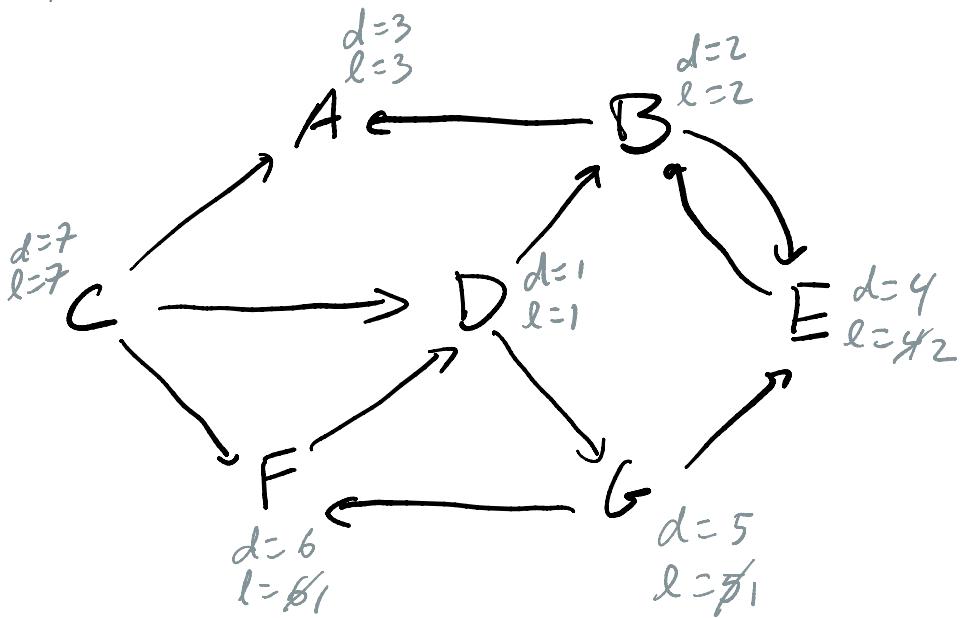


I.a)

02 September 2020 15:05

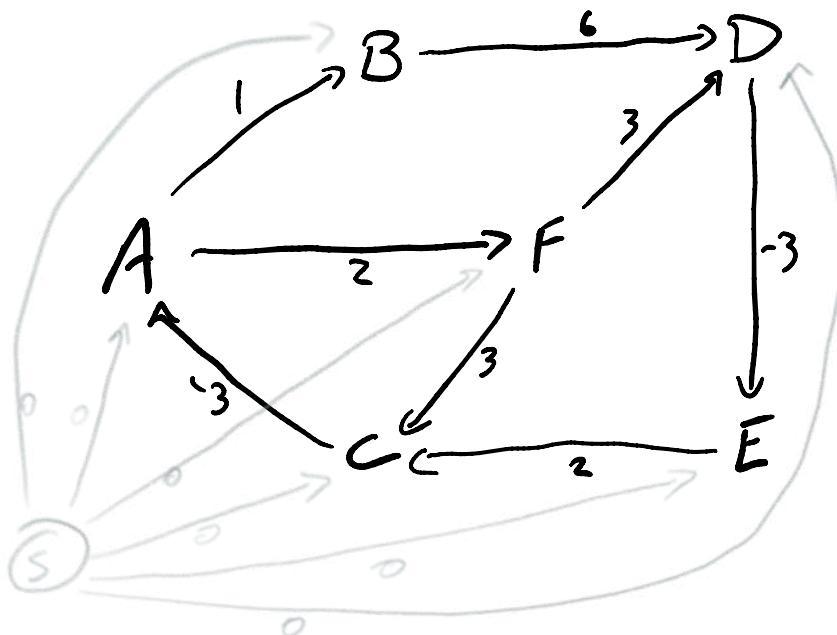


$SCCs: \{A\}, \{E, B\}, \{F, G, D\}, \{C\}$

	A	B	C	D	E	F	G
low	3	2	7	1	2	1	1

I.b)

02 September 2020 15:05

Bellman-Ford

$$h(A) = \delta(S, A) = -4$$

$$h(B) = \delta(S, B) = -3$$

$$h(C) = \delta(S, C) = -1$$

$$h(D) = \delta(S, D) = 0$$

$$h(E) = \delta(S, E) = -3$$

$$h(F) = \delta(S, F) = -2$$

$$\hat{w}(u, v) = w(u, v) + h(u) - h(v)$$

$$\hat{w}(A, B) = w(A, B) + h(A) - h(B) = 1 + (-4) - (-3) = 0$$

$$\hat{w}(A, F) = w(A, F) + h(A) - h(F) = 2 + (-4) - (-2) = 0$$

$$\hat{w}(B, D) = w(B, D) + h(B) - h(D) = 6 + (-3) - 0 = 3$$

$$\hat{w}(C, A) = w(C, A) + h(C) - h(A) = -3 + (-1) - (-4) = 0$$

$$\hat{w}(D, E) = w(D, E) + h(D) - h(E) = -3 + 0 - (-3) = 0$$

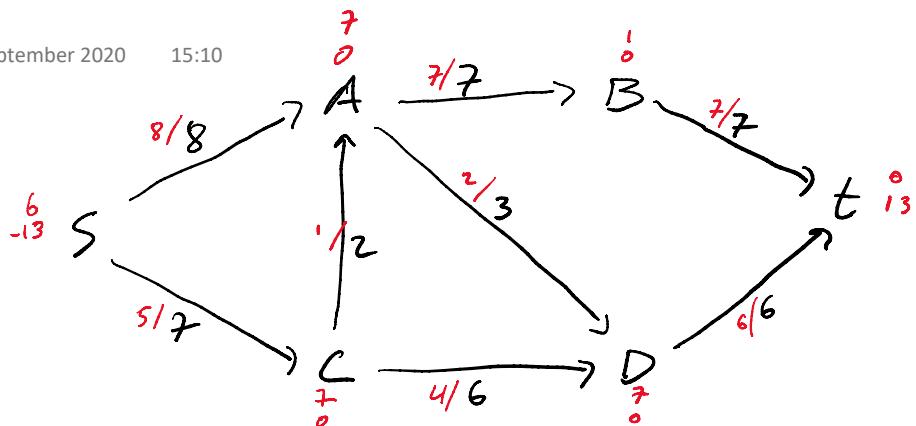
$$\hat{w}(E, C) = w(E, C) + h(E) - h(C) = 2 + (-3) - (-1) = 0$$

$$\hat{w}(F, C) = w(F, C) + h(F) - h(C) = 3 + (-2) - (-1) = 0$$

$$\hat{w}(F, D) = w(F, D) + h(F) - h(D) = 3 + (-2) - 0 = 1$$

I.C)

02 September 2020 15:10



$$N[A] = \{B, D, S, C\}$$

$$N[B] = \{t, A\}$$

$$N[C] = \{D, A, S\}$$

$$N[D] = \{t, A, C\}$$

$$L = \{C, A, B, D\}$$

	S	A	B	C	D	t	L
h	6	0	0	0	0	0	
l	-15	8	0	7	0	0	$\underline{\langle C, A, B, D \rangle}$
h	6	0	0	1	0	0	$\underline{\langle C, A, B, D \rangle}$
l	-15	9	0	0	6	0	$\underline{\langle C, A, B, D \rangle}$
h	6	1	0	1	0	0	$\underline{\langle A, C, B, D \rangle}$
l	-15	0	7	0	8	0	$\underline{\langle A, C, B, D \rangle}$
h	6	1	1	1	0	0	$\underline{\langle B, A, C, D \rangle}$
l	-15	0	0	0	8	7	$\underline{\langle B, A, C, D \rangle}$
h	6	1	1	1	2	0	$\underline{\langle D, B, A, C \rangle}$
l	-15	2	0	0	0	13	$\underline{\langle D, B, A, C \rangle}$
h	6	3	1	1	2	0	$\underline{\langle A, D, B, C \rangle}$
l	-15	0	0	1	1	13	$\underline{\langle A, D, B, C \rangle}$
h	6	3	1	1	2	0	$\underline{\langle A, D, B, C \rangle}$
l	-15	0	0	2	0	13	$\underline{\langle A, D, B, C \rangle}$
h	6	3	1	3	2	0	$\underline{\langle C, A, D, B \rangle}$
l	-15	1	0	0	1	13	$\underline{\langle C, A, D, B \rangle}$
h	6	3	1	4	2	0	$\underline{\langle C, A, D, B \rangle}$
l	-15	0	0	0	2	13	$\underline{\langle C, A, D, B \rangle}$
h	6	3	1	4	4	0	$\underline{\langle D, C, A, B \rangle}$
l	-15	2	0	0	0	13	$\underline{\langle D, C, A, B \rangle}$
h	6	5	1	4	4	0	$\underline{\langle A, D, C, B \rangle}$
l	-15	0	0	0	2	13	$\underline{\langle A, D, C, B \rangle}$
h	6	5	1	4	5	0	$\underline{\langle D, A, C, B \rangle}$
l	-15	0	0	2	0	13	$\underline{\langle D, A, C, B \rangle}$
h	6	5	1	6	5	0	$\underline{\langle C, D, A, B \rangle}$
l	-15	0	0	0	2	13	$\underline{\langle C, D, A, B \rangle}$
h	6	5	1	6	6	0	$\underline{\langle D, C, A, B \rangle}$
l	-15	2	0	0	0	13	$\underline{\langle D, C, A, B \rangle}$
h	6	7	1	6	6	0	$\underline{\langle A, D, C, B \rangle}$
l	-15	0	0	0	2	13	$\underline{\langle A, D, C, B \rangle}$
h	6	7	1	6	7	0	$\underline{\langle D, A, C, B \rangle}$
l	-15	0	0	2	0	13	$\underline{\langle D, A, C, B \rangle}$
h	6	7	1	7	7	0	$\underline{\langle C, D, A, B \rangle}$
l	-13	0	0	0	0	13	$\underline{\langle C, D, A, B \rangle}$

11c + 13

2 | -3 | 0 | 0 | 0 | 13 |

$$f(S, T) = 13$$

$L: \langle C, A, B, D \rangle, \langle A, C, B, D \rangle, \langle B, A, C, D \rangle, \langle D, B, A, C \rangle,$
 $\langle A, D, B, C \rangle, \langle C, A, D, B \rangle, \langle D, C, A, B \rangle, \langle A, D, C, B \rangle,$
 $\langle D, A, C, B \rangle, \langle C, D, A, B \rangle, \langle D, C, A, B \rangle, \langle A, D, C, B \rangle,$
 $\langle D, A, C, B \rangle, \langle C, D, A, B \rangle$

corte: $\{S, A, C, D\} / \{B, t\}$

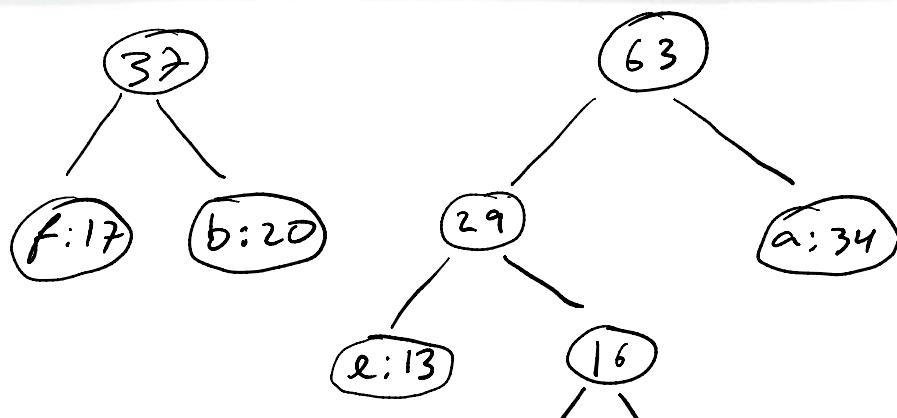
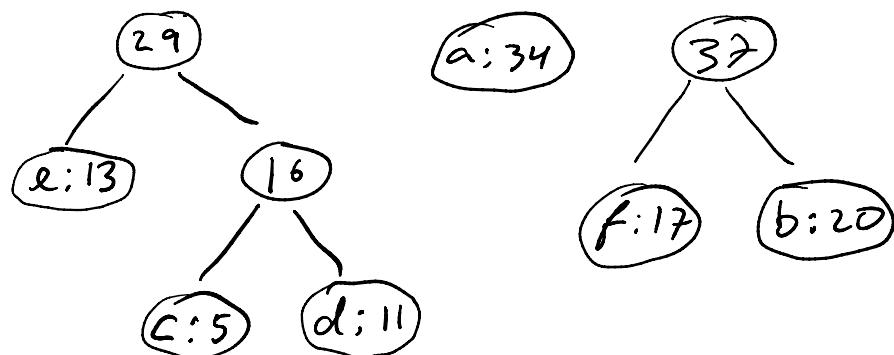
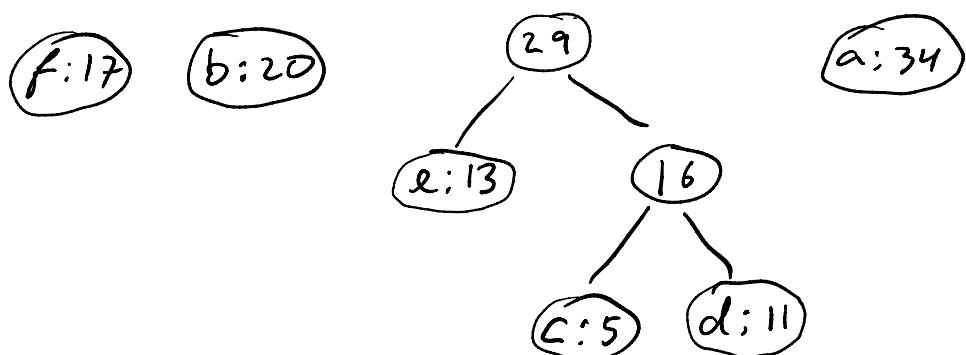
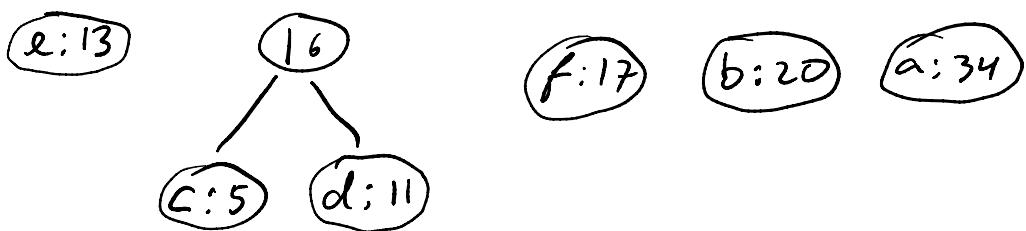
ou $\{S, A, B, C, D\} / \{t\}$

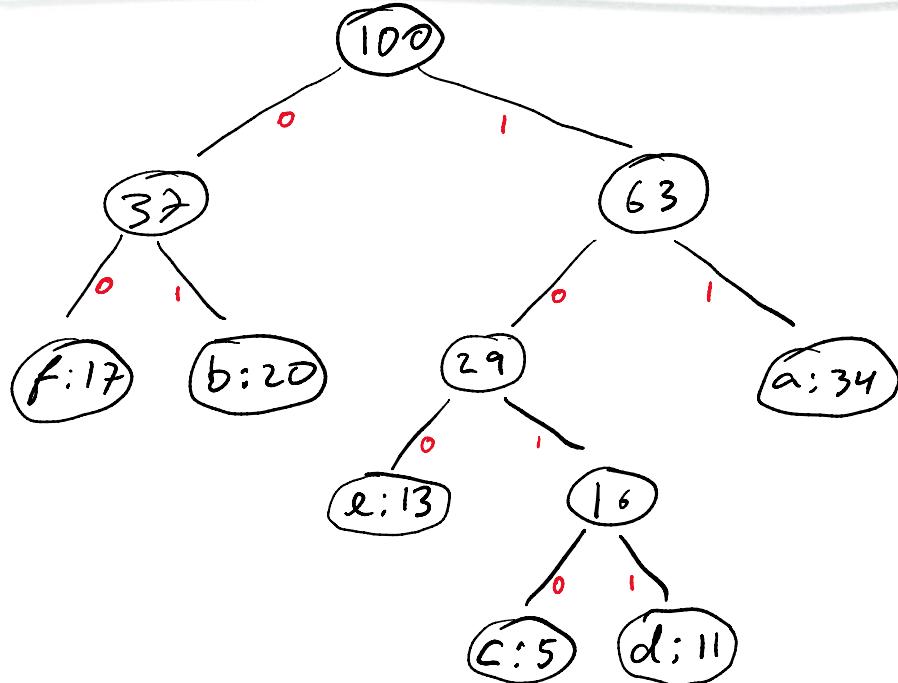
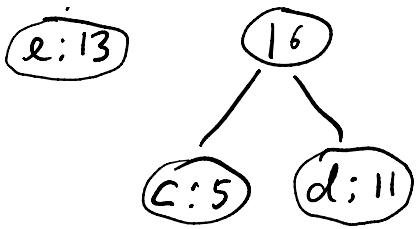
	S	A	B	C	D	t
h	6	7	1	7	7	0

I.d)

02 September 2020 15:10

$f()$	a 34	b 20	c 5	d 11	e 13	f 17
-------	---------	---------	--------	---------	---------	---------





	a	b	c	d	e	f
codificação	11	01	1010	1011	100	00

Ficheiro com 100 caracteres

$$\begin{aligned} \# \text{bits} &= 34 \times 2 + 20 \times 2 + 5 \times 4 + 11 \times 4 + 13 \times 3 + 17 \times 2 \\ &= 245 \text{ bits} \end{aligned}$$

Ficheiro com 400 caracteres

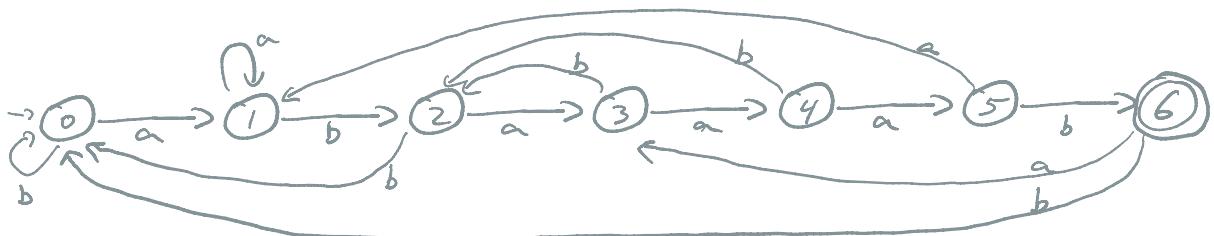
$$\# \text{bits} = 245 \times 4$$

$$= 980 \text{ bits}$$

1.e)

02 September 2020 15:10

$P = abaaabb$



$\delta(0, b)$	$\delta(1, a)$	$\delta(2, b)$	$\delta(3, b)$	$\delta(4, b)$	$\delta(5, a)$	$\delta(6, a)$	$\delta(6, b)$
0	1	0	2	2	1	3	0

T	-	a	a	b	a	a	a	b	a	a	a	b	a	a	b	a
i	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
q	0	1	1	2	3	4	5	6	3	4	5	6	3	4	2	3

#Total matches = 2

1.f)

02 September 2020 15:10

$$A_1(3 \times 2) \underset{k=1}{\times} A_2(2 \times 5) \underset{k=2}{\times} A_3(5 \times 1) \underset{k=3}{\times} A_4(1 \times 4)$$

m^i	1	2	3	4
1	0	30	16	28
2	0	0	10	18
3	0	0	0	20
4	0	0	0	0

s^i	1	2	3	4
1	M	1	1	3
2	M	M	2	3
3	M	M	M	3
4	M	M	M	M

Parenteses:

$$(A_1 \times (A_2 \times A_3)) \times A_4$$

$$m[1,3] = 16$$

$$m[1,4] = 28$$

$$m[2,4] = 18$$

$$m[1,2] = m[1,1] + m[2,2] + 3 \times 2 \times 5$$

$$= 0 + 0 + 30 = 30$$

$$m[2,3] = m[2,2] + m[3,3] + 2 \times 5 \times 1$$

$$= 0 + 0 + 10 = 10$$

$$m[3,4] = m[3,3] + m[3,4] + 5 \times 1 \times 4$$

$$= 0 + 0 + 20 = 20$$

$$m[1,3] = \min \{ m[1,1] + m[2,3] + 3 \times 2 \times 1, \\ m[1,2] + m[3,3] + 3 \times 5 \times 1 \}$$

$$= \min \{ 0 + 10 + 6, \\ 30 + 0 + 15 \} = 16 \quad k=1$$

$$m[2,4] = \min \{ m[2,2] + m[3,4] + 2 \times 5 \times 4, \\ m[2,3] + m[4,4] + 2 \times 1 \times 4 \}$$

$$= \min \{ 0 + 20 + 40, \\ 10 + 0 + 8 \} = 18 \quad k=2$$

$$m[1,4] = \min \{ m[1,1] + m[2,4] + 3 \times 2 \times 4, \\ m[1,2] + m[3,4] + 3 \times 5 \times 4, \\ m[1,3] + m[4,4] + 3 \times 1 \times 4 \}$$

$$= \min \{ 0 + 8 + 24, \\ 30 + 20 + 60, \\ 16 + 0 + 12 \} = 28 \quad k=3$$

1. Insert: $T(n) = 1 + T(n-1)$

$$= 1 + 1 + T(n-2)$$

$$= \underbrace{1 + 1 + \dots + 1}_{n} + T(1)$$

$$= \sum_{i=1}^n O(1)$$

$$= O(n)$$

Logo, a função Insert $\in O(n)$.

Remove : $T(n) = \sum_{i=1}^n O(1)$

$$= O(n)$$

Logo, a função Remove $\in O(n)$.

Extract Min : $T(n) = O(1)$

Logo, a função Extract Min $\in O(1)$.

Decrease Key : $T(n) = O(n) + O(1) + O(n) + O(1)$

$$= O(n)$$

Logo, a função Decrease Key $\in O(n)$.

2.

O Algoritmo de Dijkstra quando implementado com um amontoado tem complexidade $O((V+E) \log V)$

- Cada vértice é extraído de Q : $O(V)$

- e pode implicar uma atualização de Q : $O(\log V)$

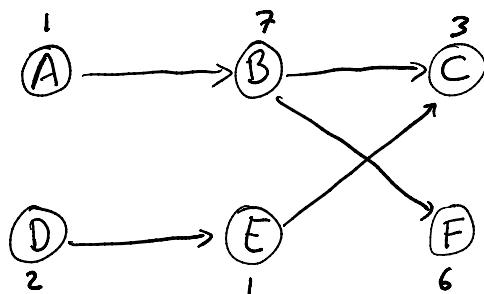
- A operação de relaxamento é aplicada 1x a cada arco : $O(E)$
- e pode implicar uma atualização de Q : $O(\log V)$

Tudo não me necessitava
caber em uma expressão
adicional

Nesta implementação naïf, o Algoritmo de Dijkstra vai fazer $|E|$ operações de relaxamento, onde cada operação

Vai fazer $|E|$ operações de relaxação, onde cada operação faz uma chamada à função DecreaseKey, que tem complexidade $O(n)$, onde $n = |V|$.

Logo, esta implementação não terá complexidade $O(E \cdot n)$.



	A	B	C	D	E	F
Precos						
custos						

1. Num DAG, depois de aplicar o DFS,
 Se existe caminho de $u \rightarrow v$, então $f[u] > f[v]$
 O que indica uma ordem topológica

- Inicializar o vetor de custos : Para cada $u \in V$, $c_u = p_u$) $O(V)$
- Iniciar uma DFS a começar em qualquer vértice) $O(V+E)$
- Sempre que um vértice é fechado, inserir numa pilha
 Com isto determinaremos uma ordem topológica inversa
- Iniciar uma segunda DFS, escalhando os
 Vértices pelo maior tempo de final, i.e., segundo
 a ordem da pilha (topologia inversa)
- Na fase de backtrace de uma aresta (u,v) , actualizar
 $c_u = c_v$ se $c_v < c_u$) $O(V+E)$

Complexidade : $O(V+E)$

2. Agora o algoritmo terá que funcionar num grafo contendo ciclos.
- Começar por aplicar o algoritmo de Tarjan para calcular
 os SCCs, criando um grafo acíclico, o grafo dos SCCs (G^{SCCs})) $O(V+E)$
 - Aplicar o algoritmo proposto na aula anterior a G^{SCCs}
 para obter o custo de cada SCC) $O(V+E)$
 - Depois atribuir a cada vértice do grafo original o
 custo do SCC correspondente) $O(V)$

• Depois atribuir a cada vértice do grafo original o custo do SCC correspondente $\rightarrow O(V)$

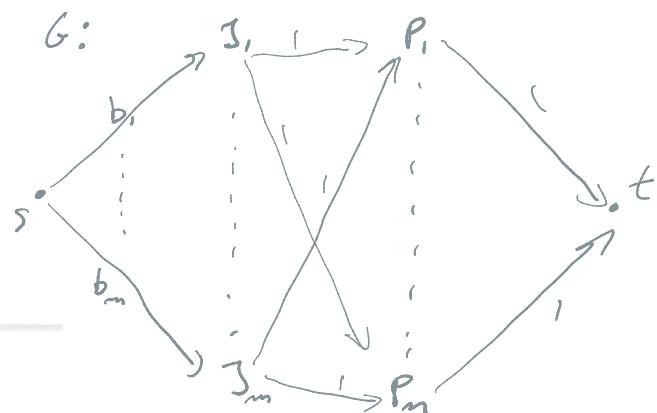
Complexidade: $O(V+E)$

m jogos $\{J_1, \dots, J_m\}$

b_i - # bilhetes disponíveis jogo J_i

n pessoas $\{P_1, \dots, P_m\}$

Cada Pessoa P_k pode pedir 1 bilhete por jogo



1. construímos uma rede de fluxos G , com:

$$V = \{J_i \mid 1 \leq i \leq m\} \quad (\text{1 vértice por jogo})$$

$$\cup \{P_k \mid 1 \leq k \leq n\} \quad (\text{1 vértice por pessoa})$$

$$\cup \{s, t\} \quad (\text{source e sink, respectivamente})$$

$$E = \{(s, J_i, b_i) \mid 1 \leq i \leq m\} \quad (\text{cada Jogo } J_i \text{ tem } \rightarrow \text{nó } b_i \text{ bilhetes})$$

$$\cup \{(J_i, P_k, 1)\} \quad (\text{cada } P_k \text{ tem no máximo 1 bilhete por jogo } J_i)$$

$$\cup \{(P_k, t, 1) \mid 1 \leq k \leq n\} \quad (\text{cada } P_k \text{ tem no máximo 1 bilhete})$$

A atribuição de bilhetes da pessoa P_k ao jogo J_i é tal que:

$$f^*(J_i, P_k) = 1$$

2.

$$|V| = m + n + 2 \in O(m+n)$$

Complexidade:

$$|E| = m + mn + m \in O(m \cdot n)$$

$$FF : O(|f^*| \cdot |E|)$$

$$|f^*| = mn \in O(m \cdot n)$$

$$= O((m \cdot n)^2)$$

$$EK : O(V \cdot E^2)$$

$$= O((m+n)(m \cdot n)^2)$$

$$RZF : O(V^3)$$

$$= O((m \cdot n)^3)$$

Logo, a melhor solução consiste em considerar o método FF.

1.

$$\text{Max } x_1 + x_2 + x_3 + x_4$$

S.a

$x_3 \geq 400$	(Produção de F3)
$2x_1 + 3x_2 + 4x_3 + 5x_4 \leq 3300$	(Mão de obra)
$3x_1 + 4x_2 + 5x_3 + 6x_4 \leq 4000$	(Materiais)
$15x_1 + 10x_2 + 9x_3 + 7x_4 \leq 12.000$	(Polimento)
$7x_4 \leq 500$	(Polimento de F4)
$x_1, x_2, x_3, x_4 \geq 0$	

2. Forma standard

$$\text{Max } x_1 + x_2 + x_3 + x_4$$

S.a

$-x_3 \leq -400$	
$2x_1 + 3x_2 + 4x_3 + 5x_4 \leq 3300$	
$3x_1 + 4x_2 + 5x_3 + 6x_4 \leq 4000$	
$15x_1 + 10x_2 + 9x_3 + 7x_4 \leq 12.000$	
$7x_4 \leq 500$	
$x_1, x_2, x_3, x_4 \geq 0$	

Esto no es necesario.
Só para ser óptimo que exista un $b_i < 0$

Forma slack

$$Z = 0 + x_1 + x_2 + x_3 + x_4$$

$$x_5 = -400 \quad +x_3$$

$$x_6 = 3300 - 2x_1 - 3x_2 - 4x_3 - 5x_4$$

$$x_7 = 4000 - 3x_1 - 4x_2 - 5x_3 - 6x_4$$

$$x_8 = 12000 - 15x_1 - 10x_2 - 9x_3 - 7x_4$$

$$x_9 = 500 \quad +7x_4$$

Solución Básica Inicial

$$x_5 = -400, x_6 = 3300, x_7 = 4000, x_8 = 12000, x_9 = 500$$

No es óptima dado que tiene solamente una variable < 0 , derivado de tener un $b_i < 0$ en forma standard.

Programa Lineal Auxiliar

$$\text{Max } -x_0$$

S.a

$$-x_3 \quad -x_0 \leq -400$$

MAX - ~0

S.a

$$\begin{aligned} -x_3 - x_0 &\leq -400 \\ 2x_1 + 3x_2 + 4x_3 + 5x_4 - x_0 &\leq 3300 \\ 3x_1 + 4x_2 + 5x_3 + 6x_4 - x_0 &\leq 4000 \\ 15x_1 + 10x_2 + 9x_3 + 7x_4 - x_0 &\leq 12000 \\ 7x_4 - x_0 &\leq 500 \\ x_1, x_2, x_3, x_4, x_0 &\geq 0 \end{aligned}$$

3. Programa linear dual

$$\text{Min } -400y_1 + 3300y_2 + 4000y_3 + 12000y_4 + 500y_5$$

$$\begin{aligned} 2y_2 + 3y_3 + 15y_4 &\geq 1 \\ 3y_2 + 4y_3 + 10y_4 &\geq 1 \\ -y_1 + 4y_2 + 5y_3 + 9y_4 &\geq 1 \\ 5y_2 + 6y_3 + 7y_4 + 7y_5 &\geq 1 \\ y_1, y_2, y_3, y_4, y_5 &\geq 0 \end{aligned}$$

4. O # mínimo de operações do Algoritmo simplex

é dado por C_m^{m+n} onde

- m é o # variáveis

- n é o # restrições

Assim, o Programa Linear Simplex executa no

$$\text{máximo } C_4^9 = \frac{9!}{4!5!}$$

O Programa Linear Dual executa no

$$\text{máximo } C_5^9 = \frac{9!}{5!4!}$$

Logo, o número mínimo de operações de pivotagem é semelhante nos dois casos.

1. $\text{Sub}(i) = \begin{cases} S[i] & , \text{ se } i=1 \\ \max\{S[i], S[i]+\text{sub}(i-1)\} & , \text{ caso contrario} \end{cases}$

2. $\text{MaxSubSeq}(S[1..n])$

let $\text{sub}[0..n]$ be a vector of size n

$\text{sub}[0] = 0$

for $i = 1$ to n do

```

 $\text{sub}[i] = S[i]$ 
if  $S[i] + \text{sub}[i-1] > \text{sub}[i]$ 
   $\text{sub}[i] += \text{sub}[i-1]$ 
end if

```

```

if  $S[i] + \text{sub}[i-1] > S[i]$ 
   $\text{sub}[i] = S[i] + \text{sub}[i-1]$ 
else
   $\text{sub}[i] = S[i]$ 
end if

```

end for

return sub

Complexidade $O(n)$

3. O valor máximo de subseqüências contíguas de S é dado pelo valor max no vetor sub .

$\max\{ \text{sub}(i) \mid 1 \leq i \leq n \}$