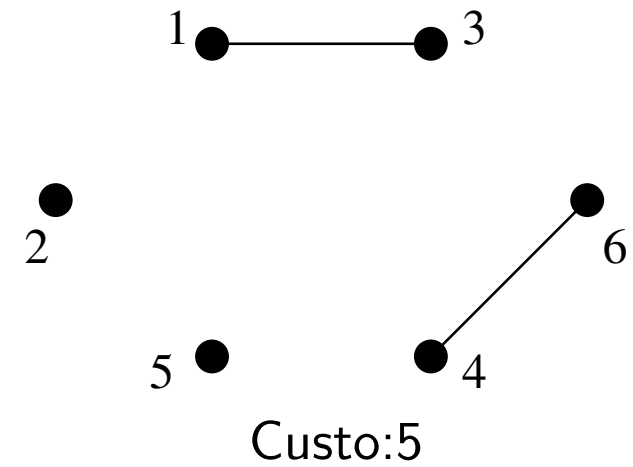
Custo:2

Exemplo: Problema do Caixeiro Viajante (Traveling Salesman Problem, TSP)

Limite Primal - Exemplo - TSP

$$c = \begin{bmatrix} - & 9 & 2 & 8 & 12 & 11 \\ & - & 7 & 19 & 10 & 32 \\ & & - & 29 & 18 & 6 \\ & & & - & 24 & 3 \\ & & & & - & 19 \\ & & & & & - \end{bmatrix}$$

Lista ordenada das arestas:
$\{(1,3),(4,6),(3,6),(2,3),(1,4),$
$(1,2),(2,5),(1,6),(1,5),(3,5),$
$(2,4),(5,6),(4,5),(3,4),(2,6)\}$



Custo:5

- Algoritmo guloso
  1. ordenar as arestas em ordem crescente de custos
  2. adicionar as arestas nessa ordem até ter adicionado n arestas
     - o grau dos vértices não pode ser maior que 2
     - não pode formar subciclo

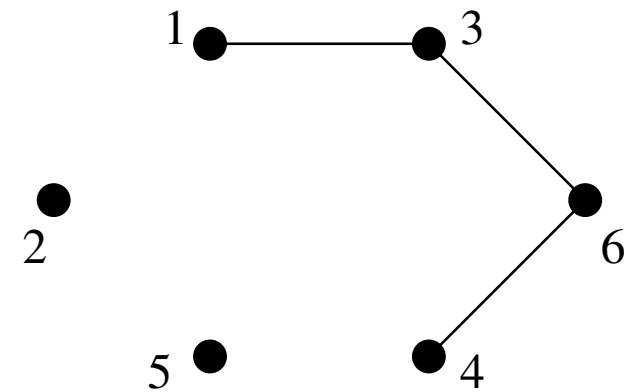
Exemplo: Problema do Caixeiro Viajante (Traveling Salesman Problem, TSP)

Limite Primal - Exemplo - TSP

$$c = \begin{bmatrix} - & 9 & 2 & 8 & 12 & 11 \\ & - & 7 & 19 & 10 & 32 \\ & & - & 29 & 18 & 6 \\ & & & - & 24 & 3 \\ & & & & - & 19 \\ & & & & & - \end{bmatrix}$$

- Algoritmo guloso
  1. ordenar as arestas em ordem crescente de custos
  2. adicionar as arestas nessa ordem até ter adicionado n arestas
     - o grau dos vértices não pode ser maior que 2
     - não pode formar subciclo

Lista ordenada das arestas:
$\{(1,3),(4,6),(3,6),(2,3),(1,4),$
$(1,2),(2,5),(1,6),(1,5),(3,5),$
$(2,4),(5,6),(4,5),(3,4),(2,6)\}$



Custo:11

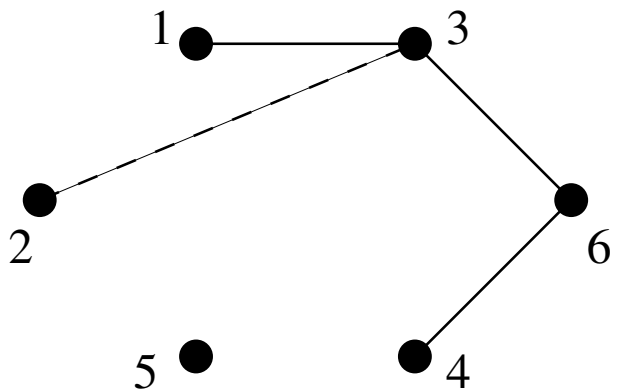
Exemplo: Problema do Caixeiro Viajante (Traveling Salesman Problem, TSP)

Limite Primal - Exemplo - TSP

$$c = \begin{bmatrix} - & 9 & 2 & 8 & 12 & 11 \\ & - & 7 & 19 & 10 & 32 \\ & & - & 29 & 18 & 6 \\ & & & - & 24 & 3 \\ & & & & - & 19 \\ & & & & & - \end{bmatrix}$$

- Algoritmo guloso
  1. ordenar as arestas em ordem crescente de custos
  2. adicionar as arestas nessa ordem até ter adicionado n arestas
     - o grau dos vértices não pode ser maior que 2
     - não pode formar subciclo

Lista ordenada das arestas:
$\{(1,3),(4,6),(3,6),(2,3),(1,4),$
$(1,2),(2,5),(1,6),(1,5),(3,5),$
$(2,4),(5,6),(4,5),(3,4),(2,6)\}$



Custo:11

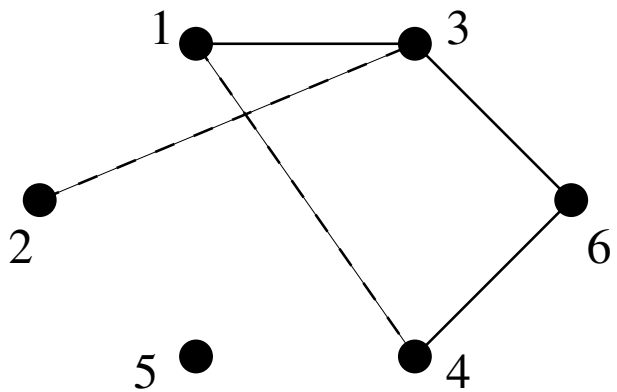
Exemplo: Problema do Caixeiro Viajante (Traveling Salesman Problem, TSP)

Limite Primal - Exemplo - TSP

$$c = \begin{bmatrix} - & 9 & 2 & 8 & 12 & 11 \\ & - & 7 & 19 & 10 & 32 \\ & & - & 29 & 18 & 6 \\ & & & - & 24 & 3 \\ & & & & - & 19 \\ & & & & & - \end{bmatrix}$$

- Algoritmo guloso
  1. ordenar as arestas em ordem crescente de custos
  2. adicionar as arestas nessa ordem até ter adicionado n arestas
     - o grau dos vértices não pode ser maior que 2
     - não pode formar subciclo

Lista ordenada das arestas:
$\{(1,3),(4,6),(3,6),(2,3),(1,4),$
$(1,2),(2,5),(1,6),(1,5),(3,5),$
$(2,4),(5,6),(4,5),(3,4),(2,6)\}$



Custo:11

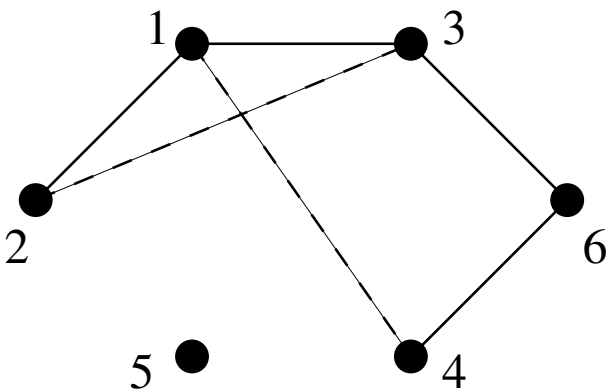
Exemplo: Problema do Caixeiro Viajante (Traveling Salesman Problem, TSP)

Limite Primal - Exemplo - TSP

$$c = \begin{bmatrix} - & 9 & 2 & 8 & 12 & 11 \\ & - & 7 & 19 & 10 & 32 \\ & & - & 29 & 18 & 6 \\ & & & - & 24 & 3 \\ & & & & - & 19 \\ & & & & & - \end{bmatrix}$$

- Algoritmo guloso
  1. ordenar as arestas em ordem crescente de custos
  2. adicionar as arestas nessa ordem até ter adicionado n arestas
     - o grau dos vértices não pode ser maior que 2
     - não pode formar subciclo

Lista ordenada das arestas:
$\{(1,3),(4,6),(3,6),(2,3),(1,4),$
$(1,2),(2,5),(1,6),(1,5),(3,5),$
$(2,4),(5,6),(4,5),(3,4),(2,6)\}$



Custo:20

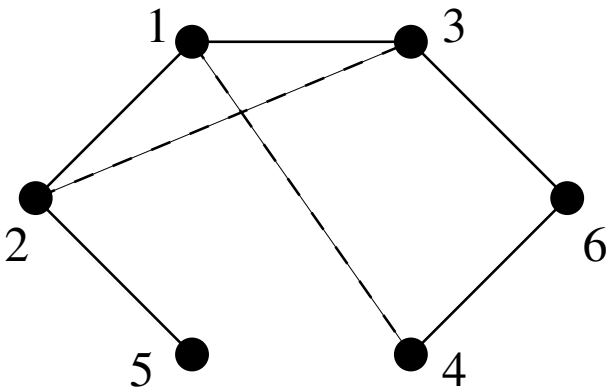
Exemplo: Problema do Caixeiro Viajante (Traveling Salesman Problem, TSP)

Limite Primal - Exemplo - TSP

$$c = \begin{bmatrix} - & 9 & 2 & 8 & 12 & 11 \\ & - & 7 & 19 & 10 & 32 \\ & & - & 29 & 18 & 6 \\ & & & - & 24 & 3 \\ & & & & - & 19 \\ & & & & & - \end{bmatrix}$$

- Algoritmo guloso
  1. ordenar as arestas em ordem crescente de custos
  2. adicionar as arestas nessa ordem até ter adicionado n arestas
     - o grau dos vértices não pode ser maior que 2
     - não pode formar subciclo

Lista ordenada das arestas:
$\{(1,3),(4,6),(3,6),(2,3),(1,4),$
$(1,2),(2,5),(1,6),(1,5),(3,5),$
$(2,4),(5,6),(4,5),(3,4),(2,6)\}$



Custo:30

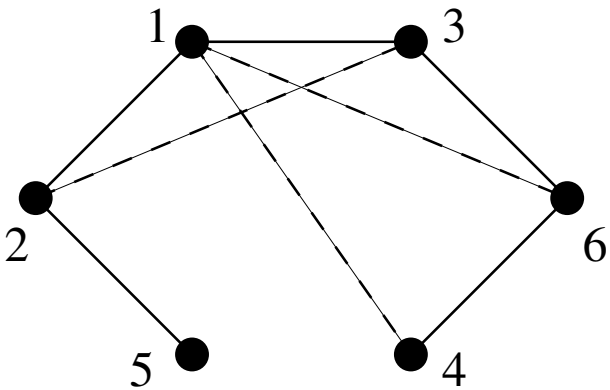
Exemplo: Problema do Caixeiro Viajante (Traveling Salesman Problem, TSP)

Limite Primal - Exemplo - TSP

$$c = \begin{bmatrix} - & 9 & 2 & 8 & 12 & 11 \\ & - & 7 & 19 & 10 & 32 \\ & & - & 29 & 18 & 6 \\ & & & - & 24 & 3 \\ & & & & - & 19 \\ & & & & & - \end{bmatrix}$$

- Algoritmo guloso
  1. ordenar as arestas em ordem crescente de custos
  2. adicionar as arestas nessa ordem até ter adicionado n arestas
     - o grau dos vértices não pode ser maior que 2
     - não pode formar subciclo

Lista ordenada das arestas:
$\{(1,3),(4,6),(3,6),(2,3),(1,4),$
$(1,2),(2,5),(1,6),(1,5),(3,5),$
$(2,4),(5,6),(4,5),(3,4),(2,6)\}$



Custo:30

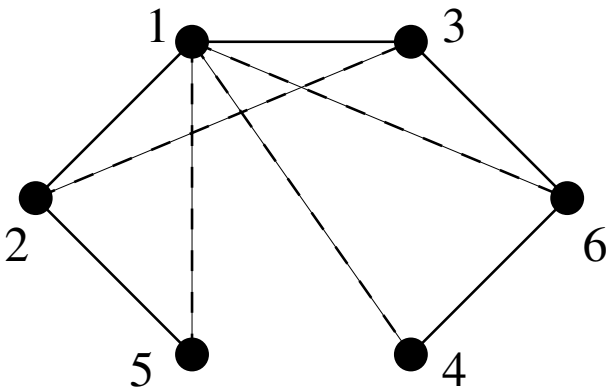
Exemplo: Problema do Caixeiro Viajante (Traveling Salesman Problem, TSP)

Limite Primal - Exemplo - TSP

$$c = \begin{bmatrix} - & 9 & 2 & 8 & 12 & 11 \\ & - & 7 & 19 & 10 & 32 \\ & & - & 29 & 18 & 6 \\ & & & - & 24 & 3 \\ & & & & - & 19 \\ & & & & & - \end{bmatrix}$$

- Algoritmo guloso
  1. ordenar as arestas em ordem crescente de custos
  2. adicionar as arestas nessa ordem até ter adicionado n arestas
     - o grau dos vértices não pode ser maior que 2
     - não pode formar subciclo

Lista ordenada das arestas:
$\{(1,3),(4,6),(3,6),(2,3),(1,4),$
$(1,2),(2,5),(1,6),(1,5),(3,5),$
$(2,4),(5,6),(4,5),(3,4),(2,6)\}$



Custo:30

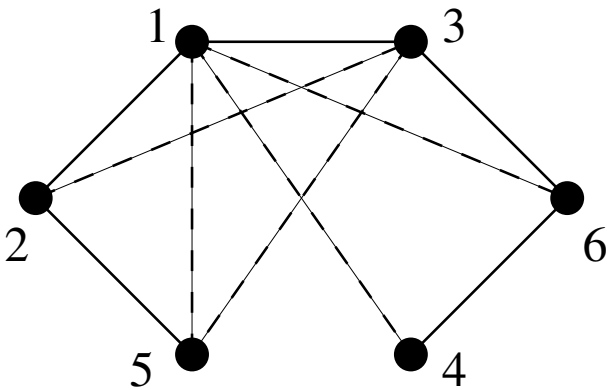
Exemplo: Problema do Caixeiro Viajante (Traveling Salesman Problem, TSP)

Limite Primal - Exemplo - TSP

$$c = \begin{bmatrix} - & 9 & 2 & 8 & 12 & 11 \\ & - & 7 & 19 & 10 & 32 \\ & & - & 29 & 18 & 6 \\ & & & - & 24 & 3 \\ & & & & - & 19 \\ & & & & & - \end{bmatrix}$$

- Algoritmo guloso
  1. ordenar as arestas em ordem crescente de custos
  2. adicionar as arestas nessa ordem até ter adicionado n arestas
     - o grau dos vértices não pode ser maior que 2
     - não pode formar subciclo

Lista ordenada das arestas:
$\{(1,3),(4,6),(3,6),(2,3),(1,4),$
$(1,2),(2,5),(1,6),(1,5),(3,5),$
$(2,4),(5,6),(4,5),(3,4),(2,6)\}$



Custo:30

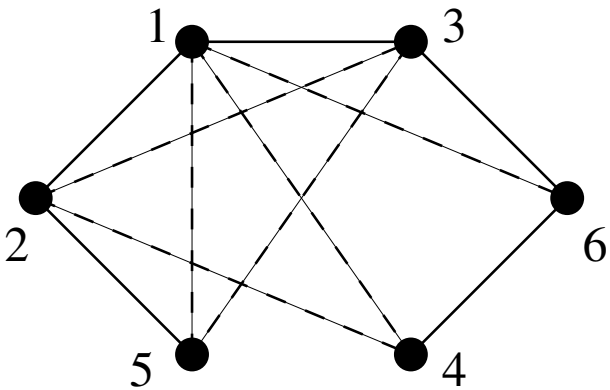
Exemplo: Problema do Caixeiro Viajante (Traveling Salesman Problem, TSP)

Limite Primal - Exemplo - TSP

$$c = \begin{bmatrix} - & 9 & 2 & 8 & 12 & 11 \\ & - & 7 & 19 & 10 & 32 \\ & & - & 29 & 18 & 6 \\ & & & - & 24 & 3 \\ & & & & - & 19 \\ & & & & & - \end{bmatrix}$$

- Algoritmo guloso
  1. ordenar as arestas em ordem crescente de custos
  2. adicionar as arestas nessa ordem até ter adicionado n arestas
     - o grau dos vértices não pode ser maior que 2
     - não pode formar subciclo

Lista ordenada das arestas:
$\{(1,3),(4,6),(3,6),(2,3),(1,4),$
$(1,2),(2,5),(1,6),(1,5),(3,5),$
$(2,4),(5,6),(4,5),(3,4),(2,6)\}$



Custo:30

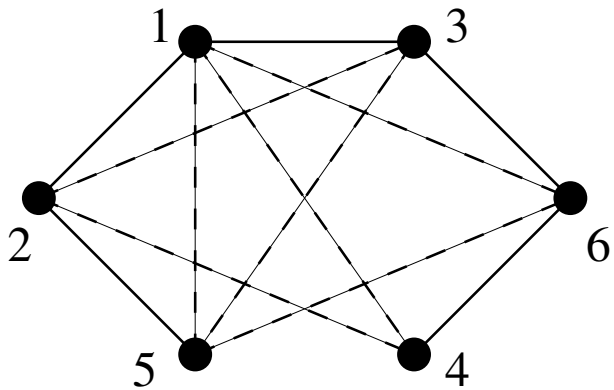
Exemplo: Problema do Caixeiro Viajante (Traveling Salesman Problem, TSP)

Limite Primal - Exemplo - TSP

$$c = \begin{bmatrix} - & 9 & 2 & 8 & 12 & 11 \\ & - & 7 & 19 & 10 & 32 \\ & & - & 29 & 18 & 6 \\ & & & - & 24 & 3 \\ & & & & - & 19 \\ & & & & & - \end{bmatrix}$$

- Algoritmo guloso
  1. ordenar as arestas em ordem crescente de custos
  2. adicionar as arestas nessa ordem até ter adicionado n arestas
     - o grau dos vértices não pode ser maior que 2
     - não pode formar subciclo

Lista ordenada das arestas:
$\{(1,3),(4,6),(3,6),(2,3),(1,4),$
$(1,2),(2,5),(1,6),(1,5),(3,5),$
$(2,4),(5,6),(4,5),(3,4),(2,6)\}$



Custo:30

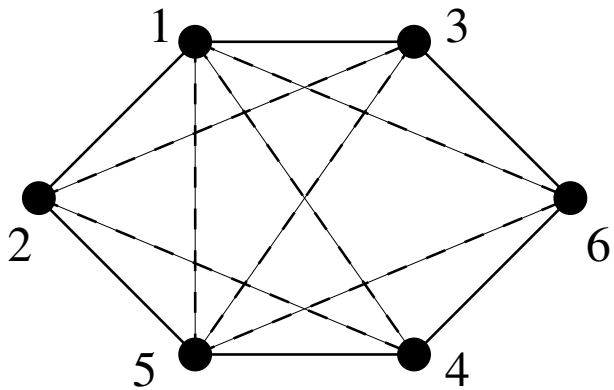
Exemplo: Problema do Caixeiro Viajante (Traveling Salesman Problem, TSP)

Limite Primal - Exemplo - TSP

$$c = \begin{bmatrix} - & 9 & 2 & 8 & 12 & 11 \\ & - & 7 & 19 & 10 & 32 \\ & & - & 29 & 18 & 6 \\ & & & - & 24 & 3 \\ & & & & - & 19 \\ & & & & & - \end{bmatrix}$$

- Algoritmo guloso
  1. ordenar as arestas em ordem crescente de custos
  2. adicionar as arestas nessa ordem até ter adicionado n arestas
     - o grau dos vértices não pode ser maior que 2
     - não pode formar subciclo

Lista ordenada das arestas:
$\{(1,3),(4,6),(3,6),(2,3),(1,4),$
$(1,2),(2,5),(1,6),(1,5),(3,5),$
$(2,4),(5,6),(4,5),(3,4),(2,6)\}$

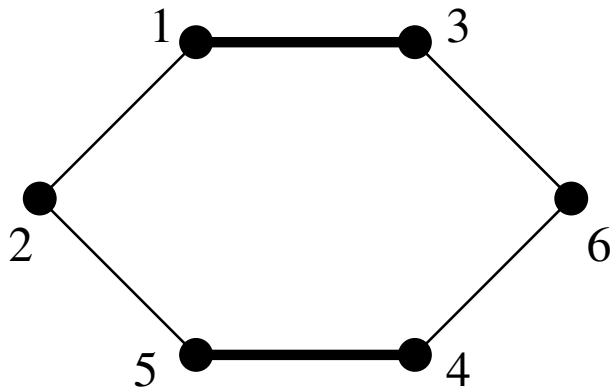


Custo:54

Exemplo: Problema do Caixeiro Viajante (Traveling Salesman Problem, TSP)

Limite Primal - Exemplo - TSP

- Algoritmo de busca local
  - Heurística da 2-troca para o TSP
  - Ciclo representado por uma permutação dos n vértices
  - Vizinhança: substituir pares de arestas (inverter sequência entre i e j)

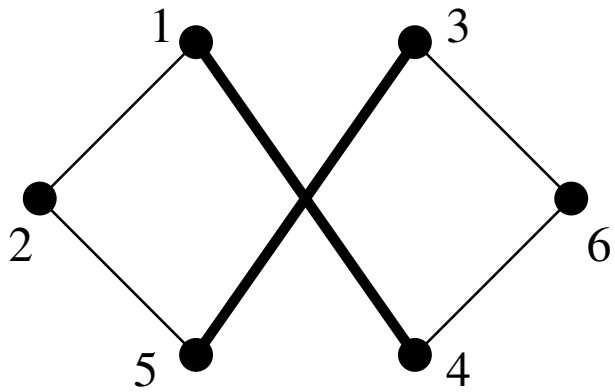


  - $(1, 3, 6, 4, 5, 2, 1)$

Exemplo: Problema do Caixeiro Viajante (Traveling Salesman Problem, TSP)

Limite Primal - Exemplo - TSP

- Algoritmo de busca local
  - Heurística da 2-troca para o TSP
  - Ciclo representado por uma permutação dos n vértices
  - Vizinhança: substituir pares de arestas (inverter sequência entre i e j)

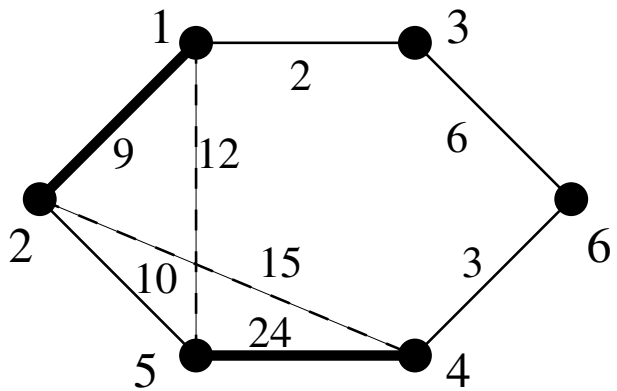


  - $(1, 4, 6, 3, 5, 2, 1)$

Exemplo: Problema do Caixeiro Viajante (Traveling Salesman Problem, TSP)

Limite Primal - Exemplo - TSP

- Algoritmo de busca local
  - Heurística da 2-troca para o TSP
  - Ciclo representado por uma permutação dos n vértices
  - Vizinhança: substituir pares de arestas (inverter sequência entre i e j)



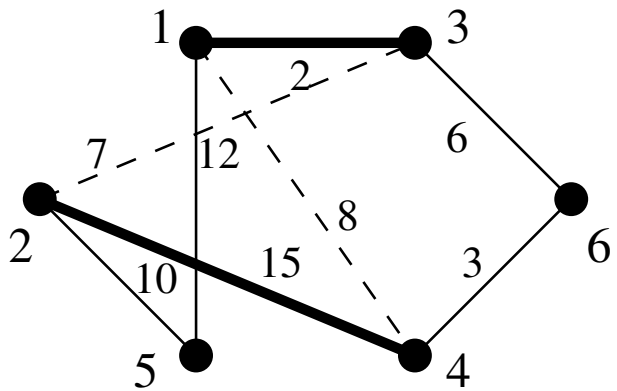
custo=54

  - $(1, 3, 6, \underline{4}, 5, 2, \underline{1}) \Rightarrow (1, 3, 6, 4, 2, 5, 1)$

Exemplo: Problema do Caixeiro Viajante (Traveling Salesman Problem, TSP)

Limite Primal - Exemplo - TSP

- Algoritmo de busca local
  - Heurística da 2-troca para o TSP
  - Ciclo representado por uma permutação dos n vértices
  - Vizinhança: substituir pares de arestas (inverter sequência entre i e j)



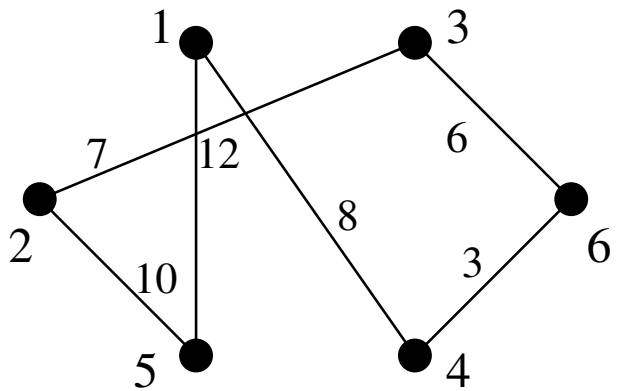
custo=48

- $(1, 3, 6, \underline{4}, 5, 2, \underline{1}) \Rightarrow (\underline{1}, 3, 6, 4, \underline{2}, 5, 1) \Rightarrow (1, 4, 6, 3, 2, 5, 1)$

Exemplo: Problema do Caixeiro Viajante (Traveling Salesman Problem, TSP)

Limite Primal - Exemplo - TSP

- Algoritmo de busca local
  - Heurística da 2-troca para o TSP
  - Ciclo representado por uma permutação dos n vértices
  - Vizinhança: substituir pares de arestas (inverter sequência entre i e j)



custo=46

- $(1, 3, 6, \underline{4}, 5, 2, \underline{1}) \Rightarrow (\underline{1}, 3, 6, 4, \underline{2}, 5, 1) \Rightarrow (1, 4, 6, 3, 2, 5, 1)$

Problema do caixeiro viajante (Traveling Salesman Problem, TSP)

Variáveis :

$$x_{ij} = \begin{cases} 1, & \text{se a cidade } i \text{ é visitada imediatamente antes da cidade } j \\ 0, & \text{caso contrário} \end{cases}$$

As variáveis x_{ii} estão indefinidas.

Função objetivo: sendo c_{ij} o tempo de viagem para ir da cidade i para a cidade j a F. Obj. é dada por

$$z = \min \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x_{ij}$$

Restrições :

- o caixeiro chega à cidade j uma única vez:

$$\sum_{i=1, i \neq j}^{n} x_{ij} = 1, \qquad \forall j \in \{1, \ldots, n\}$$

- o caixeiro sai da cidade i uma única vez:

$$\sum_{j=1, i \neq j}^{n} x_{ij} = 1, \qquad \forall i \in \{1, \ldots, n\}$$

Problema do caixeiro viajante (Traveling Salesman Problem, TSP)

Variáveis :

$$x_{ij} = \begin{cases} 1, & \text{se a cidade } i \text{ é visitada imediatamente antes da cidade } j \\ 0, & \text{caso contrário} \end{cases}$$

As variáveis x_{ii} estão indefinidas.

Função objetivo: sendo c_{ij} o tempo de viagem para ir da cidade i para a cidade j a F. Obj. é dada por

$$z = \min \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x_{ij}$$

Restrições :

- o caixeiro chega à cidade j uma única vez:

$$\sum_{i=1, i \neq j}^{n} x_{ij} = 1, \qquad \forall j \in \{1, \ldots, n\}$$

- o caixeiro sai da cidade i uma única vez:

$$\sum_{j=1, i \neq j}^{n} x_{ij} = 1, \qquad \forall i \in \{1, \ldots, n\}$$

Problema do caixeiro viajante (Traveling Salesman Problem, TSP)

Variáveis :

$$x_{ij} = \begin{cases} 1, & \text{se a cidade } i \text{ é visitada imediatamente antes da cidade } j \\ 0, & \text{caso contrário} \end{cases}$$

As variáveis x_{ii} estão indefinidas.

Função objetivo: sendo c_{ij} o tempo de viagem para ir da cidade i para a cidade j a F. Obj. é dada por

$$z = \min \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x_{ij}$$

Restrições :

- o caixeiro chega à cidade j uma única vez:

$$\sum_{i=1, i \neq j}^{n} x_{ij} = 1, \qquad \forall j \in \{1, \ldots, n\}$$

- o caixeiro sai da cidade i uma única vez:

$$\sum_{j=1, i \neq j}^{n} x_{ij} = 1, \qquad \forall i \in \{1, \ldots, n\}$$

Problema do caixeiro viajante (Traveling Salesman Problem, TSP)

Variáveis :

$$x_{ij} = \begin{cases} 1, & \text{se a cidade } i \text{ é visitada imediatamente antes da cidade } j \\ 0, & \text{caso contrário} \end{cases}$$

As variáveis x_{ii} estão indefinidas.

Função objetivo: sendo c_{ij} o tempo de viagem para ir da cidade i para a cidade j a F. Obj. é dada por

$$z = \min \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x_{ij}$$

Restrições :

- o caixeiro chega à cidade j uma única vez:

$$\sum_{i=1, i \neq j}^{n} x_{ij} = 1, \qquad \forall j \in \{1, \ldots, n\}$$

- o caixeiro sai da cidade i uma única vez:

$$\sum_{j=1, i \neq j}^{n} x_{ij} = 1, \qquad \forall i \in \{1, \ldots, n\}$$

Problema do caixeiro viajante (Traveling Salesman Problem, TSP)

- Até aqui, estamos com o modelo do Problema de Alocação!

O que está faltando?

Temos que eliminar ciclos pequenos

- Alternativa 1: (cut set constraints)

$$\sum_{i \in S} \sum_{j \notin S} x_{ij} \geq 1, \qquad \forall S \subset V,\ S \neq \emptyset$$

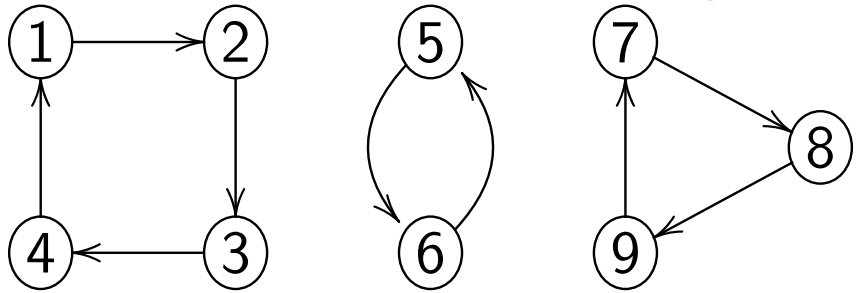
- Alternativa 2: (subtour elimination constraints)

$$\sum_{i \in S} \sum_{j \in S \setminus \{i\}} x_{ij} \leq |S| - 1, \qquad \forall S \subset V,\ 2 \leq |S| \leq |V| - 1$$

- Quantas desigualdades tem neste modelo?

Problema do caixeiro viajante (Traveling Salesman Problem, TSP)

- Até aqui, estamos com o modelo do Problema de Alocação!



O que está faltando?

Temos que eliminar ciclos pequenos

- Alternativa 1: (cut set constraints)

$$\sum_{i \in S} \sum_{j \notin S} x_{ij} \geq 1, \qquad \forall S \subset V,\ S \neq \emptyset$$

- Alternativa 2: (subtour elimination constraints)

$$\sum_{i \in S} \sum_{j \in S \setminus \{i\}} x_{ij} \leq |S| - 1, \qquad \forall S \subset V,\ 2 \leq |S| \leq |V| - 1$$

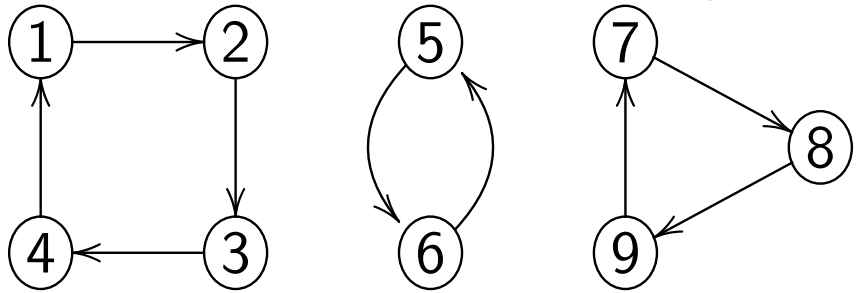
- Quantas desigualdades tem neste modelo?

Problema do caixeiro viajante (Traveling Salesman Problem, TSP)

- Até aqui, estamos com o modelo do Problema de Alocação!

  O que está faltando?

  Temos que eliminar ciclos pequenos



- Alternativa 1: (cut set constraints)

$$\sum_{i \in S} \sum_{j \notin S} x_{ij} \geq 1, \qquad \forall S \subset V,\ S \neq \emptyset$$

- Alternativa 2: (subtour elimination constraints)

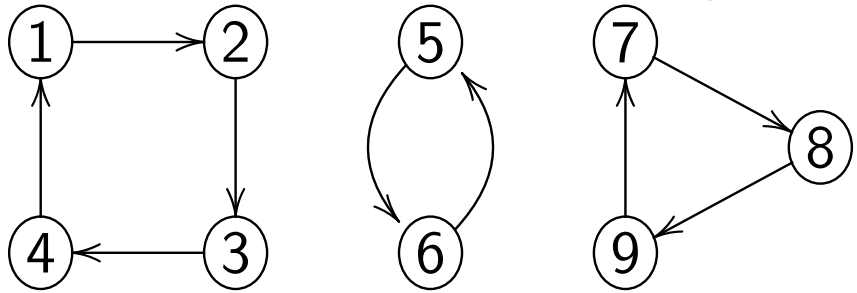
$$\sum_{i \in S} \sum_{j \in S \setminus \{i\}} x_{ij} \leq |S| - 1, \qquad \forall S \subset V,\ 2 \leq |S| \leq |V| - 1$$

- Quantas desigualdades tem neste modelo?

Problema do caixeiro viajante (Traveling Salesman Problem, TSP)

- Até aqui, estamos com o modelo do Problema de Alocação!

  O que está faltando?
  Temos que eliminar
  ciclos pequenos



- Alternativa 1: (cut set constraints)

$$\sum_{i \in S} \sum_{j \notin S} x_{ij} \geq 1, \qquad \forall S \subset V,\ S \neq \emptyset$$

- Alternativa 2: (subtour elimination constraints)

$$\sum_{i \in S} \sum_{j \in S \setminus \{i\}} x_{ij} \leq |S| - 1, \qquad \forall S \subset V,\ 2 \leq |S| \leq |V| - 1$$

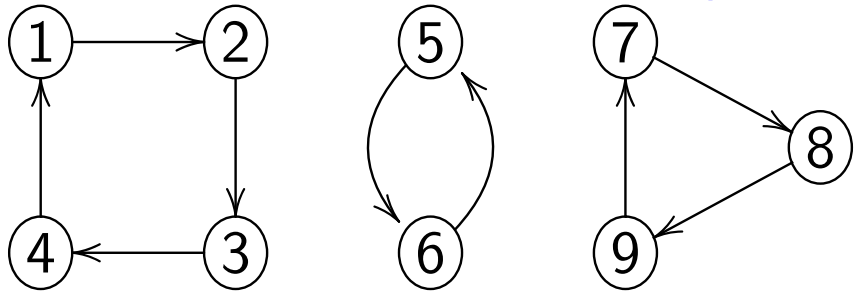
- Quantas desigualdades tem neste modelo?

Problema do caixeiro viajante (Traveling Salesman Problem, TSP)

- Até aqui, estamos com o modelo do Problema de Alocação!

  O que está faltando?

  Temos que eliminar ciclos pequenos



- Alternativa 1: (cut set constraints)

$$\sum_{i \in S} \sum_{j \notin S} x_{ij} \geq 1, \qquad \forall S \subset V,\ S \neq \emptyset$$

- Alternativa 2: (subtour elimination constraints)

$$\sum_{i \in S} \sum_{j \in S \setminus \{i\}} x_{ij} \leq |S| - 1, \qquad \forall S \subset V,\ 2 \leq |S| \leq |V| - 1$$

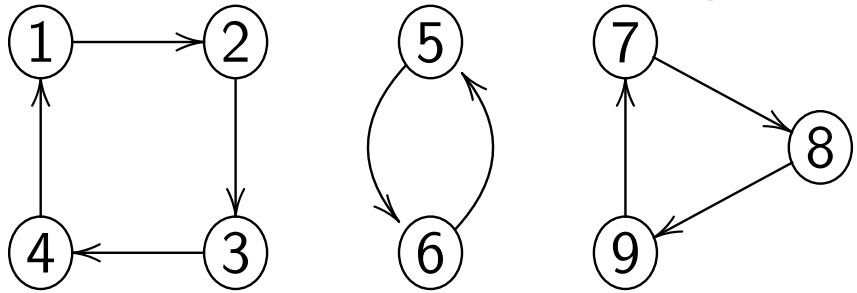
- Quantas desigualdades tem neste modelo?

Problema do caixeiro viajante (Traveling Salesman Problem, TSP)

- Até aqui, estamos com o modelo do Problema de Alocação!

  O que está faltando?

  Temos que eliminar ciclos pequenos



- Alternativa 1: (cut set constraints)

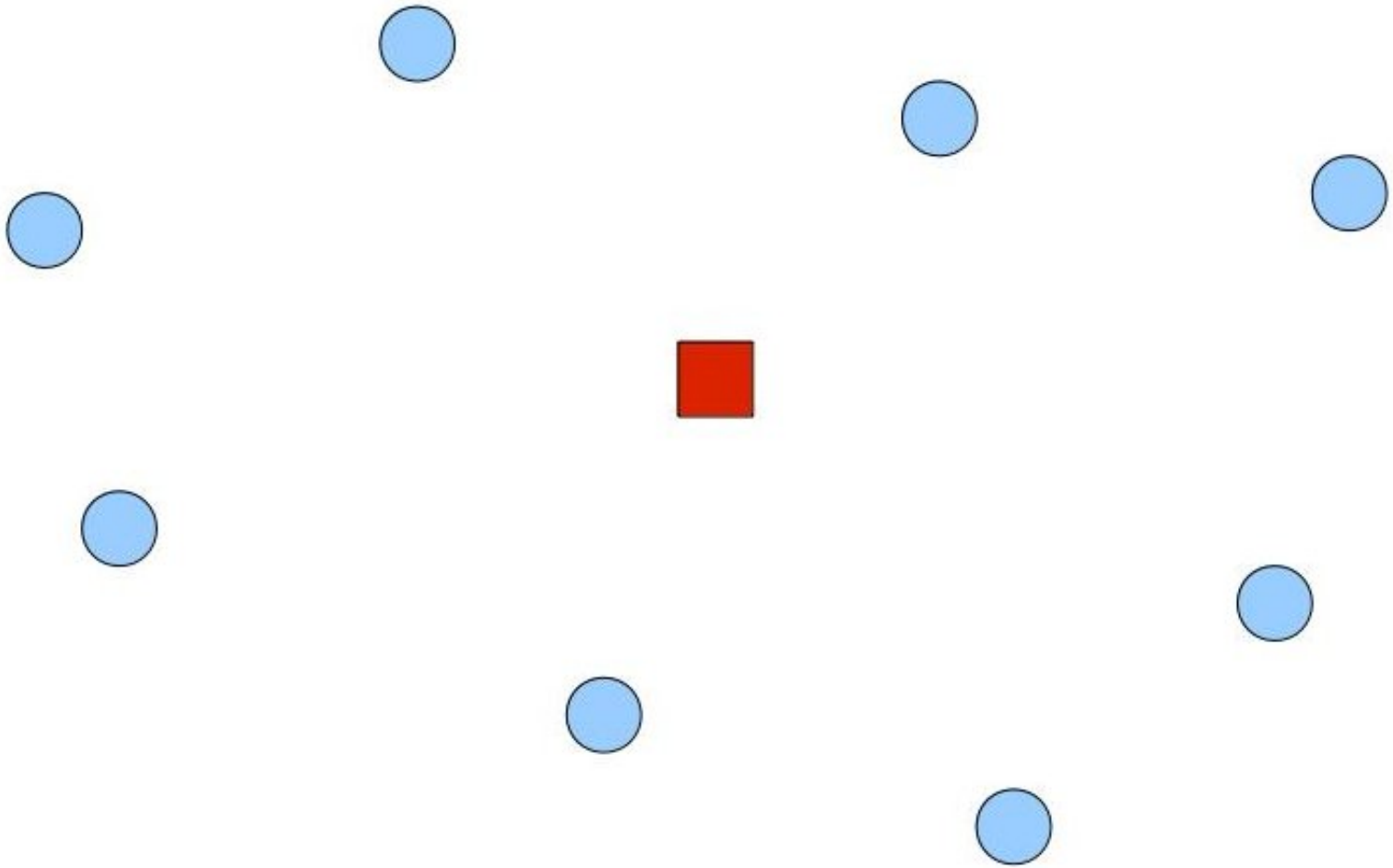
$$\sum_{i \in S} \sum_{j \notin S} x_{ij} \geq 1, \qquad \forall S \subset V,\ S \neq \emptyset$$

- Alternativa 2: (subtour elimination constraints)

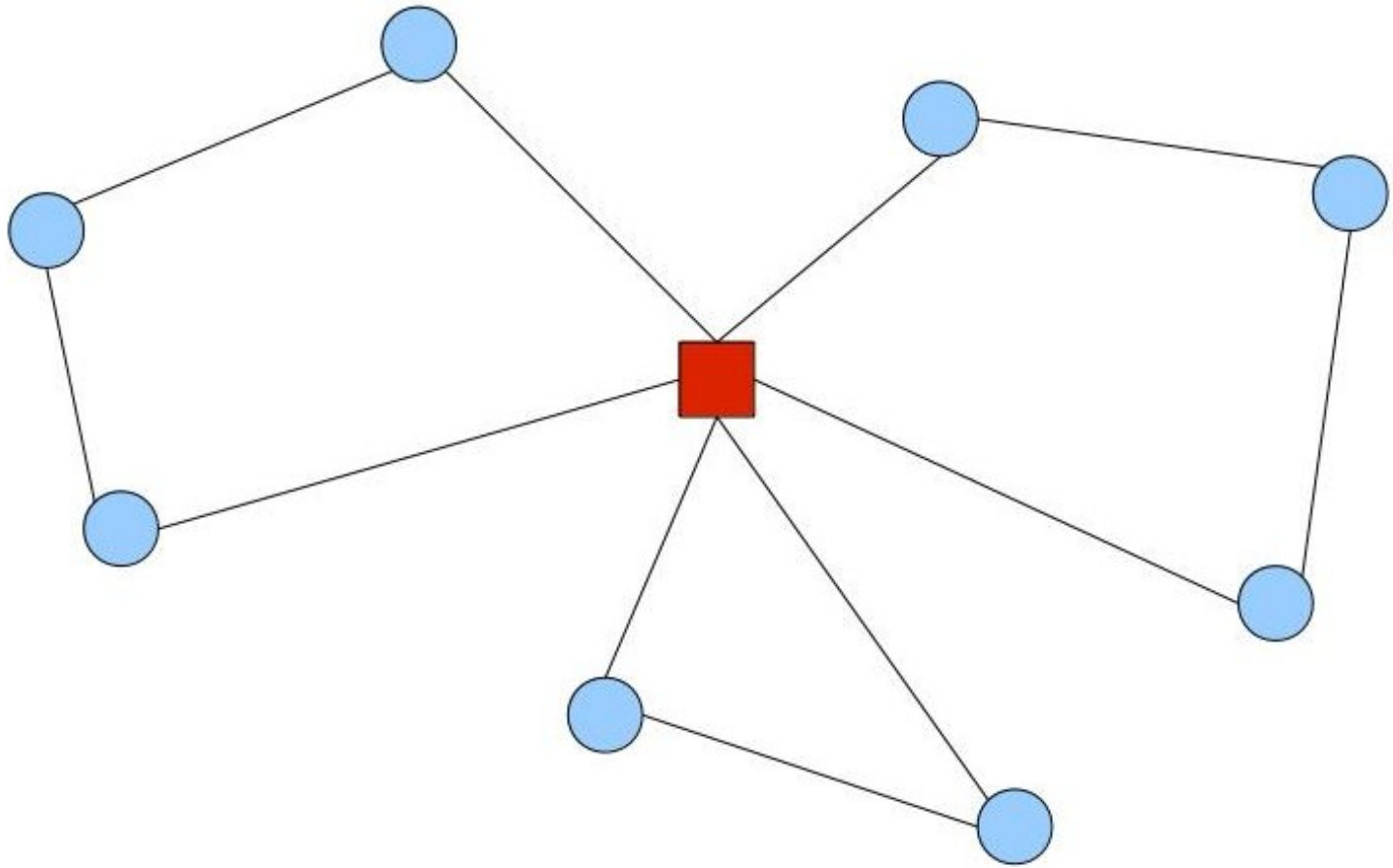
$$\sum_{i \in S} \sum_{j \in S \setminus \{i\}} x_{ij} \leq |S| - 1, \qquad \forall S \subset V,\ 2 \leq |S| \leq |V| - 1$$

- Quantas desigualdades tem neste modelo?

Exemplo: Roteamento de veículos

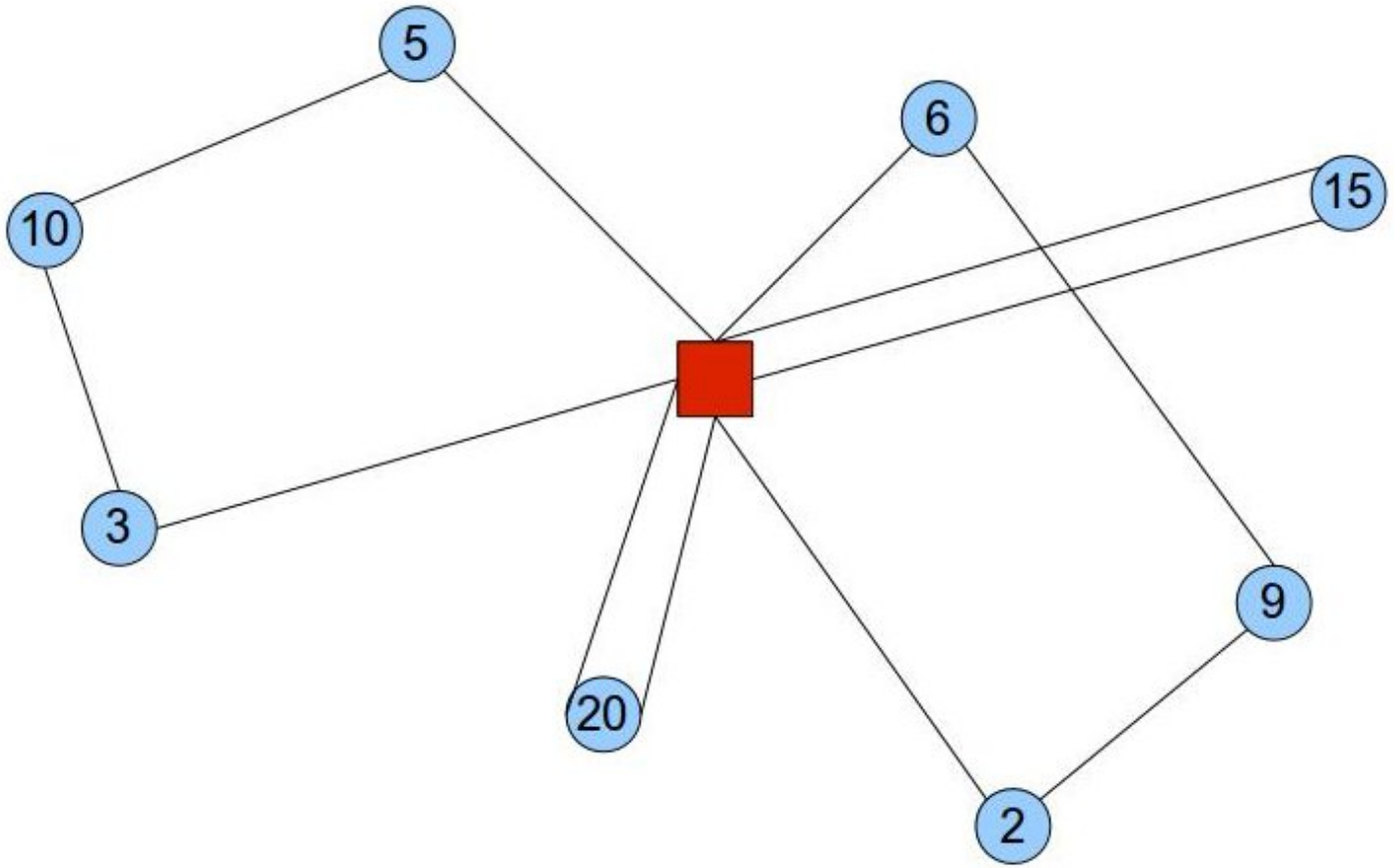


Exemplo: Roteamento de veículos



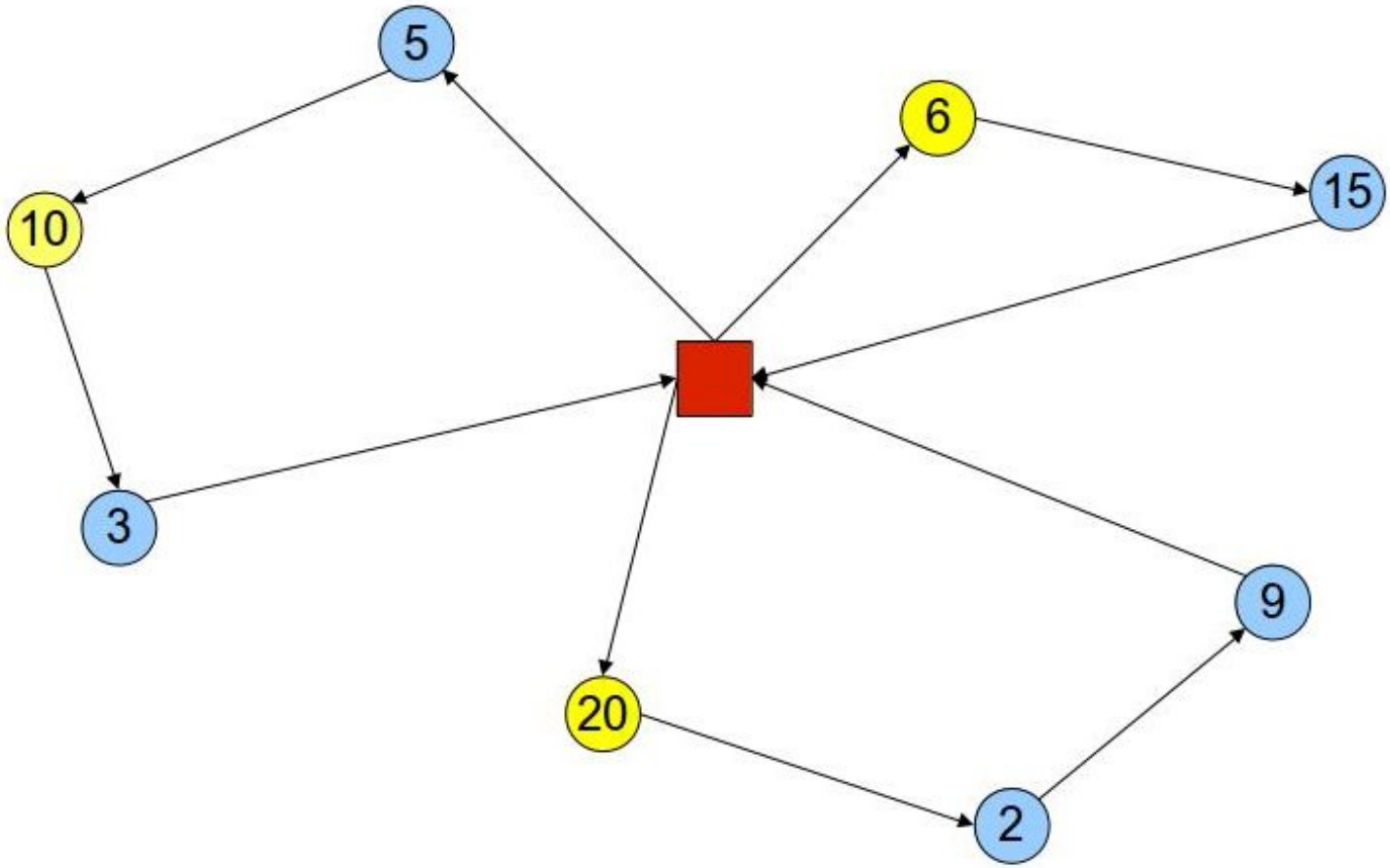
Exemplo: Roteamento de veículos

Variações - Capacitado - Ex: 20



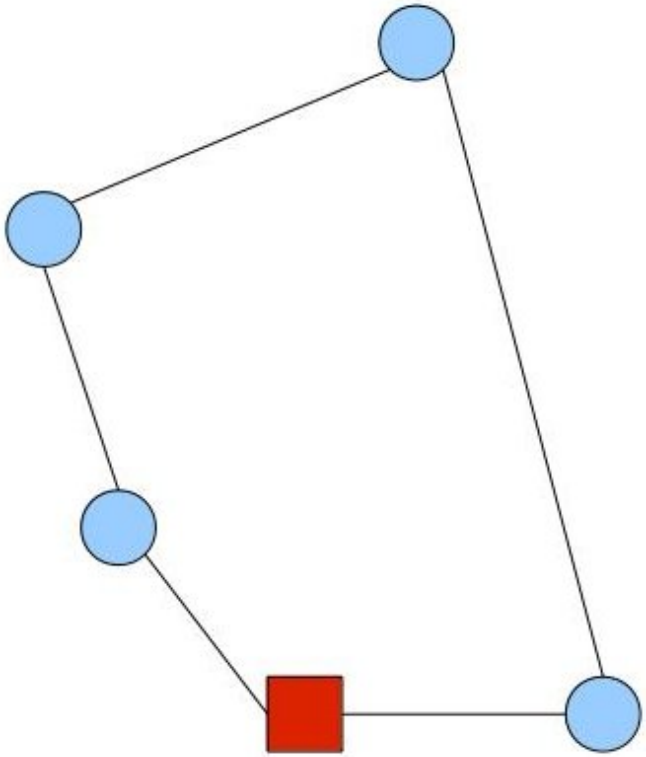
Exemplo: Roteamento de veículos

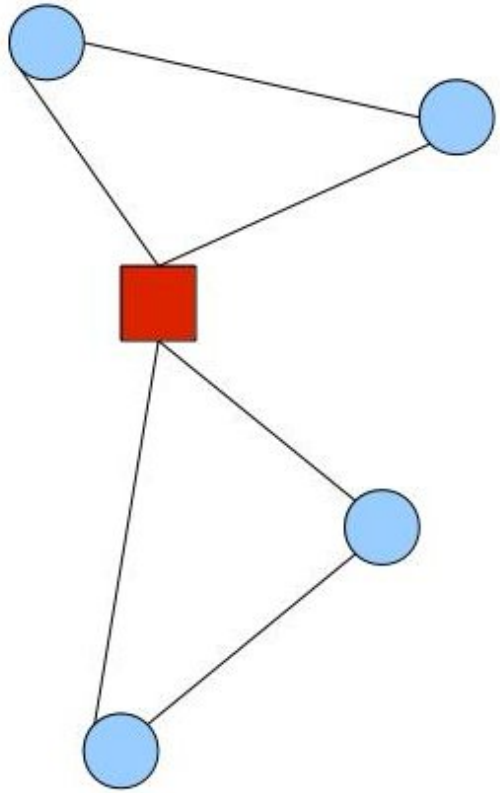
Variações - Coleta e Entrega - Ex: 20



Exemplo: Roteamento de veículos

Variações - Múltiplos Depósitos





Árvore Geradora com número máximo de folhas

Definição

Dado

Um grafo $G = (V, E)$

Achar

A árvore geradora T do grafo com maior número de folhas

Folha

Vértice adjacente a um único vértice

Árvore Geradora com número máximo de folhas

Definição

Dado

Um grafo $G = (V, E)$

Achar

A árvore geradora T do grafo com maior número de folhas

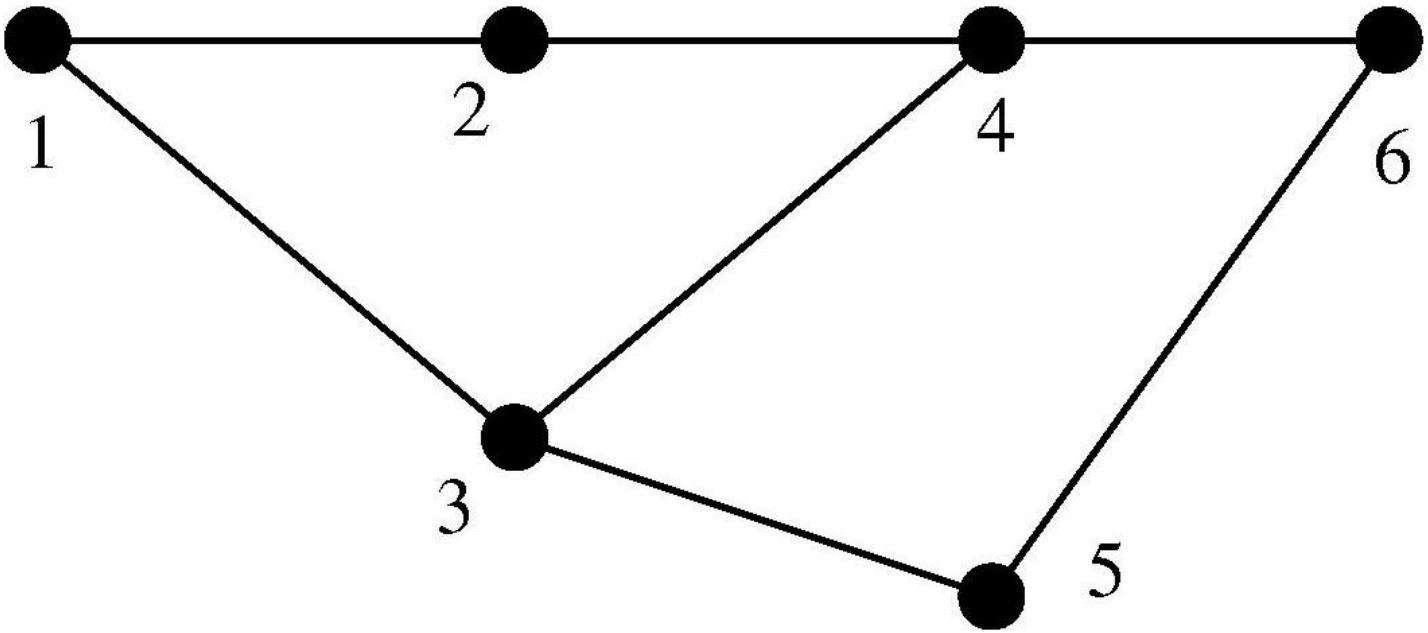
Folha

Vértice adjacente a um único vértice

Definição

Exemplo

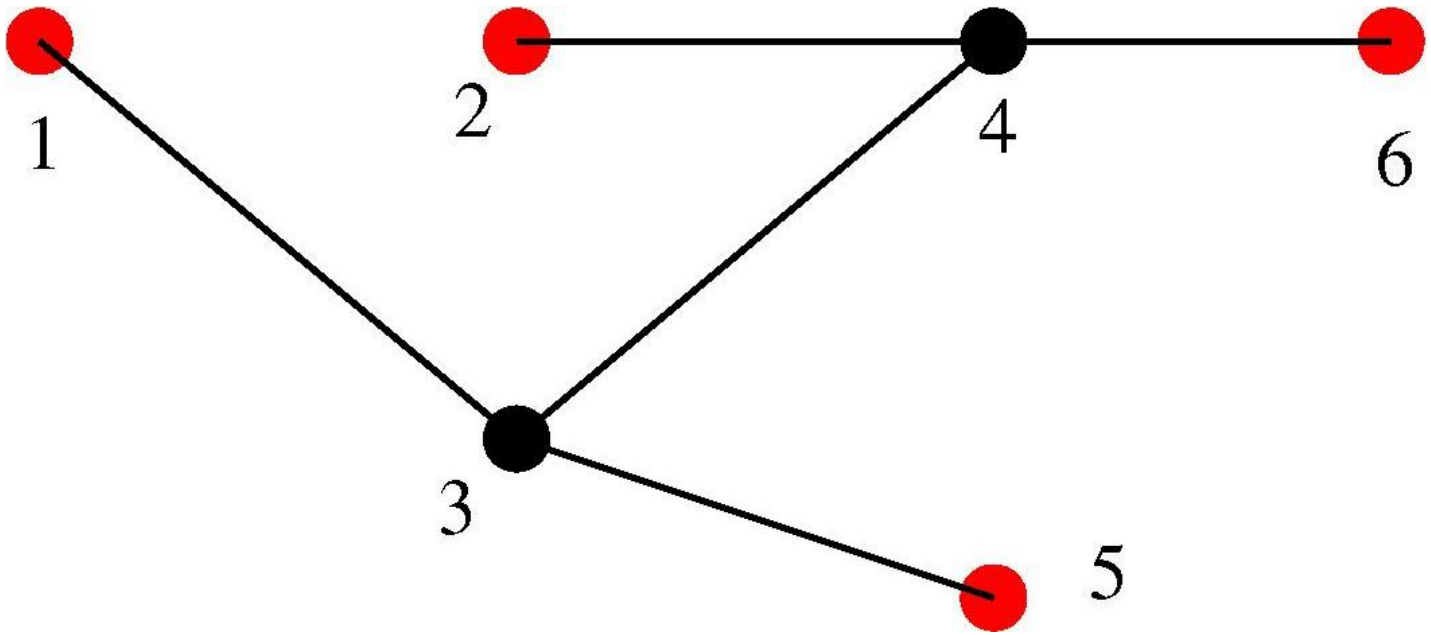
$G(V, E)$



Definição

Exemplo

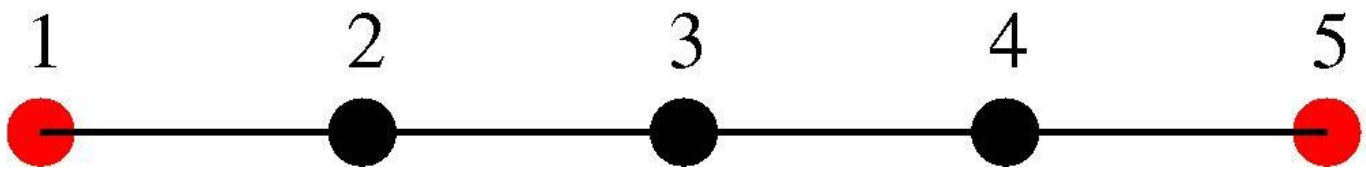
Solução (4 folhas)



Definição

Exemplo

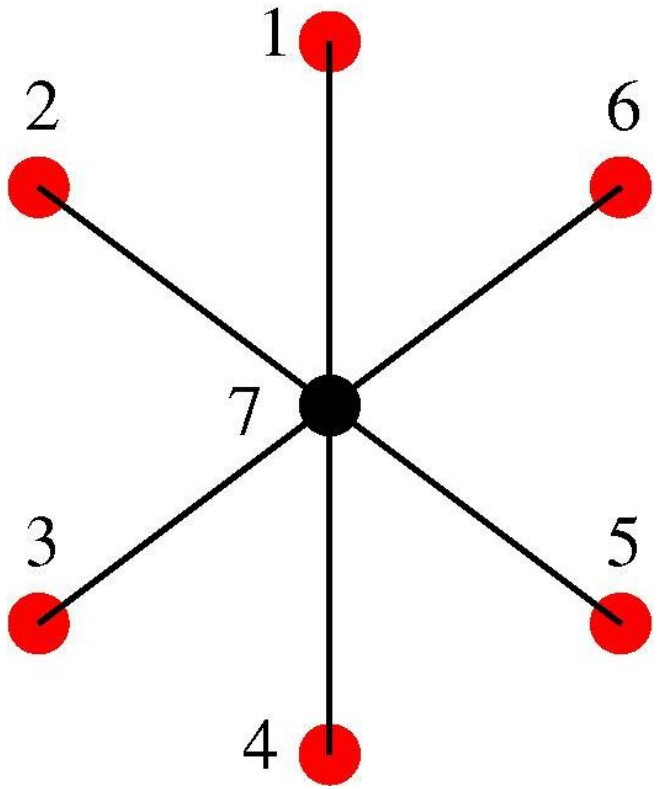
Pior caso



Definição

Exemplo

Melhor caso



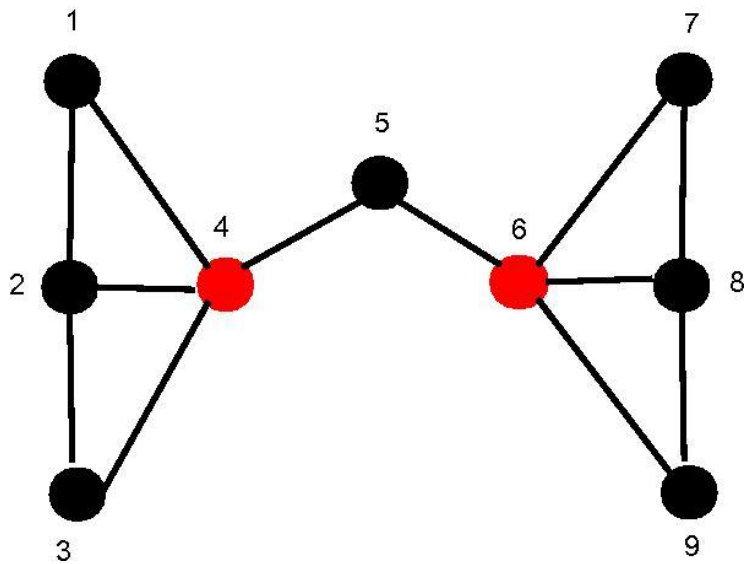
Equivalente ao problema

Conjunto dominante Mínimo de G: $C \subset V$ tal que

- para todo $i \in V \setminus C$: existe uma aresta com uma extremidade em i e outra em C.
- encontrar o conjunto dominante de G com o mínimo número de vértices possível.

Mínimo conjunto dominante conexo

- Implica numa árvore de G sem os vértices folhas $\mathcal{T}$.
- Fácil gerar a árvore geradora com maior número de folhas de G.



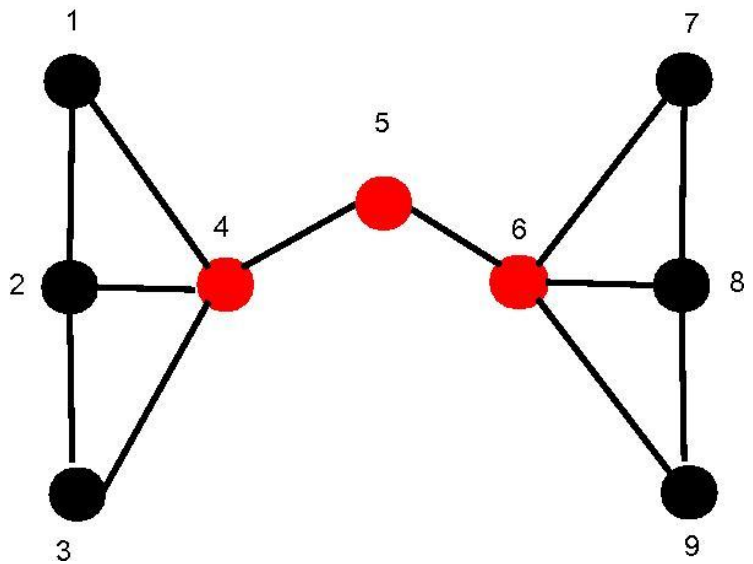
Equivalente ao problema

Conjunto dominante Mínimo de G: $C \subset V$ tal que

- para todo $i \in V \setminus C$: existe uma aresta com uma extremidade em i e outra em C.
- encontrar o conjunto dominante de G com o mínimo número de vértices possível.

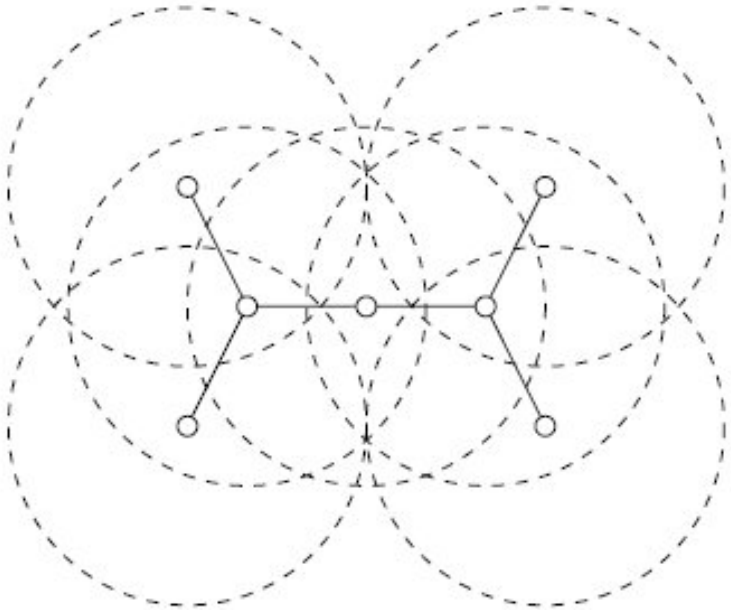
Mínimo conjunto dominante conexo

- Implica numa árvore de G sem os vértices folhas $\mathcal{T}$.
- Fácil gerar a árvore geradora com maior número de folhas de G.



Motivação

- Modela rede de telecomunicação
  Ex: Wireless Ad Hoc Networks
- Layoult de circuitos

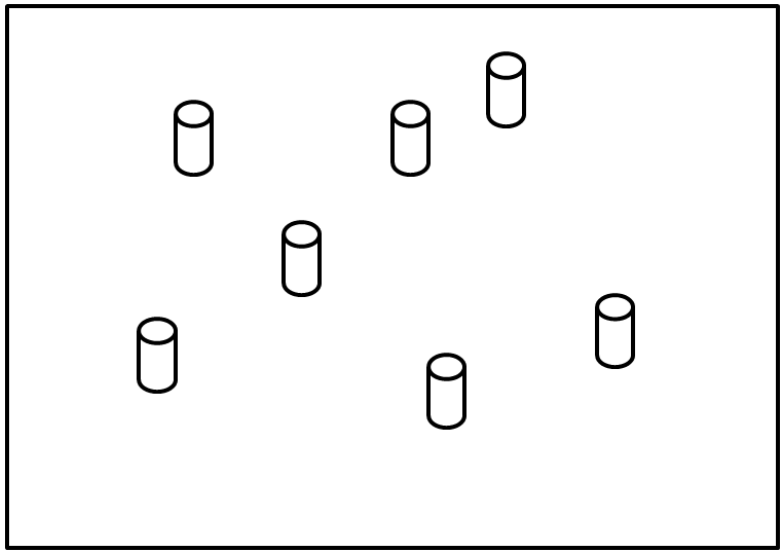


Problema de Alocação de Frequências de celulares

- Problema de alocar frequências a antenas de transmissão
  - Antenas de transmissão
  - Zona de cobertura (antenas possuem interseções entre as zonas)
  - Antenas com interseção nas zonas de cobertura devem receber frequências respeitando as distâncias para não gerar interferências)
  - Objetivo: 1) Minimizar a faixa de frequências utilizadas 2) Minimizar a interferência

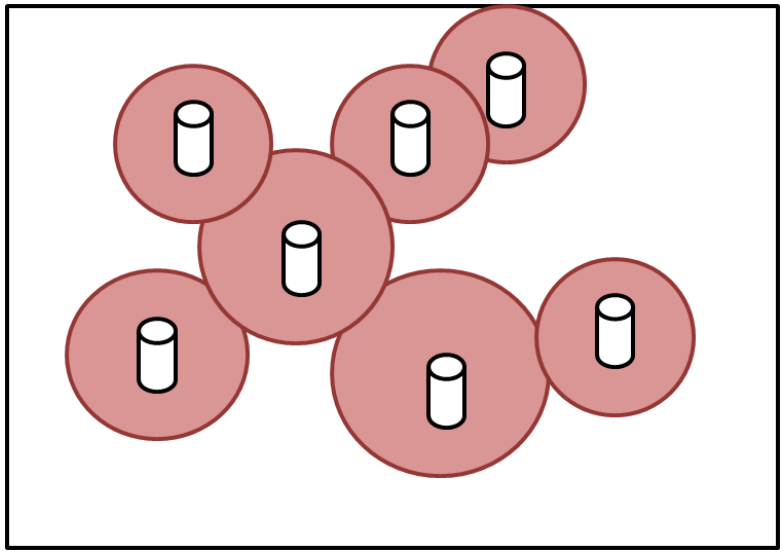
Problema de Alocação de Frequências de celulares

- Problema de alocar frequências a antenas de transmissão
  - Antenas de transmissão
  - Zona de cobertura (antenas possuem interseções entre as zonas)
  - Antenas com interseção nas zonas de cobertura devem receber frequências respeitando as distâncias para não gerar interferências)
  - Objetivo: 1) Minimizar a faixa de frequências utilizadas 2) Minimizar a interferência



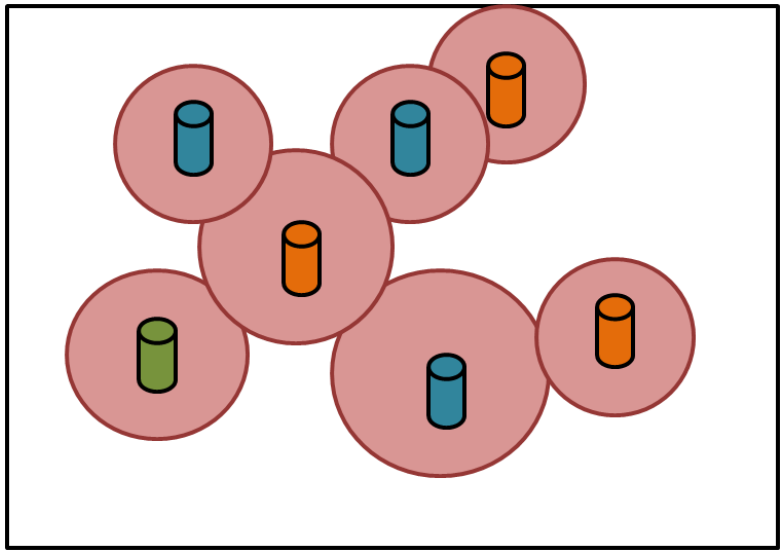
Problema de Alocação de Frequências de celulares

- Problema de alocar frequências a antenas de transmissão
  - Antenas de transmissão
  - Zona de cobertura (antenas possuem interseções entre as zonas)
  - Antenas com interseção nas zonas de cobertura devem receber frequências respeitando as distâncias para não gerar interferências)
  - Objetivo: 1) Minimizar a faixa de frequências utilizadas 2) Minimizar a interferência



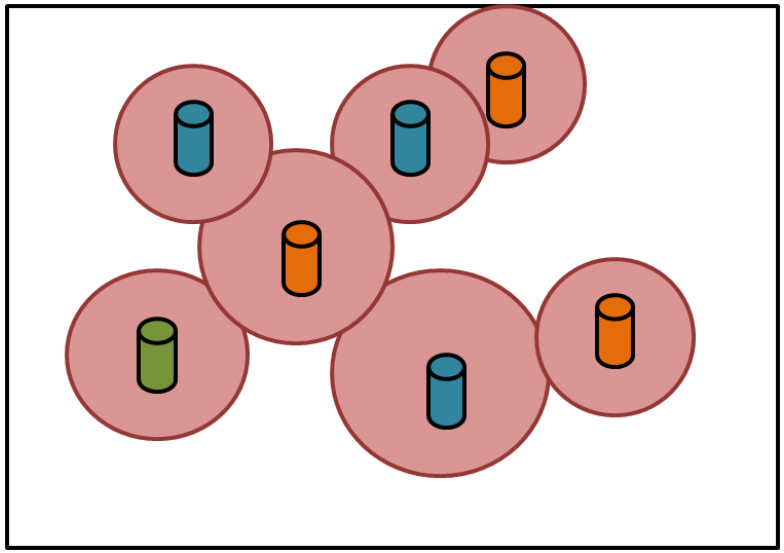
Problema de Alocação de Frequências de celulares

- Problema de alocar frequências a antenas de transmissão
  - Antenas de transmissão
  - Zona de cobertura (antenas possuem interseções entre as zonas)
  - Antenas com interseção nas zonas de cobertura devem receber frequências respeitando as distâncias para não gerar interferências)
  - Objetivo: 1) Minimizar a faixa de frequências utilizadas 2) Minimizar a interferência



Problema de Alocação de Frequências de celulares

- Problema de alocar frequências a antenas de transmissão
  - Antenas de transmissão
  - Zona de cobertura (antenas possuem interseções entre as zonas)
  - Antenas com interseção nas zonas de cobertura devem receber frequências respeitando as distâncias para não gerar interferências)
  - Objetivo: 1) Minimizar a faixa de frequências utilizadas 2) Minimizar a interferência

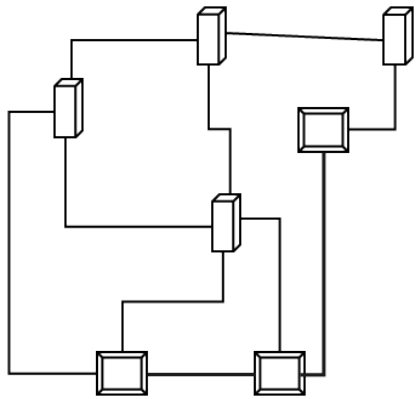


RWA (Routing and Wavelength Assignment)

- Em redes de computadores de alto desempenho, dados são transmitidos entre terminais
- para haver comunicação entre 2 terminais temos 1) criação de um caminho 2) estabelecer qual frequencia os dados serão transmitidos
- Uma fibra óptica pode transmitir dados em várias frequencias diferentes
- Interferência
- Objetivo : 1) Roteamento das comunicações a serem realizadas 2) Atribuição de frequências sem interferência

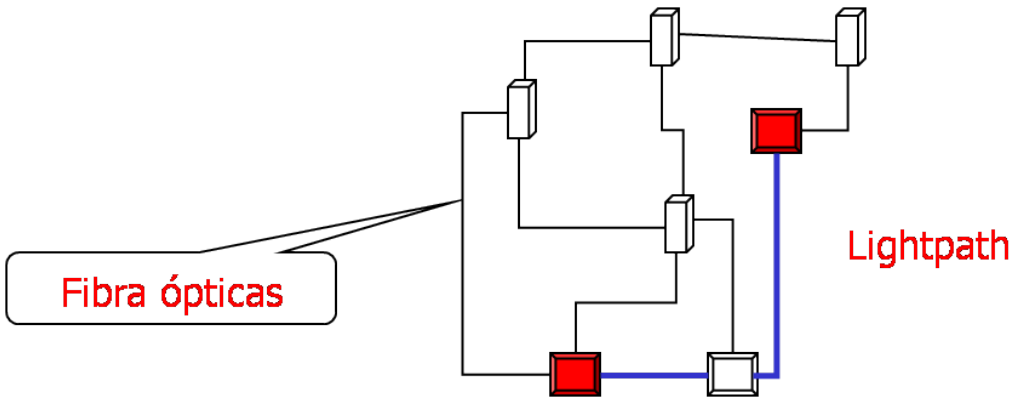
RWA (Routing and Wavelength Assignment)

- Em redes de computadores de alto desempenho, dados são transmitidos entre terminais
- para haver comunicação entre 2 terminais temos 1) criação de um caminho 2) estabelecer qual frequencia os dados serão transmitidos
- Uma fibra óptica pode transmitir dados em várias frequencias diferentes
- Interferência
- Objetivo : 1) Roteamento das comunicações a serem realizadas 2) Atribuição de frequências sem interferência



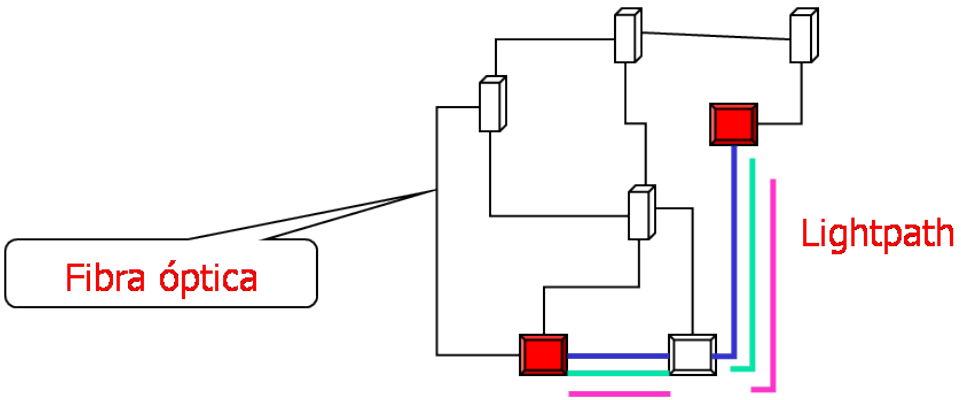
RWA (Routing and Wavelength Assignment)

- Em redes de computadores de alto desempenho, dados são transmitidos entre terminais
- para haver comunicação entre 2 terminais temos 1) criação de um caminho 2) estabelecer qual frequencia os dados serão transmitidos
- Uma fibra óptica pode transmitir dados em várias frequencias diferentes
- Interferência
- Objetivo : 1) Roteamento das comunicações a serem realizadas 2) Atribuição de frequências sem interferência



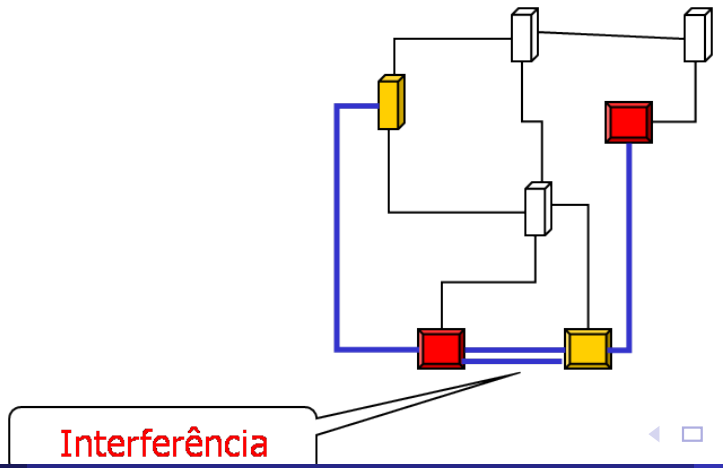
RWA (Routing and Wavelength Assignment)

- Em redes de computadores de alto desempenho, dados são transmitidos entre terminais
- para haver comunicação entre 2 terminais temos 1) criação de um caminho 2) estabelecer qual frequencia os dados serão transmitidos
- Uma fibra óptica pode transmitir dados em várias frequencias diferentes
- Interferência
- Objetivo : 1) Roteamento das comunicações a serem realizadas 2) Atribuição de frequências sem interferência



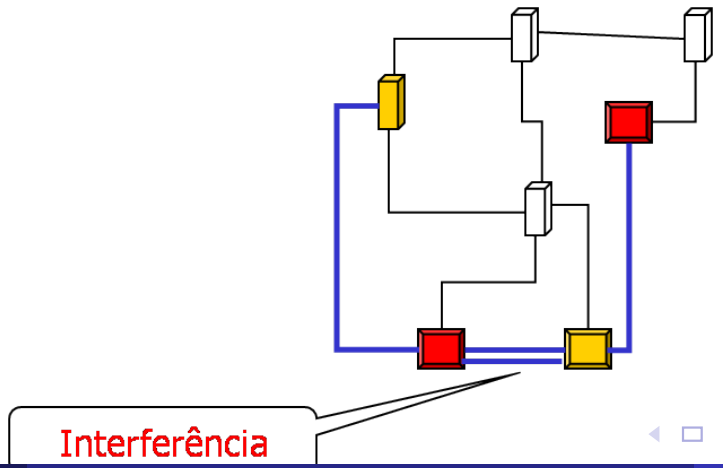
RWA (Routing and Wavelength Assignment)

- Em redes de computadores de alto desempenho, dados são transmitidos entre terminais
- para haver comunicação entre 2 terminais temos 1) criação de um caminho 2) estabelecer qual frequencia os dados serão transmitidos
- Uma fibra óptica pode transmitir dados em várias frequencias diferentes
- Interferência
- Objetivo : 1) Roteamento das comunicações a serem realizadas 2) Atribuição de frequências sem interferência



RWA (Routing and Wavelength Assignment)

- Em redes de computadores de alto desempenho, dados são transmitidos entre terminais
- para haver comunicação entre 2 terminais temos 1) criação de um caminho 2) estabelecer qual frequencia os dados serão transmitidos
- Uma fibra óptica pode transmitir dados em várias frequencias diferentes
- Interferência
- Objetivo : 1) Roteamento das comunicações a serem realizadas 2) Atribuição de frequências sem interferência



Obrigado