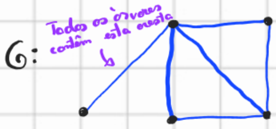
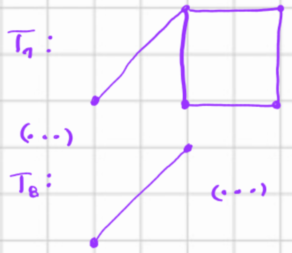
Aula 25

Árvores abrangentes

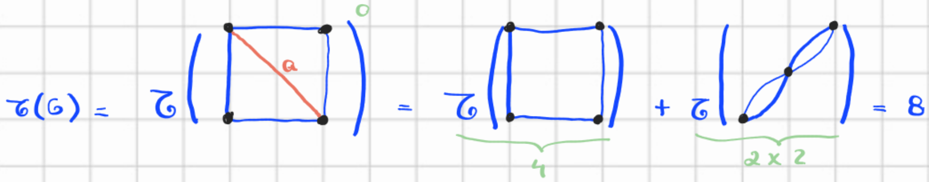
$$\underbrace{\tau(G)}_{\text{nº de árvores abranjentes de } G} = \tau(G-a) + \tau(G/\!/a)$$

ex:





$$\tau(G) = \tau(G-v_1) + \underbrace{\tau(G/\!/v_1)}_{0}$$



— // —

Fórmula de Cayley

$$\boxed{\tau(K_n) = n^{n-2}}$$ → Importante!

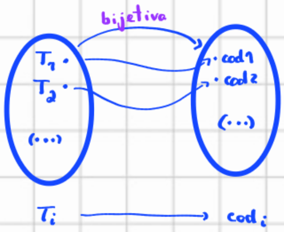
(*) ↗ Cada árvore tem um código único!

Nota: Não precisamos de saber os códigos de Prüfer (Dem e Slides)

Dem: bij $\varphi: \mathcal{F} \longrightarrow (a_1, a_2, \dots, a_{n-2})$ - cod., $a_i \in V$

conjunto de todas as árvores $T=(V,E)$

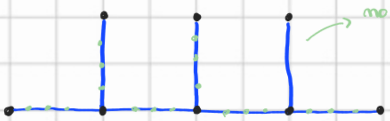
↳ código de Prüfer



• Depois existe uma maneira para descodificar!

ex:

(*)



no final ficamos com 2 vértices!

← Vai-se apagando as arestas

Para $i = 1, \dots, n-2$:

→ Eliminar sucessivamente o vértice folha de menor etiqueta i, s_i, cujo único vizinho é t_i, construindo-se a lista $(t_1, t_2, \dots, t_{n-2})$ designada por código de Prüfer

i	1	2	3	4	5	6	7	8
s_i	1	2	6	7	3	4		
t_i	4	3	5	3	4	5		

Código de Prüfer: $\underbrace{(4, 3, 5, 3, 4, 5)}_{n-2 \text{ uplo}}$

$\Rightarrow$ $\underbrace{n \times n \times \dots \times n}_{n-2} \Rightarrow \tau(K_n) = n^{n-2}$

—/ /—

Árvores abrangentes de custo (total) mínimo T (ótimos)

$G = (V, E, \psi)$ finito conexo com uma função de custo

$$w : E \longrightarrow [0, +\infty]$$

• obter $T = (V, E')$, $E' \subseteq E$

é T ótima!

Algoritmos:

→ **Kruskal:** Ordenam-se as arestas por ordem não decrescente do seu custo (custo computacional) e vão adicionando-se arestas a E' enquanto o grafo não é conexo (não existindo ciclos em T)

pode haver custos iguais!

1 Ordenar o custo de forma "crescente"

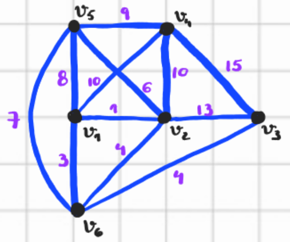
2 Adicionamos, pela ordem, a E'

(Condições de paragem)

→ **de Prim:** Inicialmente começa-se por um vértice arbitrário, e vão adicionando-se a V' (inicialmente vazio) selecionando-se as arestas e^* de menor custo que ligam $v_i^* \in V'$ com $v_j^* \in V \setminus V'$, até que $V' = V$

Aplicação: "ex."

G:



→ **Kruskal:** Input: $G = (V, E, \psi)$

Output: $T = (V, E')$ → árvore de custo mínimo

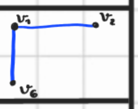
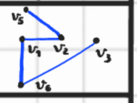
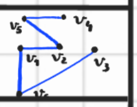
1 Ordenar arestas (por ordem do custo)

$e_1 = \underbrace{v_1 v_2}_{w(e_1) = 1}$, $e_2 = \underbrace{v_1 v_6}_{3}$, $e_3 = \underbrace{v_2 v_6}_{4}$, $e_4 = \underbrace{v_3 v_6}_{4}$, $e_5 = \underbrace{v_2 v_5}_{6}$

$e_6 = \underbrace{v_5 v_6}_{7}$, $e_7 = \underbrace{v_1 v_5}_{8}$, $e_8 = \underbrace{v_4 v_5}_{9}$, $e_9 = \underbrace{v_1 v_4}_{10}$, $e_{10} = \underbrace{v_2 v_4}_{10}$,

$e_{11} = \underbrace{v_2 v_3}_{13}$, $e_{12} = \underbrace{v_3 v_4}_{15}$

2

i	e_i	$w(e_i)$	Insere e_i ?	$T = G[E']$
1	$v_1 v_2$	1	Sim	
2	$v_1 v_6$	3	Sim	
3	$v_2 v_6$	4	Não	Contem ciclo!
4	$v_3 v_6$	4	Sim	
5	$v_2 v_5$	6	Sim	
6	$v_5 v_6$	7	Não	Contem ciclo!
7	$v_1 v_5$	8	Não	Contem ciclo!
8	$v_4 v_5$	9	Sim	
~~9~~	~~$v_1 v_4$~~	~~10~~	PARAR!	Conexo!

$\Longrightarrow$ $T = (V, E')$ conexo $|E'| = 5$

Logo: $T = (V, E')$ de custo total $w(e_1) + w(e_2) + w(e_4) + w(e_5) + w(e_8) = 23$

—//—

Outra resolução!

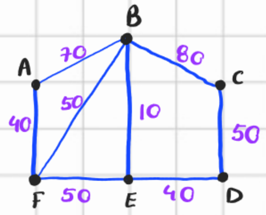
→ De Prim: notação: $v_i \equiv i$, aresta ij com custo w_{ij}, $i \in V'$, $j \in V \setminus V'$

$e^* = i^* j^*$, aresta de menor custo

Podiomos escolher outro →

V'	E'	(ij, w_{ij}), $i \in V'$, $j \in V \setminus V'$	$e^* = i^* j^*$
$\{3\}$	$\{\}$	(32, 13), (34, 15), (36, 4) ($v_3 v_2$, 13)*	36
$\{3, 6\}$	$\{36\}$	(32, 13), (34, 15), (62, 4), (61, 3), (65, 7)	61
$\{3, 6, 1\}$	$\{36, 61\}$	(32, 13), (34, 15), (62, 4), (65, 7), (12, 1), (14, 10), (15, 8)	12
$\{3, 6, 1, 2\}$	$\{36, 61, 12\}$	(34, 15), (65, 7), (14, 10), (15, 8) (24, 10), (25, 6)	25
$\{3, 6, 1, 2, 5\}$	$\{36, 61, 12, 25\}$	(34, 15), (14, 10), (24, 10), (54, 9)	54
$\{3, 6, 1, 2, 5, 4\}$	$\{36, 61, 12, 25, 54\}$	PARAR, $V' = V$	

$\Longrightarrow$ $T = (V, E')$ árvore ótima coincide com a que foi obtida pelo alg de Kruskal (custo total 23) ex:

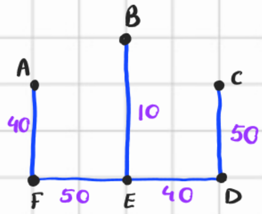


Notação: aresta ij com custo w_{ij}, $i \in V'$, $j \in V \setminus V'$

$e^* = i^*j^*$, aresta de menor custo

V'	E'	$(ij, w_{ij}), i \in V', j \in V \setminus V'$	$e^* = i^*j^*$
{A}	{ }	(AB,70), (AF,40)	AF
{A, F}	{AF}	(AB,70), (FB,50), (FE,50)	FE
{A,F,E}	{AF,FE}	(AB,70), (FB,50), (EB,10), (ED,40)	EB
{A,B,F,E}	{AF,FE, EB}	(ED,40), (BC,80)	ED
{A,B,F,E,D}	{AF,FE, EB, ED}	(BC,80), (DC,50)	DC
{A,B,F,E,D,C}	{AF,FE, EB,ED, DC}	PARAR, $V' = V$	

→ Podíamos escolher outra!



Custo total: 190 mil