Análise Projeto de Algoritmos

Loana Tito Nogueira

21 de maio de 2012

Algoritmo Gulosp

Uma das técnicas de construção de algoritmos heurísticos mais explorada é tentar obter uma boa solução (eventualmenete ótima), considerando a casa iteração a melhor decisão um passo à frente ou seja utilizando um critério de otimização meramente local.

Heurísticas deste tipo são chamadas Míopes ou Gulosas

Algoritmo Guloso

- Questão Importante: Discutir em quais casos, ou para quais classes de problemas, uma heurística do tipo míope garante a obtenção da solução ótima para qualquer instância do problema.

Algoritmo Guloso - Características

- Resolve problemas de otimização
  - Dados um conjunto finito E={1, 2,..., n}, uma coleção de subconjuntos F $\subseteq 2^E$ e uma função c: $2^E \rightarrow R$, determinar o conjunto S* $\in$ F, satisfazendo a
    - c(S*) $\geq$ c(S), $\forall$ S $\in$ F, é um Problema de Otimização combinatória.

    - Problema Geral: dado um conjunto C, determinar se um subconjunto S $\subseteq$ C tal que:
  - S satisfaz uma dada propriedade P, e
  - S é mínimo (ou máximo) em relação a algum critério

Algoritmo Guloso - Características

- Não necessitar usar procedimentos sofisticados para desfazer decisões tomadas previamente;

- O algoritmo guloso para resolver o problema geral consiste em um processo iterativo em que S é construído adicionando-se ao mesmo elementos de C um a um.

Algoritmo Guloso - Exemplo de Aplicaçao

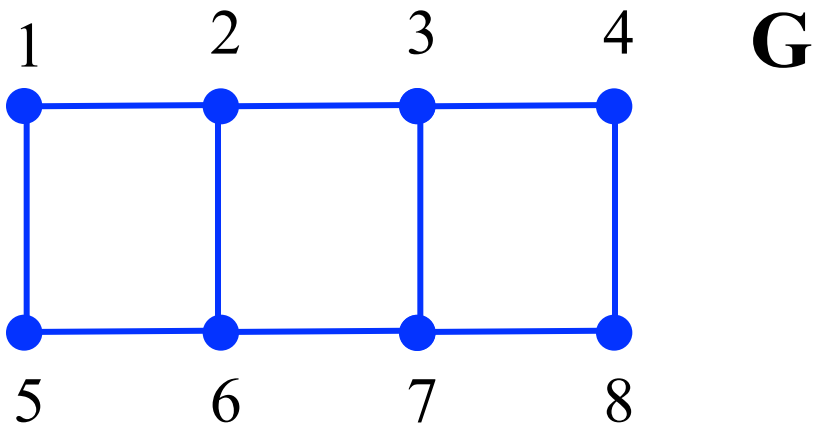
- Vamos utilizar um algoritmo guloso para resolver o problema da árvore geradora mínima.

Definições Básicas

Um grafo G é um par (V,E), onde V é um conjunto de pontos, também chamados de vértices e E é um conjunto de linhas ligando esses pontos, chamados arestas.

Definições Básicas

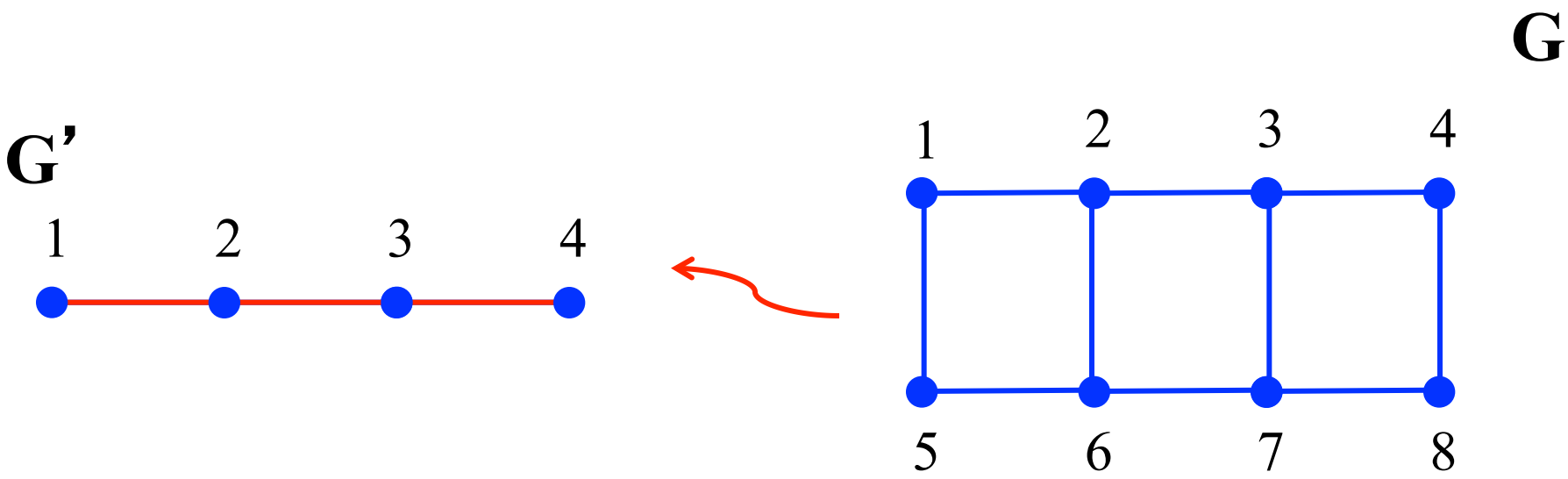
Um grafo G é um par (V,E), onde V é um conjunto de pontos, também chamados de vértices e E é um conjunto de linhas ligando esses pontos, chamados arestas.



Definições Básicas

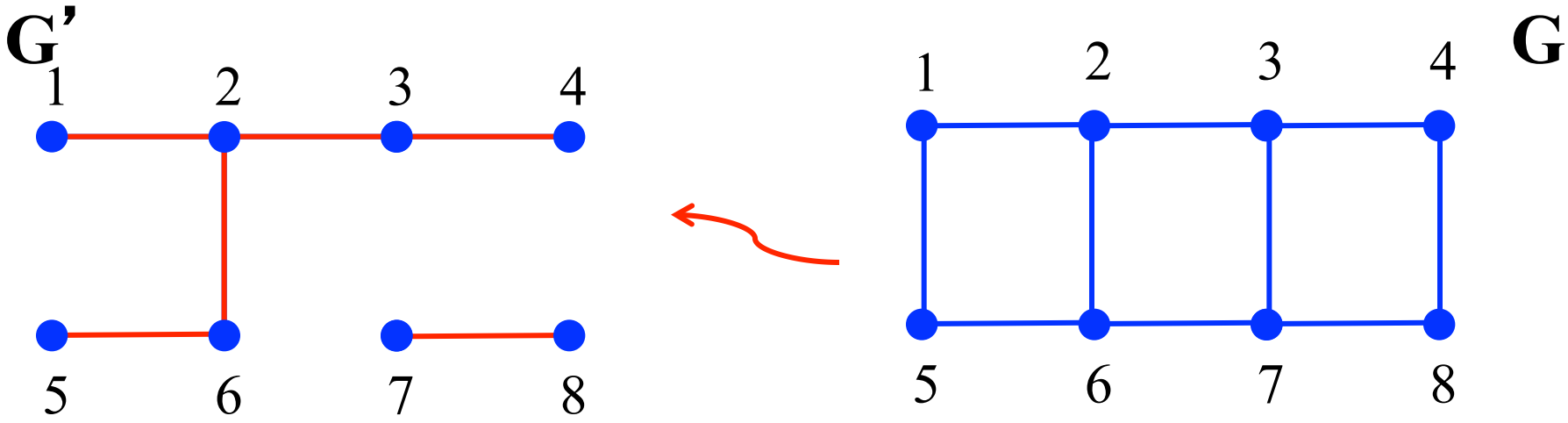
Dizemos que um grafo G' é um subgrafo de um grafo G se:

- V(G') ⊆ V(G) e
- E(G') ⊆ E(G).



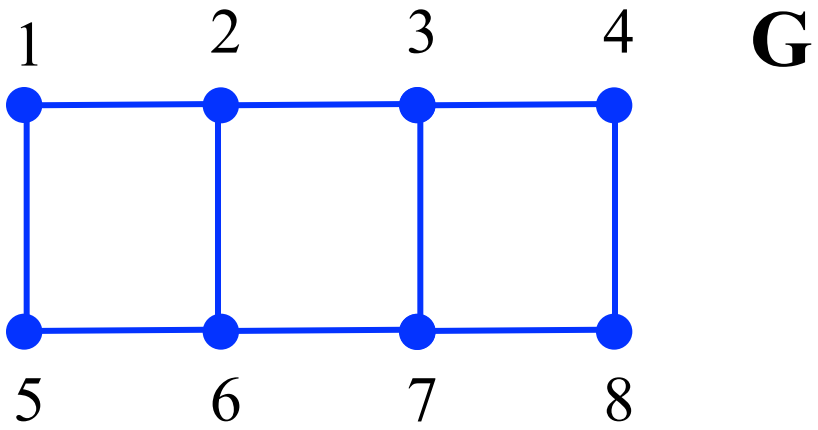
Definições Básicas

Em particular, um grafo G' é um subgrafo gerador de um grafo G se G' ⊆ G e além disso, V(G') = V(G).



Definições Básicas

Um grafo G é conexo se para todo par de vértices distintos v e w de G, existe um caminho de v a w.

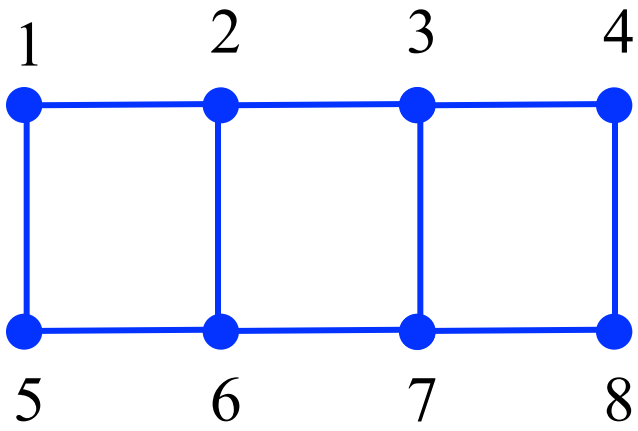


Definições Básicas

Um grafo G é acíclico se G não contém um ciclo como subgrafo.

Definições Básicas

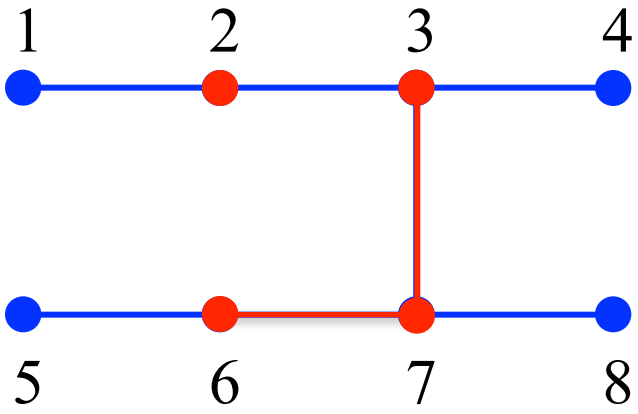
Um grafo G é acíclico se G não contém um ciclo como subgrafo.



G não é acíclico

Definições Básicas

Um grafo G é acíclico se G não contém um ciclo como subgrafo.

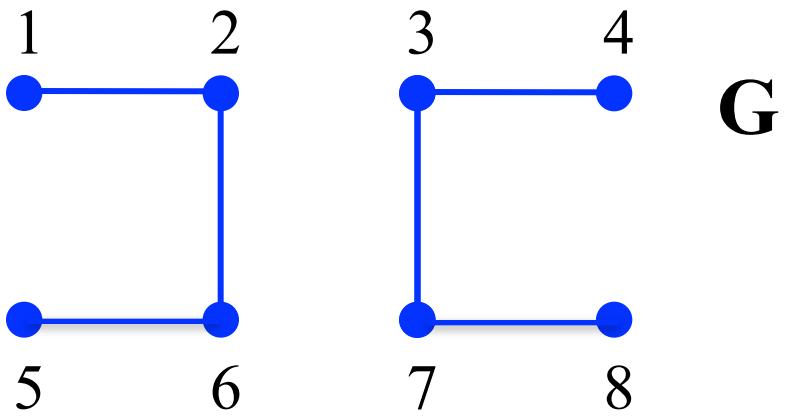


G

G é acíclico

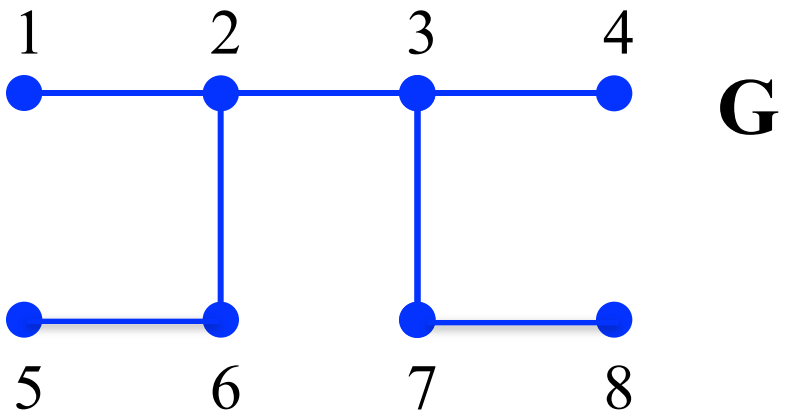
Definições Básicas

Uma floresta F é um grafo acíclico.



Definições Básicas

Uma árvore T é um grafo acíclico e conexo.

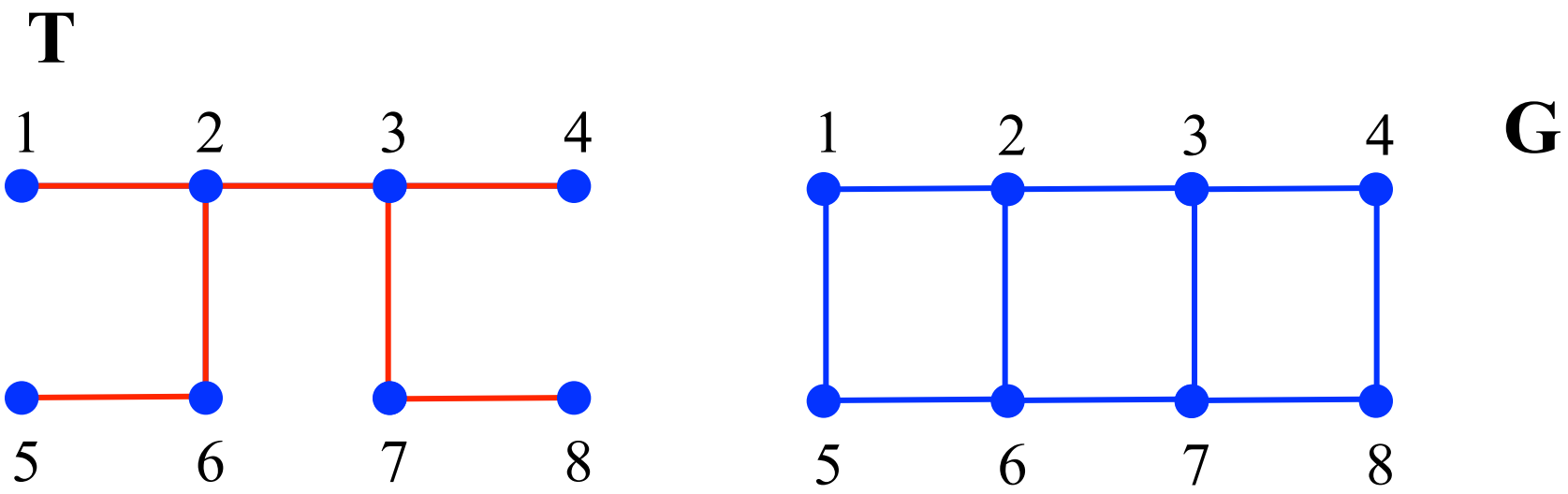


Definições Básicas

Uma árvore geradora T de um grafo G é um árvore tal que $V(G)=V(T)$ e $E(T) \subseteq E(G)$.

Uma árvore geradora T é um subgrafo gerador de G que é acíclico e conexo.

Exemplo: Árvore Geradora



Árvore Geradora Mínima

- Dados:

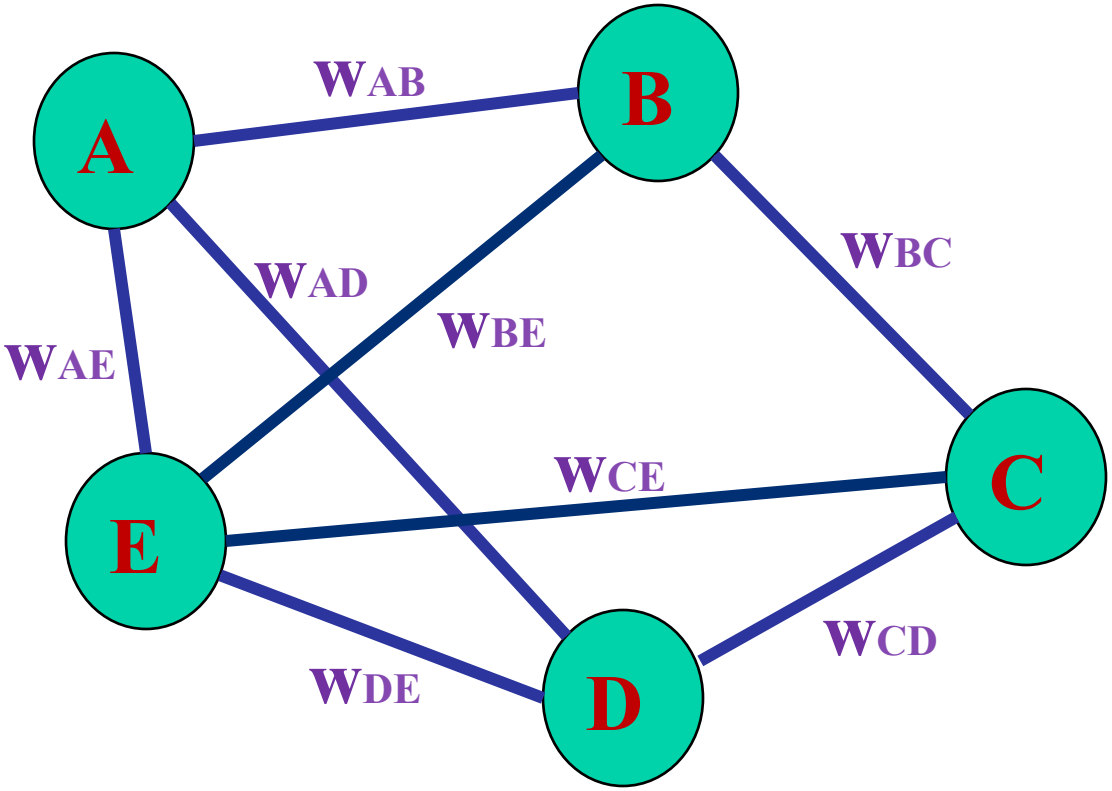
  $\begin{cases} G = (V,E) \text{ grafo não-orientado, com } |V|=n \text{ e } |E|=m \\ \text{peso } w(e), \forall e \in E(G) \end{cases}$

- Problema

  Obter $F \subseteq E$ tal que:

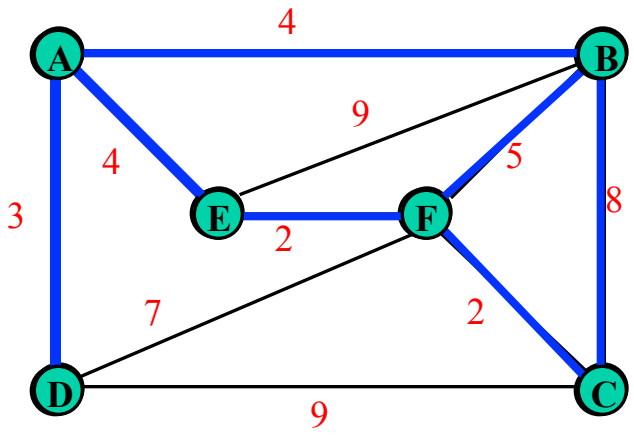
  - o grafo $G'=(V,F)$ é acíclico e conexo (G' é gerador de G)
  - $w(F) = \Sigma_{e \in E} c(e)$ é mínimo

Árvore Geradora Mínima



Árvore Geradora Mínima

Exemplo:



4 4

4 5 4

3 8 3

2

2

árvore geradora
peso = 24

árvore geradora
peso = 15

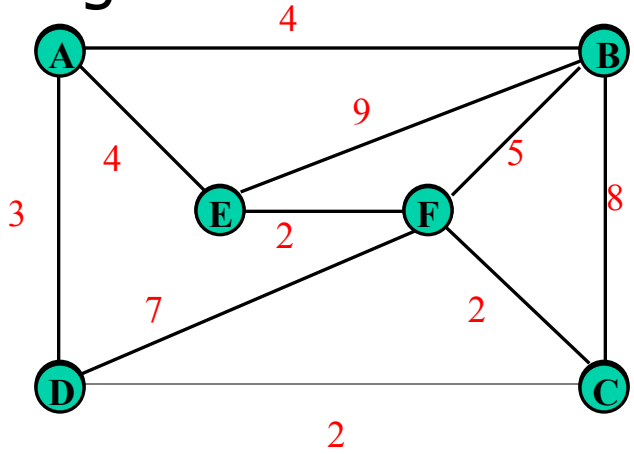
Árvore Geradora Mínima

- A árvore geradora mínima não é única

Árvore Geradora Mínima

- A árvore geradora mínima não é única

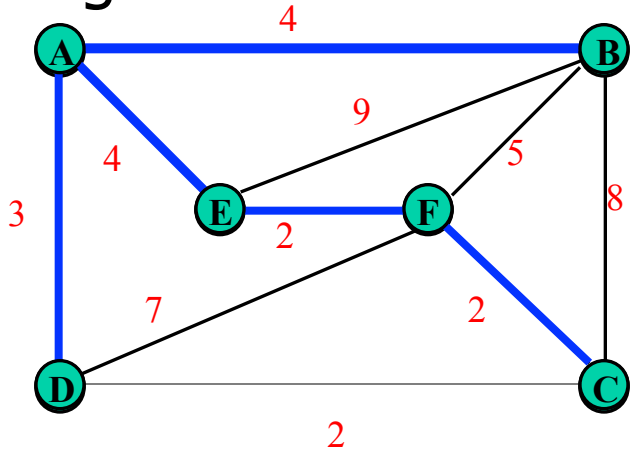
Exemplo:



Árvore Geradora Mínima

- A árvore geradora mínima não é única

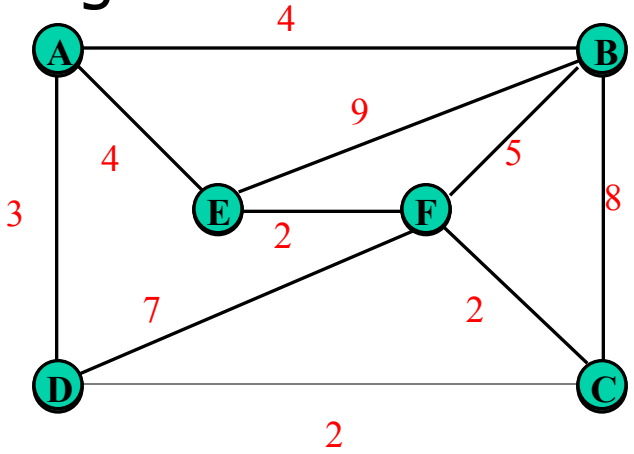
Exemplo:

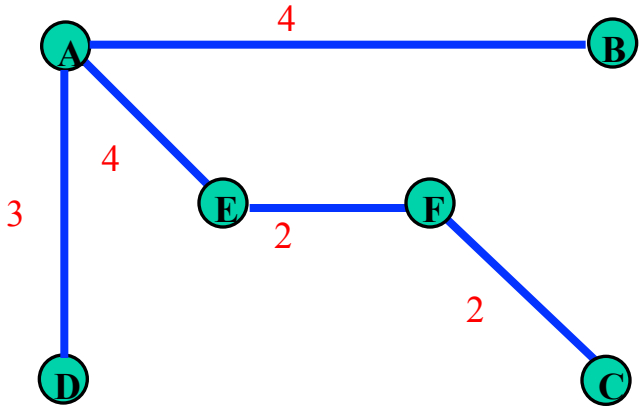


Árvore Geradora Mínima

- A árvore geradora mínima não é única

Exemplo:



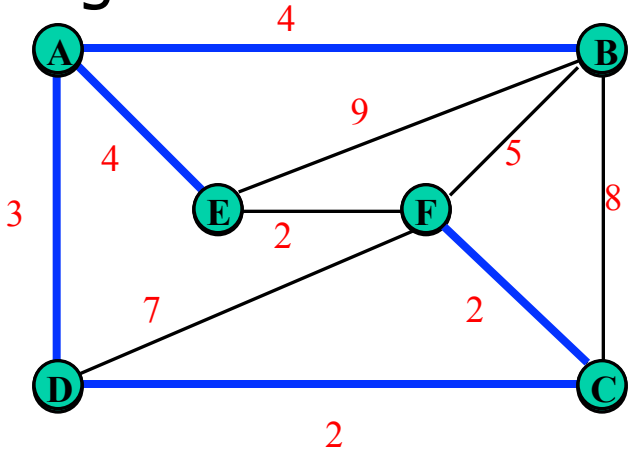


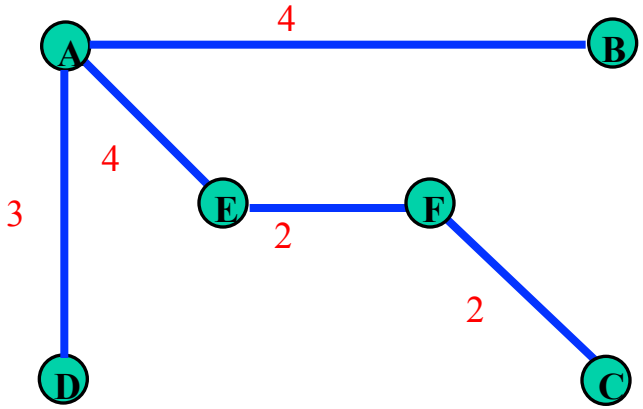
AGM de peso = 15

Árvore Geradora Mínima

- A árvore geradora mínima não é única

Exemplo:



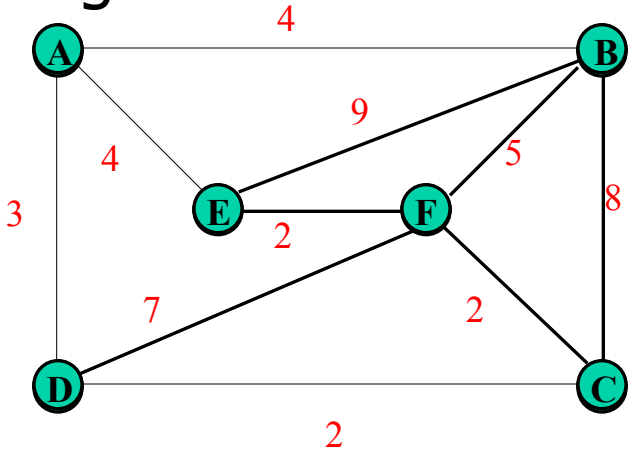


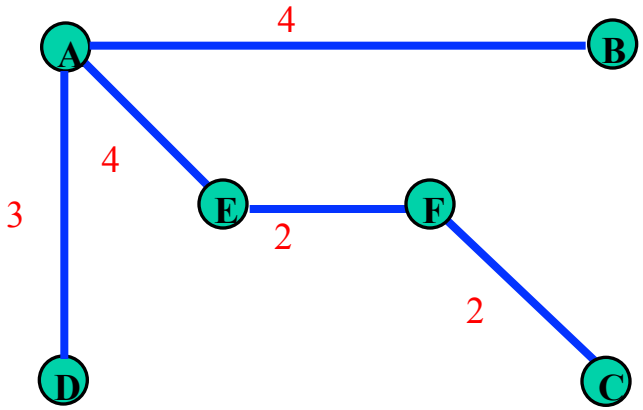
AGM de peso = 15

Árvore Geradora Mínima

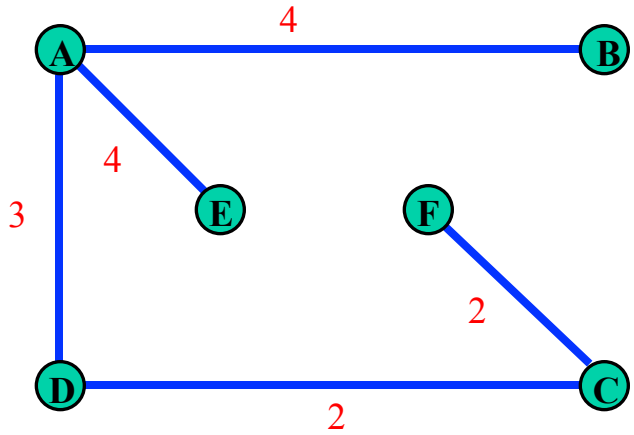
- A árvore geradora mínima não é única

Exemplo:





AGM de peso = 15



AGM de peso 15

Algoritmo de Kruskal

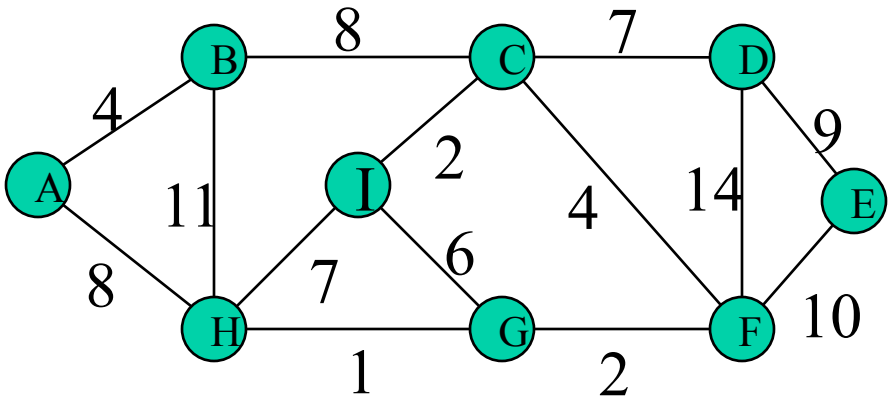
Idéia do Algoritmo

- Em cada passo adiciona uma aresta de peso mínimo de maneira a não formar ciclo;

Algoritmo de Kruskal

Idéia do Algoritmo

- Em cada passo adiciona uma aresta de peso mínimo de maneira a não formar ciclo;

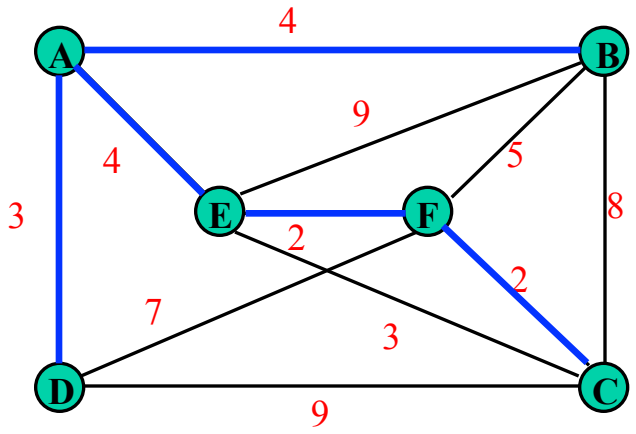


Algoritmo de Kruskal

```
A ← ∅
For cada vértice v ∈ V(G) do
    L(v) ← ∅;
End-for
Ordenar as arestas em ordem não decrescente
pelos pesos.
For cada aresta (u,v) ∈ E(G), tomadas pela ordem
não-decrescente de pesos do
    If L(u)≠ L(v) then
        A ← A ∪ {(u,v)};
        Faça L(u) ∪ L(v) ;
    end-if
end-for
Return A
```

Algoritmo de Kruskal

Exemplo:



c((A))=215

Subárvor

{ A, B, C, D, E, F }

Lista L

e	c(e)
(C,F)	2
(E,F)	2
(A,D)	3
(C,E)	3
(A,B)	4
(A,E)	4
(B,F)	5
(D,F)	7
(B,C)	8
(B,E)	9
(C,D)	9

Algoritmo de Kruskal

Teorema 1: Seja G=(V,E) um grafo conexo. Seja $T=(V,E_T)$ o subgrafo obtido pela aplicação do algoritmo de Kruskal. Então T é uma árvore geradora de G.

Algoritmo de Kruskal

Teorema 1: Seja G=(V,E) um grafo conexo. Seja $T=(V,E_T)$ o subgrafo obtido pela aplicação do algoritmo de Kruskal. Então T é uma árvore geradora de G.

Prova: - T é um subgrafo gerador acíclico.

Algoritmo de Kruskal

Prova: - T é um subgrafo gerador acíclico.

- Resta mostrar que T é conexo.
- Suponha, por contradição, que T é desconexo.
- Sejam T’ e T” duas árvores distintas da floresta T.
- A primeira aresta (v,w) da sequência ordenada de arestas tal que v está em T’ e w em T”, quando adicionada a T não pode produzir ciclo.

Consequentemente, (v,w) não pode ser rejeitada pelo algoritmo guloso.

- Logo, G é desconexo.
- Contradição!!!

Algoritmo de Kruskal

Teorema 2: Seja G um grafo conexo e T a árvore geradora obtida pela aplicação do algoritmo de Kruskal. Então T possui peso mínimo.

Algoritmo de Kruskal

Teorema 2: Seja G um grafo conexo e T a árvore geradora obtida pela aplicação do algoritmo de Kruskal. Então T possui peso mínimo.

Prova:

- ✓ Suponha, por contradição, que a árvore geradora T encontrada pelo algoritmo de Kruskal não é mínima.

Algoritmo de Kruskal

Teorema 2: Seja G um grafo conexo e T a árvore geradora obtida pela aplicação do algoritmo de Kruskal. Então T possui peso mínimo.

Prova:

- ✓ Suponha, por contradição, que a árvore geradora T encontrada pelo algoritmo de Kruskal não é mínima.
- ✓ Sejam e_1, e_2,...,e_m as arestas de T ordenadas não-descrescentes, isto é, $w(e_1) \leq w(e_2) \leq \ldots \leq w(e_m)$.

Algoritmo de Kruskal

Teorema 2: Seja G um grafo conexo e T a árvore geradora obtida pela aplicação do algoritmo de Kruskal. Então T possui peso mínimo.

Prova:

- ✓ Suponha, por contradição, que a árvore geradora T encontrada pelo algoritmo de Kruskal não é mínima.
- ✓ Sejam e_1, e_2,...,e_m as arestas de T ordenadas não-descrescentes, isto é, $w(e_1) \leq w(e_2) \leq \ldots \leq w(e_m)$.
- ✓ Seja T' a árvore geradora mínima.

Algoritmo de Kruskal

Teorema 2: Seja G um grafo conexo e T a árvore geradora obtida pela aplicação do algoritmo de Kruskal. Então T possui peso mínimo.

Prova:

- ✓ Suponha, por contradição, que a árvore geradora T encontrada pelo algoritmo de Kruskal não é mínima.
- ✓ Sejam e_1, e_2,...,e_m as arestas de T ordenadas não-descrescentes, isto é, $w(e_1) \leq w(e_2) \leq \ldots \leq w(e_m)$.
- ✓ Seja T' a árvore geradora mínima.
- ✓ Sejam e'_1, e'_2,...,e'_m as arestas de T' ordenadas não-descrescentes, isto é, $w(e'_1) \leq w(e'_2) \leq \ldots \leq w(e'_m)$.

Algoritmo de Kruskal

- ✓ Seja j o maior índice tal que:

$e_1 = e'_1$

$e_2 = e'_2$

.
.
.

$e_j = e'_j$

$e_{j+1} \neq e'_{j+1}$

Algoritmo de Kruskal

- ✓ Seja j o maior índice tal que:

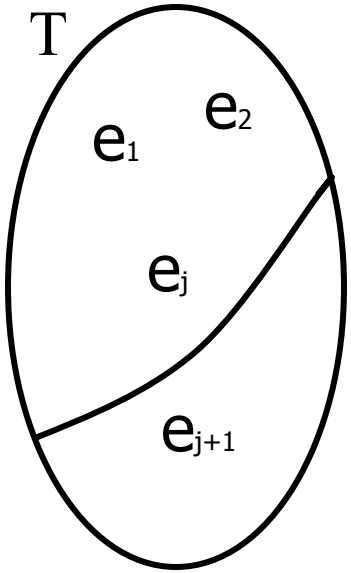
$e_1 = e'_1$

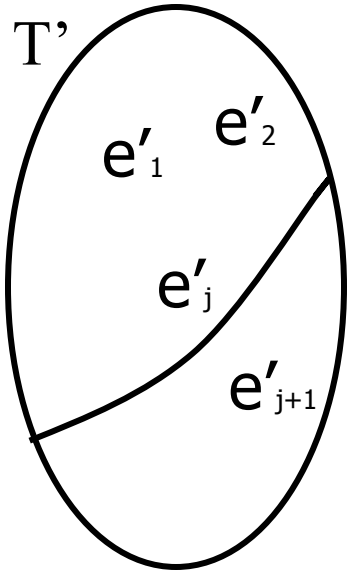
$e_2 = e'_2$

.
.
.

$e_j = e'_j$

$e_{j+1} \neq e'_{j+1}$





Algoritmo de Kruskal

- ✓ Seja j o maior índice tal que:

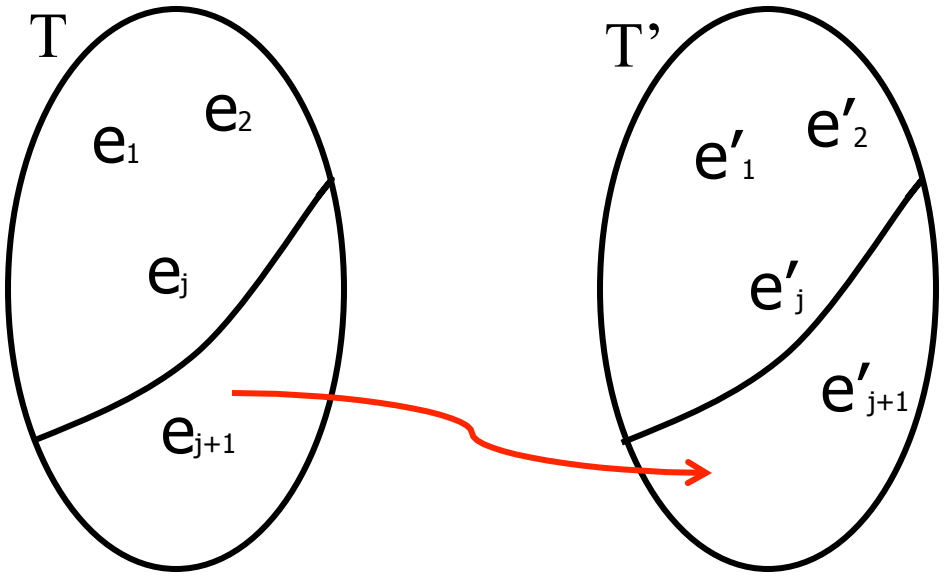
$e_1 = e'_1$

$e_2 = e'_2$

.
.
.

$e_j = e'_j$

$e_{j+1} \neq e'_{j+1}$



Algoritmo de Kruskal

- ✓ Seja j o maior índice tal que:

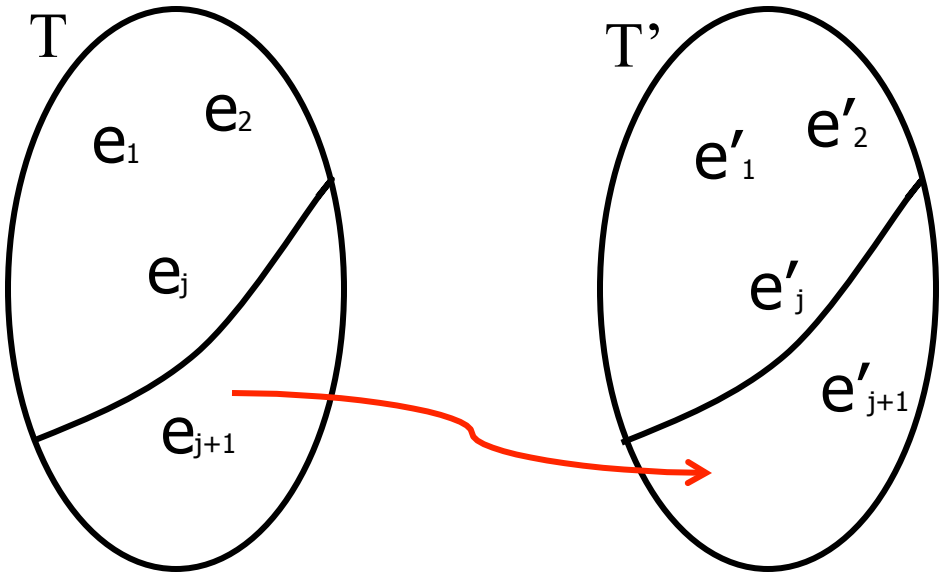
$e_1 = e'_1$

$e_2 = e'_2$

.
.
.

$e_j = e'_j$

$e_{j+1} \neq e'_{j+1}$



- ✓ O ciclo que se forma está contido necessariamente em algum e_x tal que $x > j+1$

Algoritmo de Kruskal

- ✓ Suponha que:

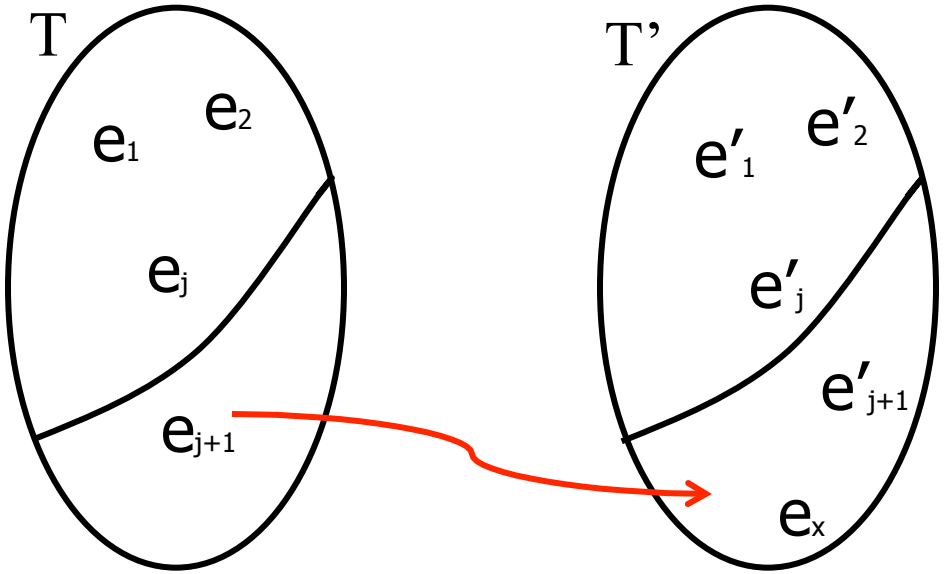
$e_1 = e'_1$

$e_2 = e'_2$

.
.
.

$e_j = e'_j$

$e_{j+1} \neq e'_{j+1}$



- ✓ O ciclo que se forma está contido necessariamente em algum e_x tal que $x > j+1$
- ✓ Além disso, $w(e_x) \geq w(e_{j+1})$

Algoritmo de Kruskal

- ✓ Suponha que:

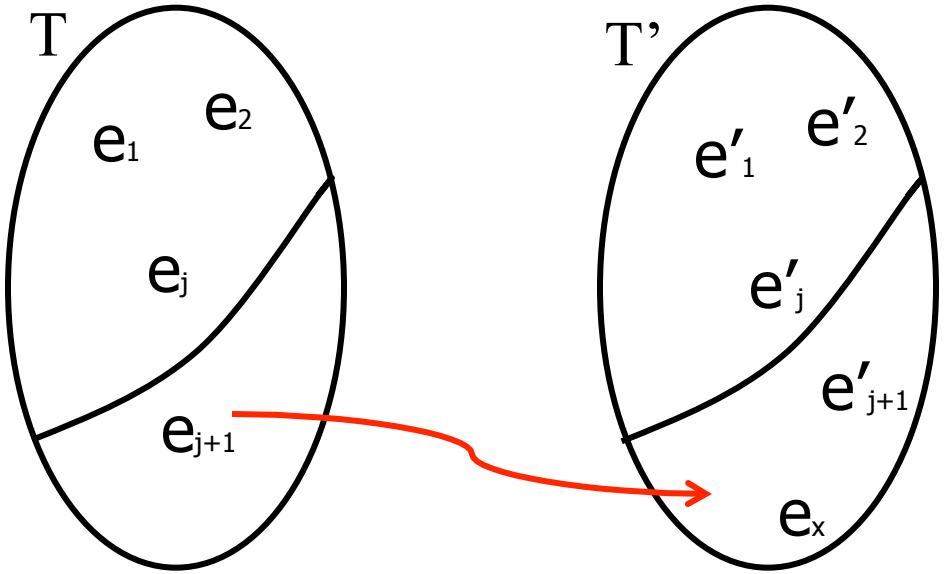
$e_1 = e'_1$

$e_2 = e'_2$

.
.
.

$e_j = e'_j$

$e_{j+1} \neq e'_{j+1}$



- ✓ O ciclo que se forma está contido necessariamente em algum e_x tal que $x > j+1$
- ✓ Além disso, $w(e_x) \geq w(e_{j+1})$
  - ✓ $w(ex) > w(ej+1)$. Absurdo!

Algoritmo de Kruskal

- ✓ Suponha que:

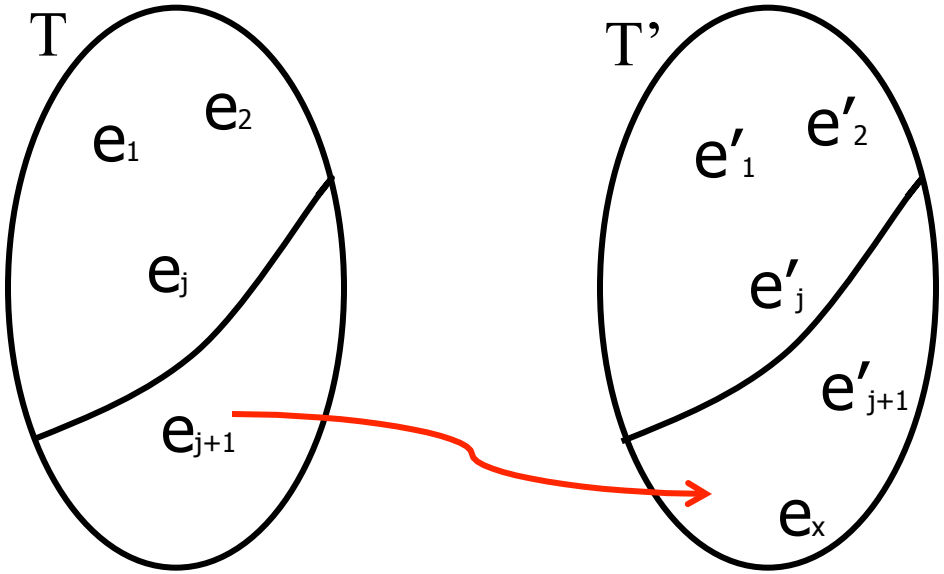
$e_1 = e'_1$

$e_2 = e'_2$

.
.
.

$e_j = e'_j$

$e_{j+1} \neq e'_{j+1}$



- ✓ O ciclo que se forma está contido necessariamente em algum e_x tal que $x > j+1$
- ✓ Além disso, $w(e_x) \geq w(e_{j+1})$
  - ✓ $w(ex) > w(ej+1)$. Absurdo!
  - ✓ $w(ex) = w(ej+1)$. Absurdo!

Algoritmo Guloso

Quando aplica a problemas de otimização combinatória sempre funciona, isto é, retorna a solução ótima?

Algoritmo Guloso

Quando aplica a problemas de otimização combinatória sempre funciona, isto é, retorna a solução ótima?

NÃO

Algoritmo Guloso – mais um exemplo

Problema da Mochila:

Algoritmo Guloso – mais um exemplo

Problema da Mochila:

Consider a seguinte situação em que é necessário
carregar uma mochila com capacidade limitada,
com um conjunto objetos de pesos e valores diferentes.

Algoritmo Guloso – mais um exemplo

Problema da Mochila:

Consider a seguinte situação em que é necessário
carregar uma mochila com capacidade limitada,
com um conjunto objetos de pesos e valores diferentes.

Objetivo: ocupar a mochila com o maior valor possível,
não ultrapassando o seu peso máximo.

Algoritmo Guloso – mais um exemplo

Problema da Mochila:

Consider a seguinte situação em que é necessário
carregar uma mochila com capacidade limitada,
com um conjunto objetos de pesos e valores diferentes.

Objetivo: ocupar a mochila com o maior valor possível,
não ultrapassando o seu peso máximo.

Solução: subconjunto de objetos cujo peso não ultrapasse
o limite da mochila e ao mesmo tempo maximizando
o seu valor total.

Algoritmo Guloso – mais um exemplo

Problema da Mochila:

Formulação Matemática:

$$(KP): \text{ maximizar } \sum v_j \, . \, x_j$$

$$\text{Sujeito a: } \sum w_j \, . \, x_j \leq c, \;\; j = 1,2, \ldots, n$$

$$x_j \in \{0,1\}$$

Algoritmo Guloso – mais um exemplo

Problema da Mochila:

Capacidade da Mochila: 50 Kg
Objetos e respectivos pesos:

	R$ 2.000	R$ 5.000	R$ 10.000
20 Kg			
20 Kg			
		50 Kg	20 gramas

Algoritmo Guloso – mais um exemplo

Problema da Mochila:

Capacidade da Mochila: 50 Kg

Objetos e respectivos pesos:

Solução Gulosa: {geladeira}

5.000,00

	R$ 2.000	R$ 5.000	R$ 10.000
20 Kg			
20 Kg			
		50 Kg	20 gramas

Algoritmo Guloso – mais um exemplo

Problema da Mochila:

Solução ótima: {TV1, TV2, Anel}

14.000,00

Capacidade da Mochila: 50 Kg

Objetos e respectivos pesos:

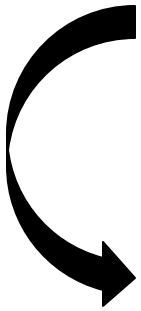
	R$ 2.000	R$ 5.000	R$ 10.000
20 Kg			
20 Kg			
		50 Kg	20 gramas

Algoritmo Guloso – mais um exemplo

Torna-se interessante saber quando utilizar o algoritmo guloso com garatia de obtenção da solução ótima

Algoritmo Guloso – mais um exemplo

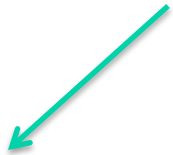
Torna-se interessante saber quando utilizar o algoritmo guloso com garatia de obtenção da solução ótima



MATRÓIDES

Matróides

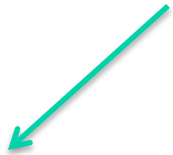
Seja E um conjunto finito e não vazio e F uma coleção não vazia de subconjuntos de E, fechado por inclusão.



$$I_1 \subseteq I_2, I_2 \in F \rightarrow I_1 \in F$$

Matróides

Seja E um conjunto finito e não vazio e F uma coleção não vazia de subconjuntos de E, fechado por inclusão.



$$I_1 \subseteq I_2, I_2 \in F \rightarrow I_1 \in F$$

(E,F) é denominado Sistema de Independência (ou de subconjuntos) e os elementos de F são chamados Independentes.

Matróides

Um sistema de subconjuntos é um Matróide quando o algoritmo Guloso resolver otimamente o problema de otimização combinatória associado

Matróides

Caracterização: Um sistema de subconjuntos M=(E,F). As seguintes afirmações são equivalentes:

(i)M é matróide

(ii) Sejam I_1 e I_2 dois independentes de F tal que $|I_1|=|I_2|+1$ então existe $e \in I_1 \setminus I_2$ tal que $I_1 \cup e \in F$

(iii) Seja $A \subseteq E$ e I_1 e I_2 independentes maximais de A, então $|I_1|=|I_2|$