

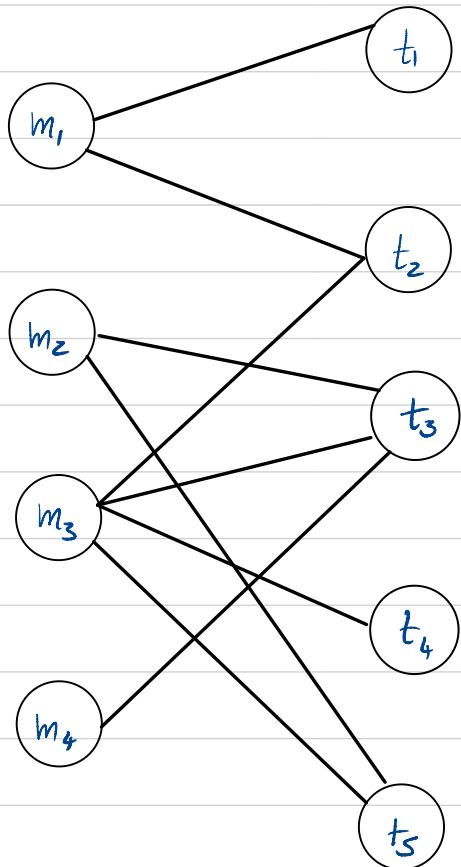
• Sumário

- Fluxo Máximo: Aplicações
 - Correspondência Bipartida Máxima
 - Atribuição de Bilhetes
 - Escape Problem
- Algoritmos de Push-Relabel
 - Introdução
 - Exemplos

Aula 11



Problema da Correspondência Bipartida Máxima



Problema da Correspondência Bipartida Máxima

Definição [Bi-partição]

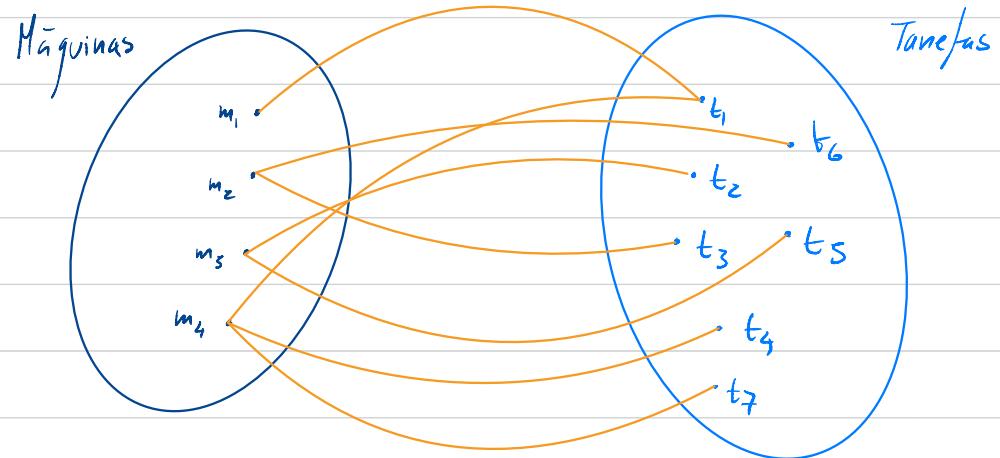
Dado um grafo $G = (V, E)$, uma bi-partição de $G = (V, E)$ é um par (L, R) tal que: $L \cap R = V$ e $L \cup R = \emptyset$.

Definição [Correspondência Bipartida]

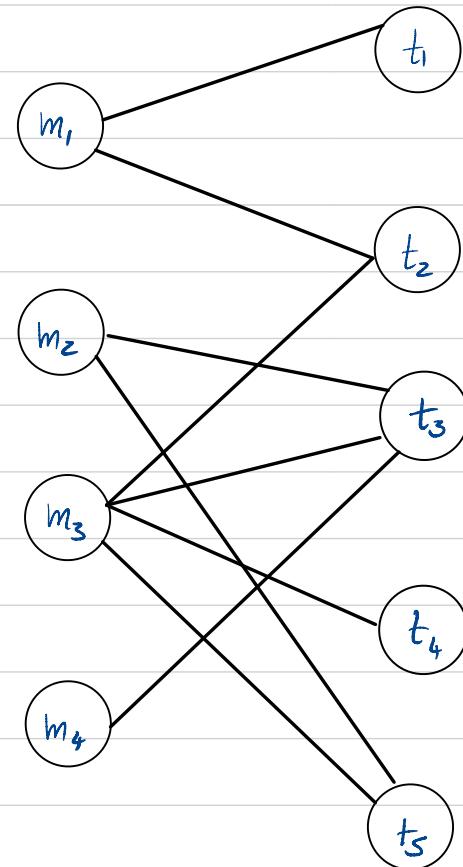
Dado um grafo $G = (V, E)$ e uma bi-partição (L, R) de G , $M \subseteq E$ diz-se uma correspondência bipartida de G se todos os vértices de V têm apenas um arco incidente em M .

Definição [Problema da Correspondência Bipartida Máxima]

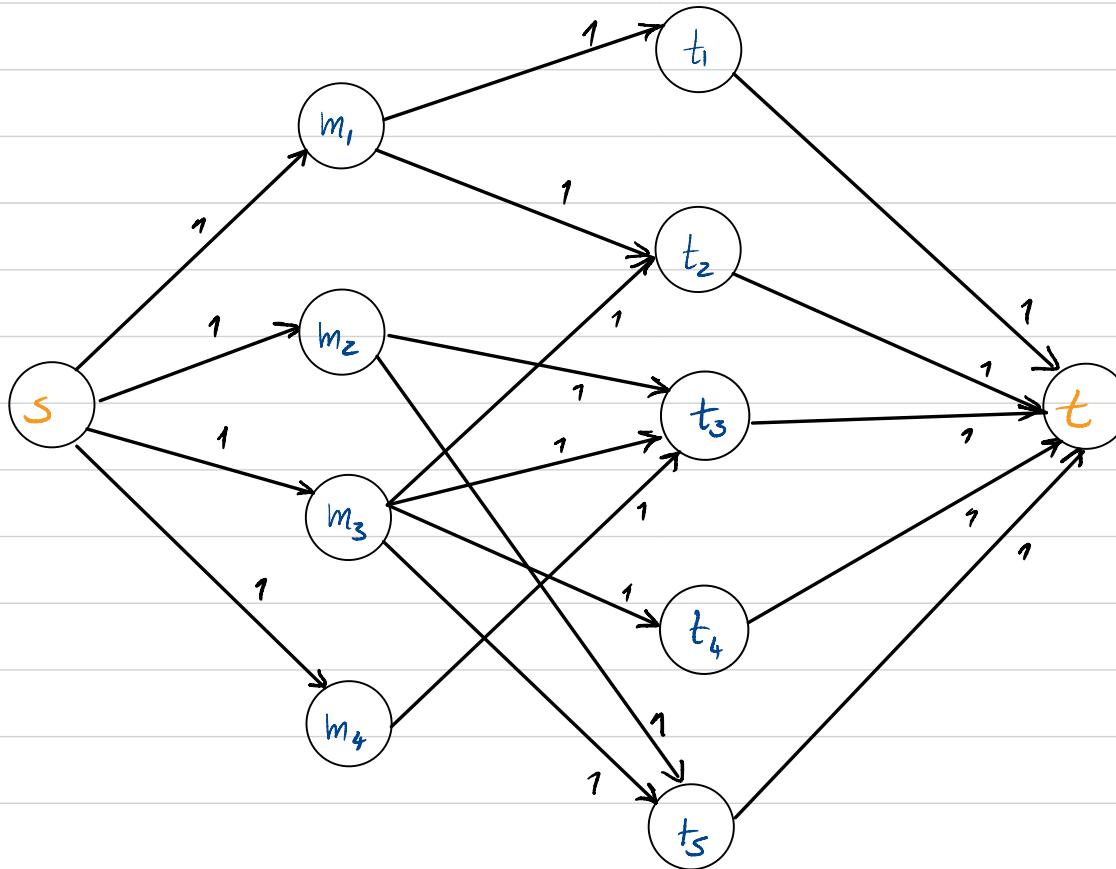
Dado um grafo $G = (V, E)$ e uma bi-partição (L, R) de G , o problema da correspondência bipartida máxima consiste em determinar a correspondência bipartida M com maior cardinalidade.



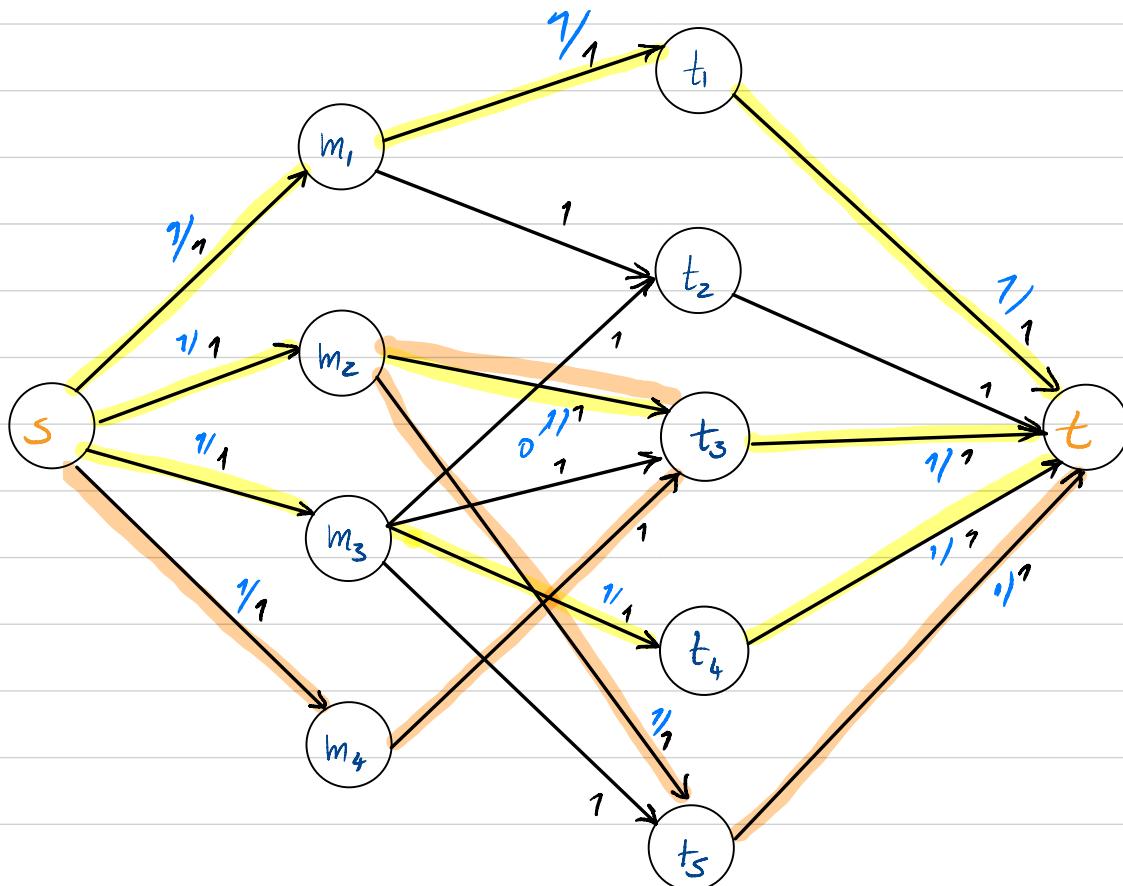
Problema da Correspondência Bipartida Máxima



Problema da Correspondência Bipartida Máxima



Problema da Correspondência Bipartida Máxima



Correspondência

- $m_1 \mapsto t_1$
- $m_2 \mapsto t_5$
- $m_3 \mapsto t_4$
- $m_4 \mapsto t_3$

$$|f^*| = 4$$

Conexão Bipartida Máxima

- Conexão Bipartida via Rede de Fluxo

$$G = (V, E), (L, R)$$



$$G' = (V', E', \delta, t, c)$$

$$\cdot V' = V \cup \{s, t\}$$

$$\cdot E' = E \cup \{(s, v) \mid v \in L\} \cup$$

$$\{(v, t) \mid v \in R\}$$

$$\cdot c(u, v) = \begin{cases} 1 & \text{se } (u, v) \in E' \\ 0 & \text{c.c.} \end{cases}$$

- Fluxo na rede \Rightarrow Conexão Bipartida
(Como calcular a conexão bipartida máxima dado o fluxo)

Concordância Bipartida Máxima

- Concordância Bipartida via Rede de Fluxo

$$G = (V, E), (L, R)$$



$$G' = (V', E', \delta, t, c)$$

$$\cdot V' = V \cup \{s, t\}$$

$$\cdot E' = E \cup \{(s, v) \mid v \in L\} \cup$$

$$\{(v, t) \mid v \in R\}$$

$$\cdot c(u, v) = \begin{cases} 1 & \text{se } (u, v) \in E' \\ 0 & \text{c.c.} \end{cases}$$

- Fluxo na rede \Rightarrow Concordância
(como calcular a concordância máxima dado o fluxo)

$$M_f = \left\{ (u, v) \mid u \in L \wedge v \in R \wedge f(u, v) = 1 \right\}$$

(TPC: provar que M é uma concordância)

Concessão Bipartida Máxima

- Concessão Bipartida via Rede de Fluxo

$$G = (V, E), (L, R)$$



$$G' = (V', E', \alpha, t, c)$$

$$\cdot V' = V \cup \{s, t\}$$

$$\cdot E' = E \cup \{(s, v) \mid v \in L\} \cup$$

$$\{(v, t) \mid v \in R\}$$

$$\cdot c(u, v) = \begin{cases} 1 & \text{se } (u, v) \in E' \\ 0 & \text{c.c.} \end{cases}$$

- Concessão bipartida \Rightarrow Fluxo na rede
(Como calcular o fluxo dado uma concessão)

Conexão Bipartida Máxima

- Conexão Bipartida via Rede de Fluxo

$$G = (V, E), (L, R)$$



$$G' = (V', E', \delta, t, c)$$

$$\cdot V' = V \cup \{s, t\}$$

$$\cdot E' = E \cup \{(s, v) \mid v \in L\} \cup \{(v, t) \mid v \in R\}$$

$$\cdot c(u, v) = \begin{cases} 1 & \text{se } (u, v) \in E' \\ 0 & \text{c.c.} \end{cases}$$

- Conexão Bipartida \Rightarrow Fluxo na rede
(Como calcular o fluxo dado uma conexão bipartida)

$$f_M(u, v) = \begin{cases} 1 & \text{se } (u, v) \in M \\ 1 & \text{se } u=s \text{ e } \exists w. (v, w) \in M \\ 1 & \text{se } v=t \text{ e } \exists u. (u, v) \in M \\ 0 & \text{c.c.} \end{cases}$$

(TPC: mostrar que f_M é um fluxo)

Correspondência Bipartida Máxima

- Correspondência Bipartida via Rede de Fluxo

$$G = (V, E), (L, R)$$



$$G' = (V', E', \delta, t, c)$$

$$\cdot V' = V \cup \{s, t\}$$

$$\cdot E' = E \cup \{(s, v) \mid v \in L\} \cup \{(v, t) \mid v \in R\}$$

$$\cdot c(u, v) = \begin{cases} 1 & \text{se } (u, v) \in E' \\ 0 & \text{c.c.} \end{cases}$$

• Complexidade:

- Upper Bound de Ford-Fulkerson
 $O(|E| \cdot |M|)$

- Upper Bound de Edmonds-Karp
 $O((|M| + |T|)|E|^2)$

Correspondência Bipartida Máxima

- Torneio:

- Jogos: $1 \leq j \leq m$

- Pessoas: $1 \leq i \leq n$

- Cada jogo tem um número máximo de bilhetes disponíveis $\max(j_i)$

- Cada pessoa pode assistir a único jogo (no máximo)

- $wishes(p_i) \Rightarrow$ Jogo j p_i gostaria de assistir

Problema: Determinar o melhor menor no entre jogos e pessoas já permite ao maior número possível de pessoas assistir a jogos.

Problema [Teste 1 - 2008/2009]

- Torneio:

- Jogos: $\{ J_i \mid 1 \leq i \leq m \}$

- Pessoas: $\{ P_i \mid 1 \leq i \leq n \}$

- Cada jogo tem um número máximo de bilhetes disponíveis $\max(J_i)$

- Cada pessoa pode assistir a único jogo (no máximo)

- $wishes(P_i) \Rightarrow$ Jogos J_j P_i gostaria de assistir

- Complejidade

- Upper Bound Ford Fulkerson

$$O(n \cdot W)$$

onde $W = \sum_{i=1}^n |wishes(P_i)|$

- Upper Bound Edmonds Karp

$$O(nW^2)$$

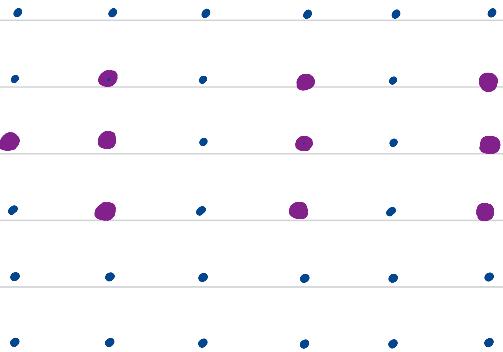
Problema: Determinar o melhor menor entre jogos e pessoas já permite ao maior número possível de pessoas assistir a jogos.

$$G = (V, E, s, t, c)$$

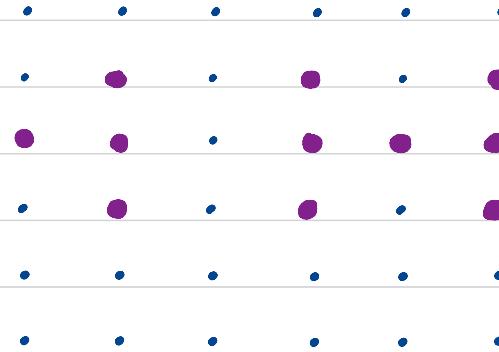
$$V = \{ J_i \mid 1 \leq i \leq m \} \\ \cup \{ P_i \mid 1 \leq i \leq n \} \cup \{ s, t \}$$

$$E = \{ (P_i, J_k) \mid J_k \in wishes(P_i) \} \\ \cup \{ (s, P_i) \mid 1 \leq i \leq n \} \\ \cup \{ (J_i, t) \mid 1 \leq i \leq m \}$$

Problema da Fuga



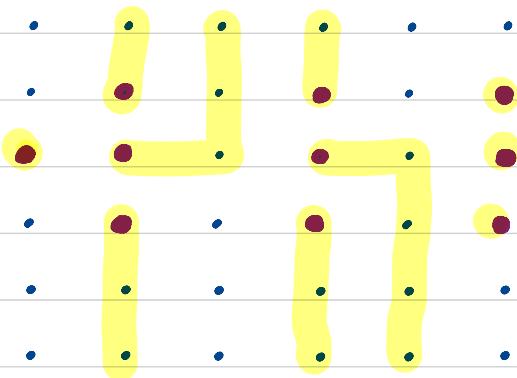
Grelha 1



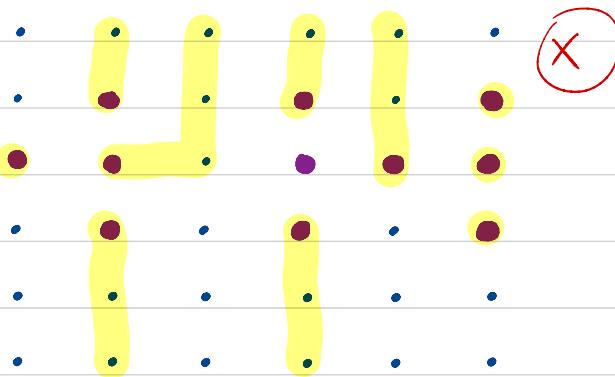
Grelha 2

Objectivo: Dada uma grelha quadrada ($n \times n$) com m pontos notáveis determinar se existem m caminhos disjuntos que ligam esses pontos à fronteira da grelha.

Problema da Fuga



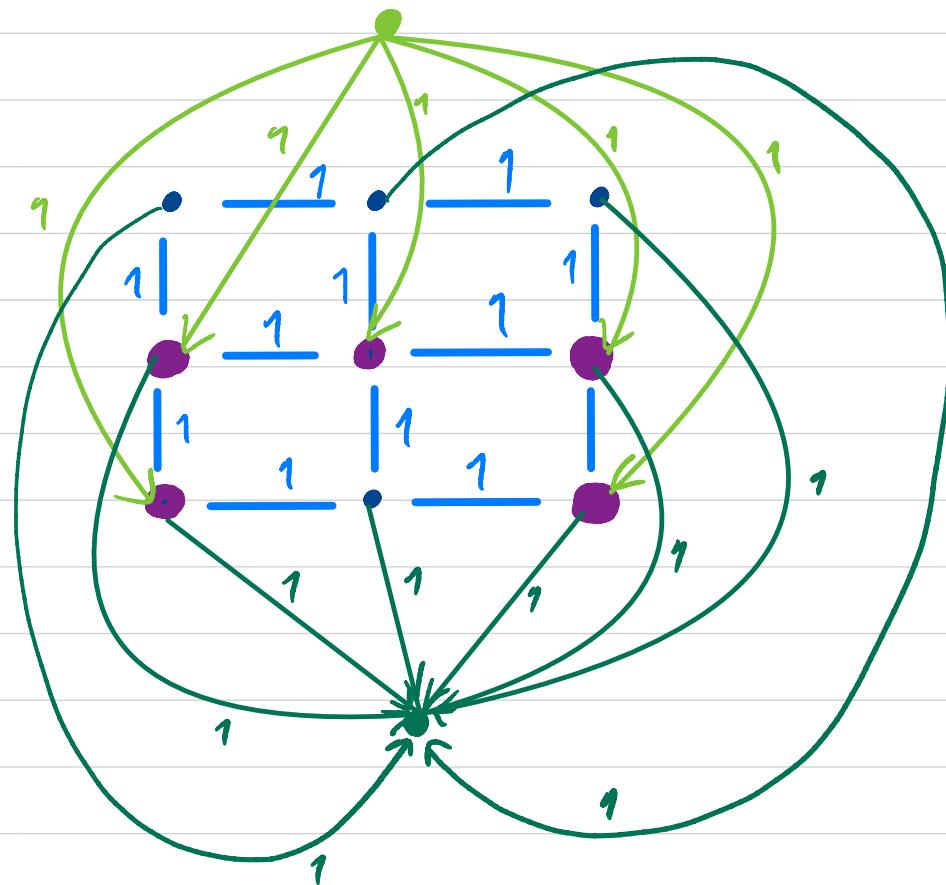
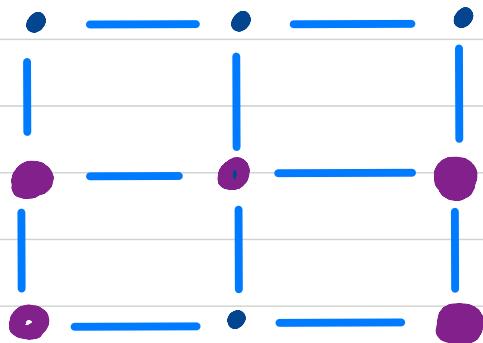
Gaelha 1



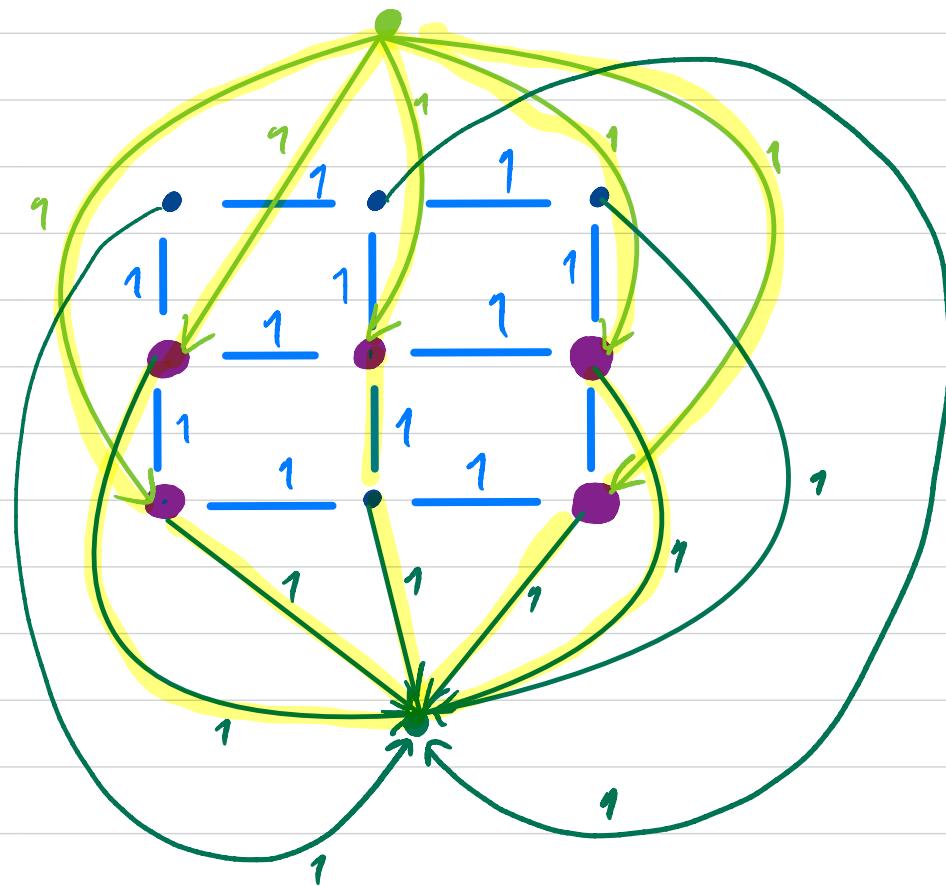
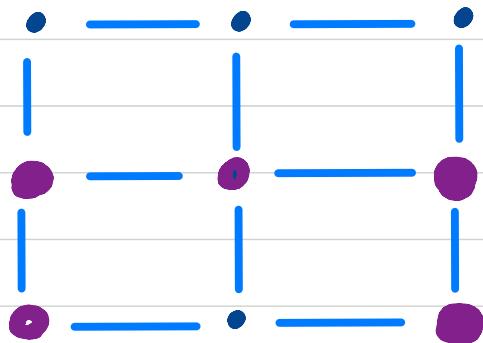
Gaelha 2

Objectivo: Dada uma grelha quadrada ($n \times n$) com m pontos notáveis determinar se existem m caminhos **disjuntos** que ligam esses pontos à fronteira da grelha.

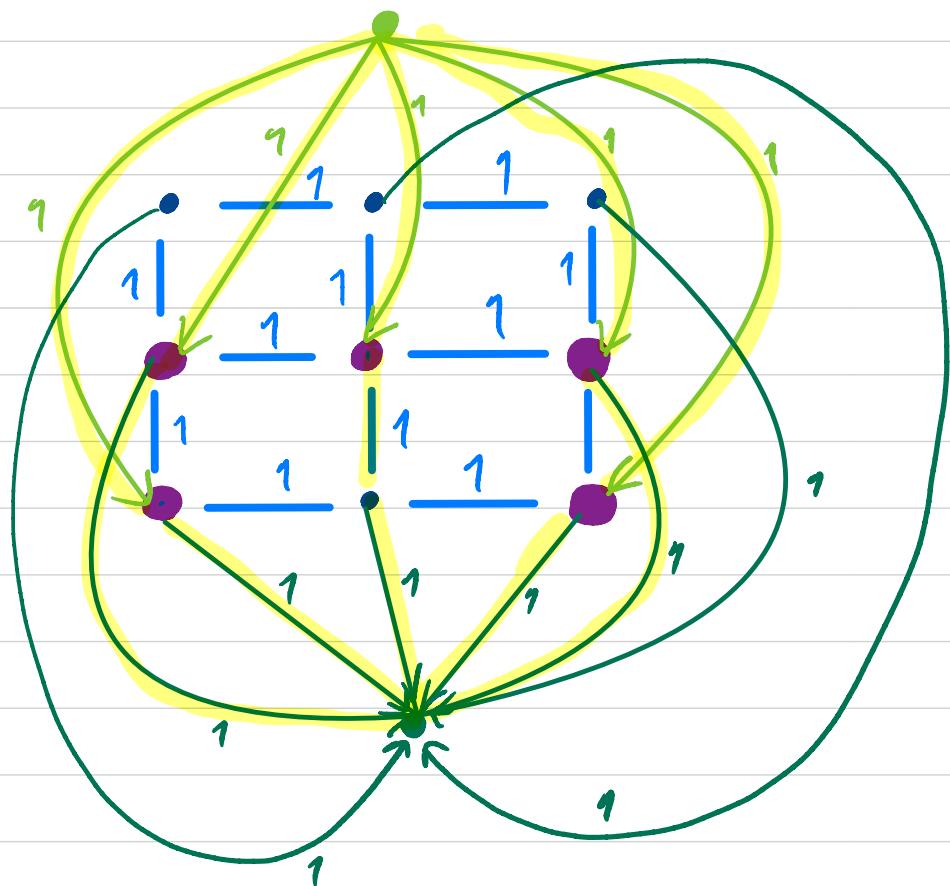
Problema da Fuga



Problema da Fuga

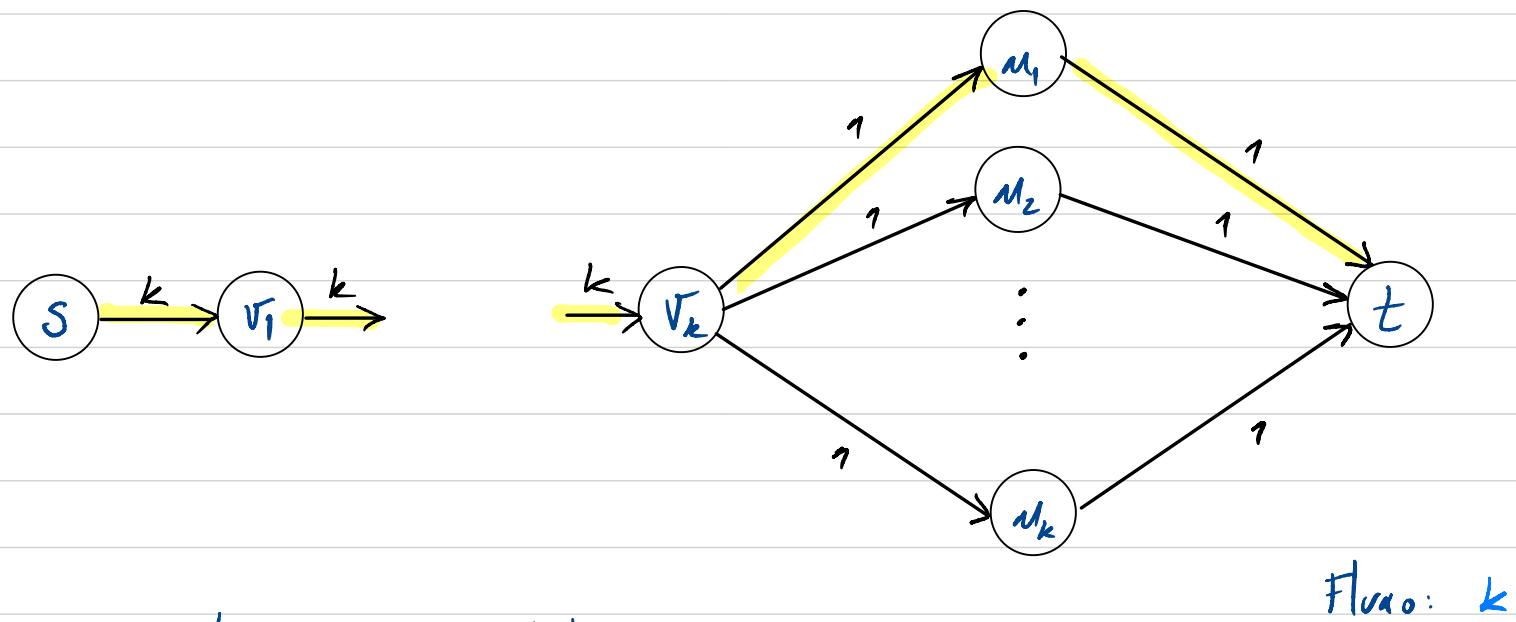


Problema da Fuga



- Upper Bound de Ford-Fulkerson
 $O(m \cdot n^2)$
- Upper Bound de Edmonds-Karp
 $O((n^2)^2 \cdot n^2) = \underline{\underline{O(n^6)}}$

Limitações de Abordagens Baseadas em Caminhos de Aumento



Fluxo: k

- Análise de complexidade

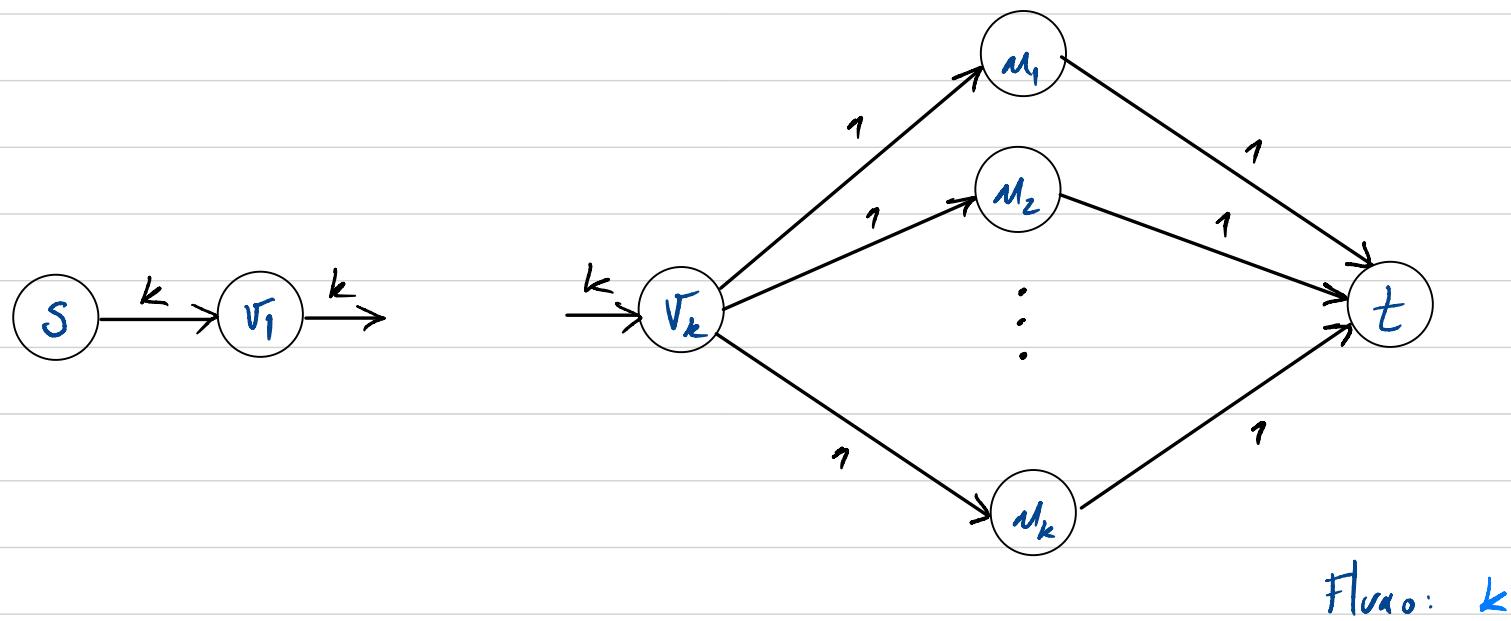
- Comprimento do caminho de aumento: $k+2$

- N° total de caminhos de aumento: k

- Capacidade de cada caminho de aumento: 1

$O(k^2)$

Limitações de Abordagens Baseadas em Caminhos de Aumento



Ideia do Método de Push/Relabel

- Levamos k unidades de fluxo de s a V_k
 - depois distribuímos essas k unidades pelos vértices m_1, \dots, m_k

Definição [Fluxo]

f é um **fluxo** em $G = (V, E, s, t, c)$ se e somente se:
satisfaz as seguintes restrições:

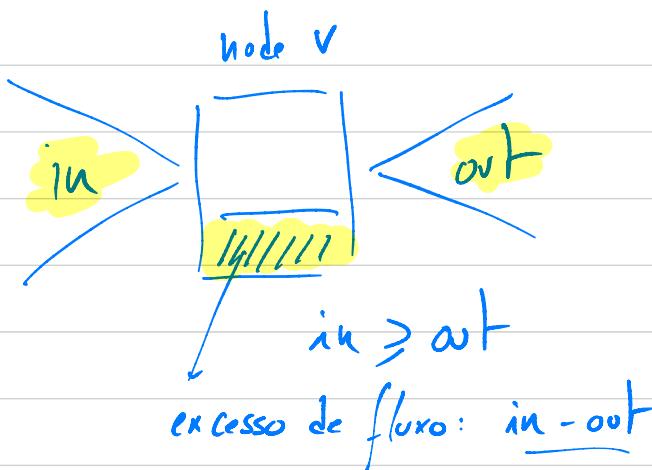
[Restrição de Capacidade]

$$\forall u, v \in V \quad 0 \leq f(u, v) \leq c(u, v)$$

[Conservação do Fluxo]

$$\forall u \in V \quad \sum_{v \in V} f(v, u) \geq \sum_{v \in V} f(u, v) \quad \left\{ \begin{array}{l} \text{Os vértices podem receber mais} \\ \text{fluxo do que aquele que têm} \\ \text{de "descartar"} \end{array} \right.$$

"flow in" \geq "flow out"



Definição [Fluxo]

f é um **fluxo** em $G = (V, E, s, t, c)$ se e só se

satisfaz as seguintes restrições:

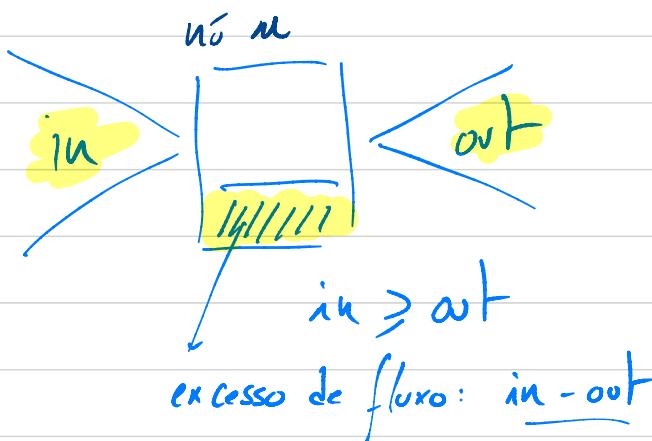
[Restrição de Capacidade]

$$\forall u, v \in V \quad 0 \leq f(u, v) \leq c(u, v)$$

[Conservação do Fluxo]

$$\forall u \in V \quad \sum_{v \in V} f(v, u) \geq \sum_{v \in V} f(u, v) \quad \left\{ \begin{array}{l} \text{Os vértices podem receber mais} \\ \text{fluxo do que aquele que têm} \\ \text{de "descartar"} \end{array} \right.$$

"flow in" \geq "flow out"



Definição [Fluxo]

f é um **fluxo** em $G = (V, E, s, t, c)$ se e somente se:
satisfaz as seguintes restrições:

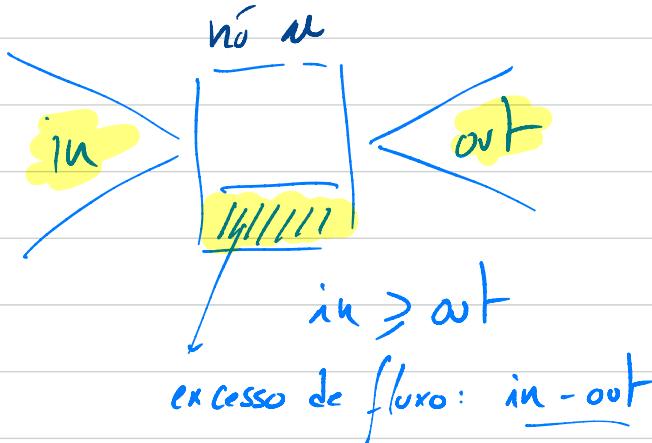
[Restrição de Capacidade]

$$\forall u, v \in V \quad 0 \leq f(u, v) \leq c(u, v)$$

[Conservação do Fluxo]

$$\forall u \in V \quad \sum_{v \in V} f(v, u) \geq \sum_{v \in V} f(u, v) \quad \left\{ \begin{array}{l} \text{Os vértices podem receber mais} \\ \text{fluxo do que aquele que têm} \\ \text{de "descartar"} \end{array} \right.$$

"flow in" \geq "flow out"



Definição [Excesso de Fluxo]

$$e_f(u) = \sum_{v \in V} f(v, u) - \sum_{v \in V} f(u, v)$$

Algoritmo de Push

Push (u, v)

$$\Delta := \min(u.e, c_f(u, v))$$

if $(u, v) \in E$

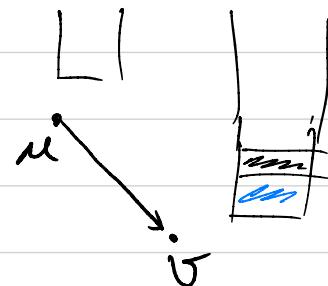
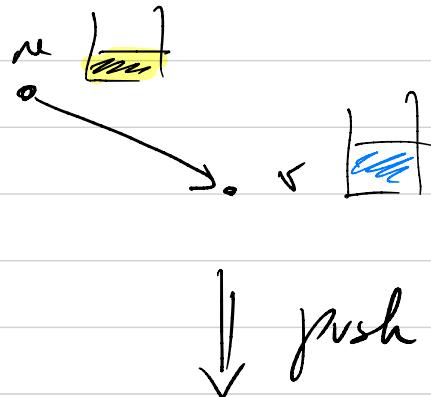
$$(u, v).f := (u, v).f + \Delta$$

else

$$(v, u).f := (v, u).f - \Delta$$

$$u.e := u.e - \Delta$$

$$v.e := v.e + \Delta$$



Complexidade:

Algoritmo de Push

Push (u, v)

$$\Delta := \min(u.e, c_f(u, v))$$

if $(u, v) \in E$

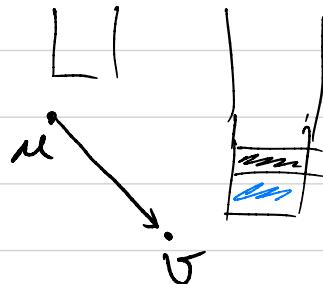
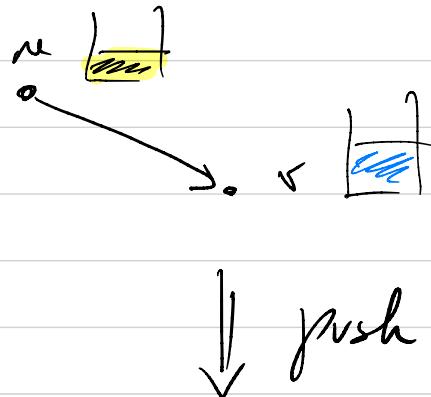
$$(u, v).f := (u, v).f + \Delta$$

else

$$(v, u).f := (v, u).f - \Delta$$

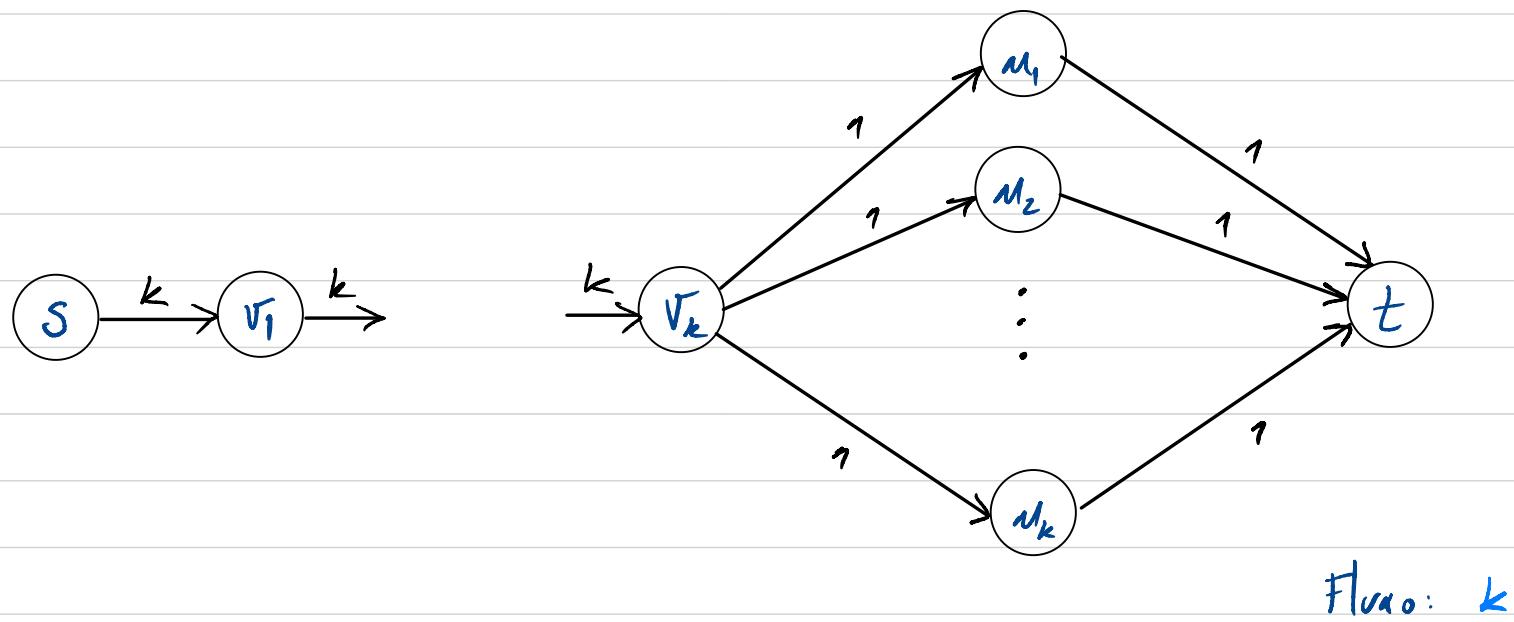
$$u.e := u.e - \Delta$$

$$v.e := v.e + \Delta$$



Complejidad: $O(1)$

Algoritmo de Push-Relabel

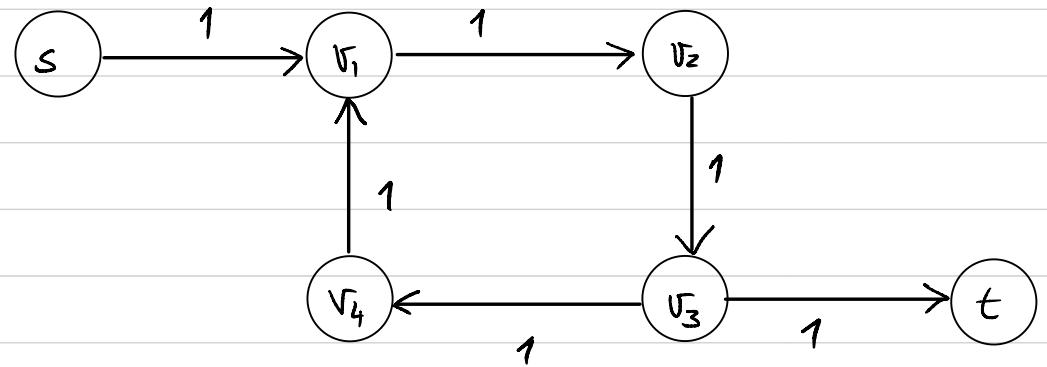


- Secuencia de operaciones de push

- Push (S, V_1)
- Push (V_1, V_2), ..., Push (V_{k-1}, V_k)
- Push (V_k, M_1), ..., Push (V_k, M_k)
- Push (M_1, t), ..., Push (M_k, t)

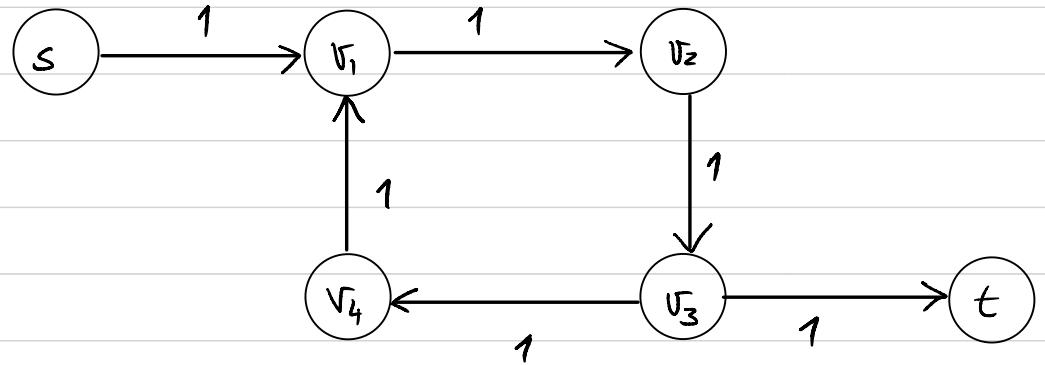
} N° total de operaciones de Push: $O(k)$
Complejidad: $\underline{O(k)}$

Algoritmo de Push-Pebble - Função de Altura



Problema?

Algoritmo de Push-Rebel - Função de Altura



Problema?

- Como é que garantimos q não fazemos pushes em loop?

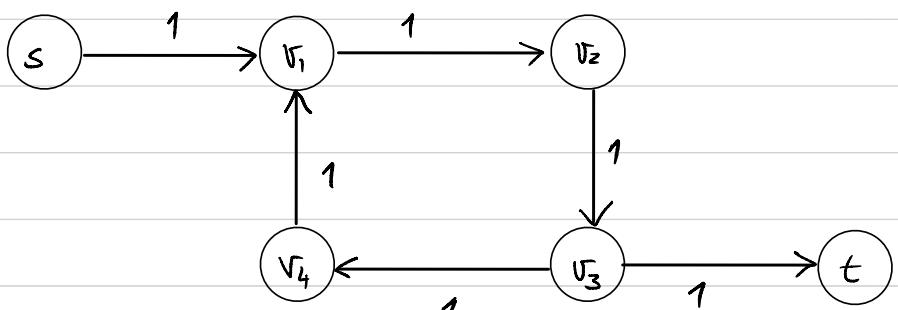
↳ Precisamos de uma "memória"
q nos impede de repetir pushes
já fizemos

Algoritmo de Push-Prelabel - Função de Altura

- Associamos a cada vértice uma altura.

No início o nó fonte (s) tem altura ∞ e todos os outros nós têm altura 0.

- Empurramos fluxo "downhill" (para baixo) mas não demais do necessário



$$h(v) = h(u) + 1$$

- A altura do vértice é incrementada quando o vértice tem excesso de fluxo e não tem nenhum vizinho mais baixo.

Método de Push-Relabel

Initialize (G, s)

for each $v \in V$

$v.e := 0$; $v.h := 0$

for each $(u, v) \in E$

$(u, v).f := 0$

for each $v \in G.\text{Adj}[s]$

$s.e := s.e - c(s, v)$

$v.e := c(s, v)$

$(s, v).f = c(s, v)$

$s.h := |G.V|$

Relabel (u)

$$u.h = \min \left\{ v.h + 1 \mid (u, v) \in E_f \right\}$$

Push (u, v)

$$\Delta := \min(u.e, c_f(u, v))$$

if $(u, v) \in E$

$$(u, v).f := (u, v).f + \Delta$$

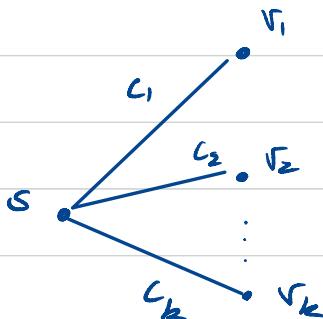
else

$$(v, u).f := (v, u).f - \Delta$$

$$u.e := u.e - \Delta$$

$$v.e := v.e + \Delta$$

- No início, esgotamos a capacidade de todas as arestas que partem do nó fonte



Método de Push-Relabel

Initialize (G, s)

for each $v \in V$

$v.e := 0$; $v.h := 0$

for each $(u, v) \in E$

$(u, v).f := 0$

for each $v \in G.V \setminus \{s\}$

$s.e := s.e - c(s, v)$

$v.e := c(s, v)$

$(s, v).f = c(s, v)$

$s.h := |G.V|$

Relabel (u)

$$u.h = \min \left\{ v.h + 1 \mid (u, v) \in E_f \right\}$$

Push (u, v)

$$\Delta := \min(u.e, c_f(u, v))$$

if $(u, v) \in E$

$$(u, v).f := (u, v).f + \Delta$$

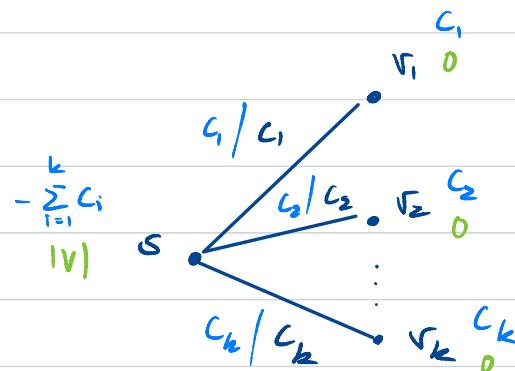
else

$$(v, u).f := (v, u).f - \Delta$$

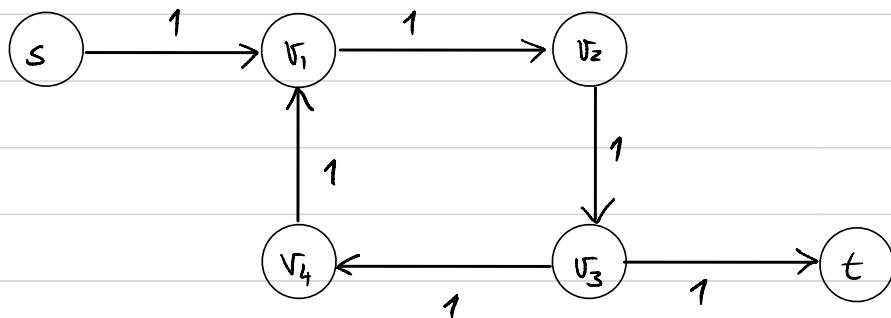
$$u.e := u.e - \Delta$$

$$v.e := v.e + \Delta$$

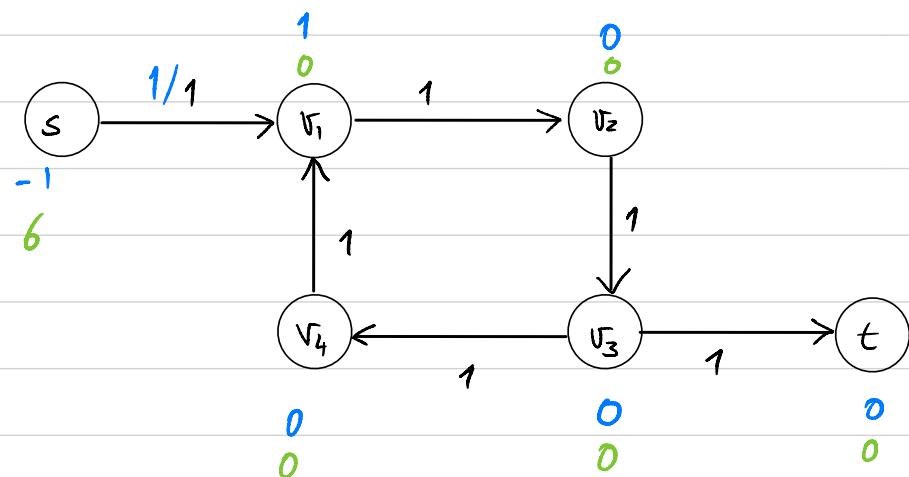
No início, esgotamos a capacidade
de todas as arestas que partem do nó fonte



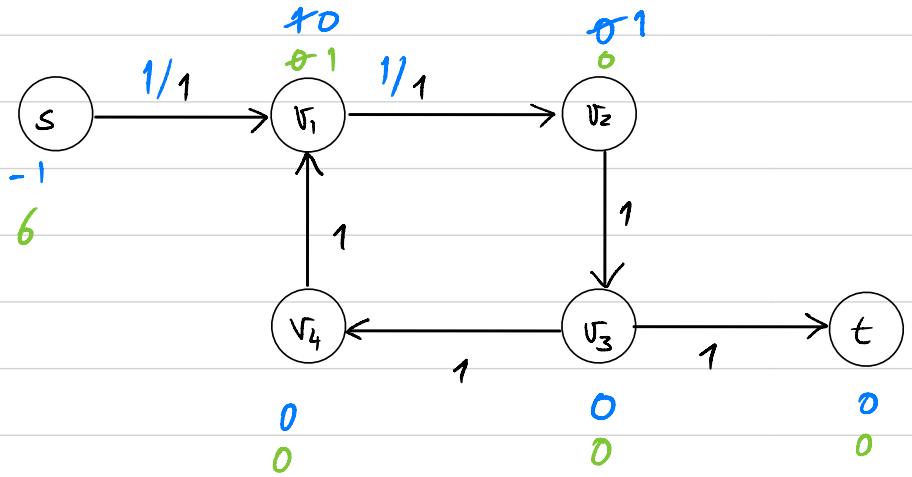
Método de Push-Relabel - Exemplo



I

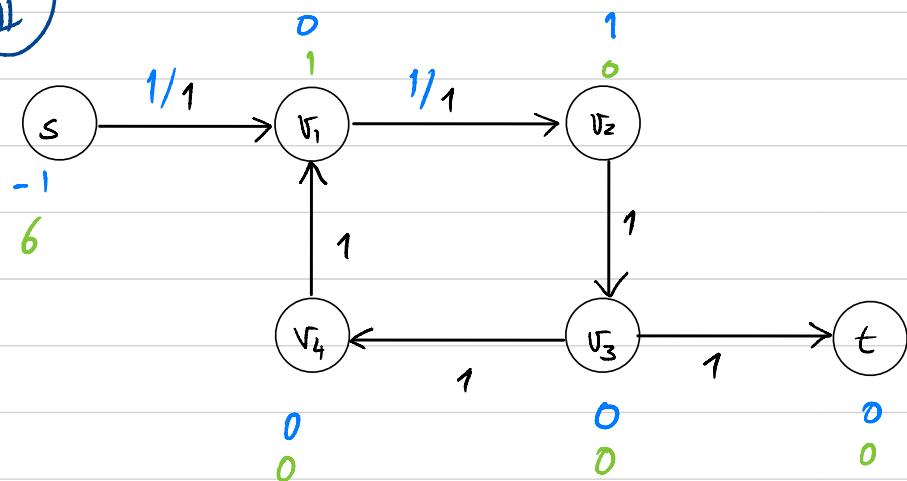


II

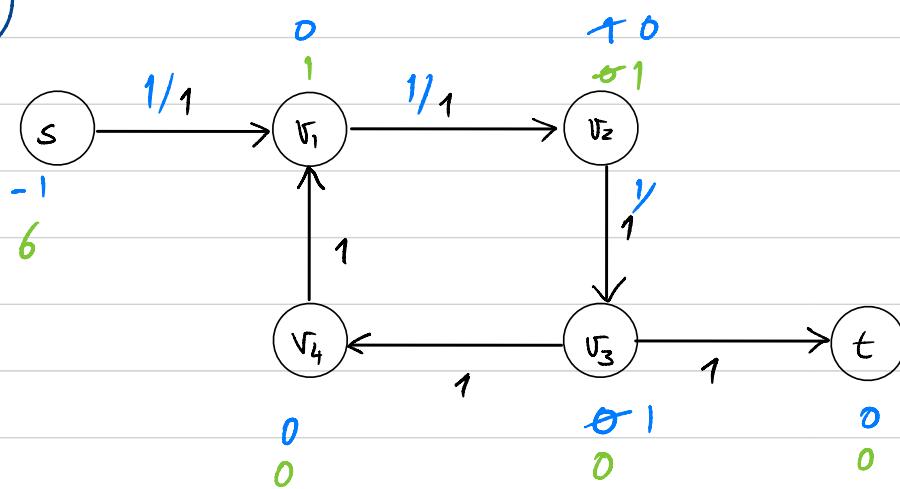


Método de Push-Relabel - Exemplo

II

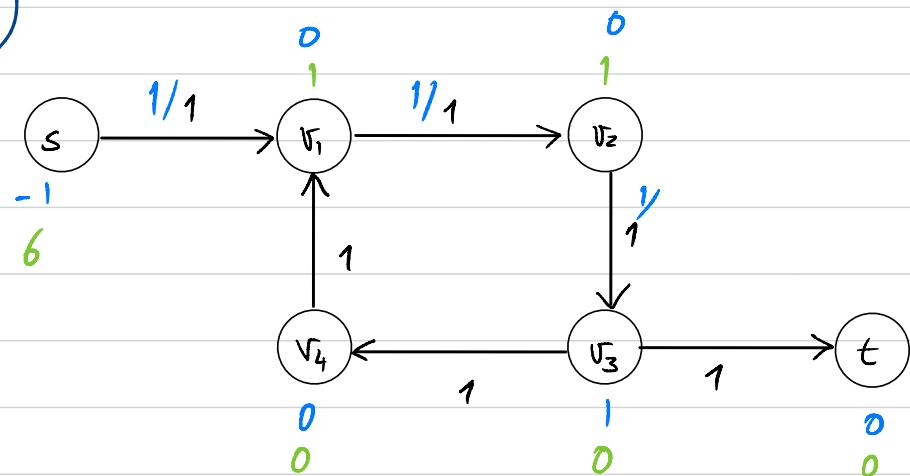


III

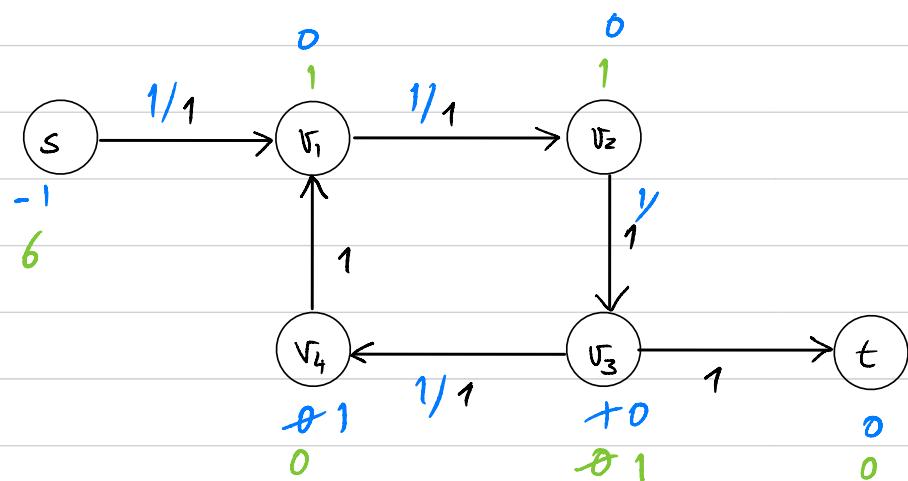


Método de Push-Relabel - Exemplo

III

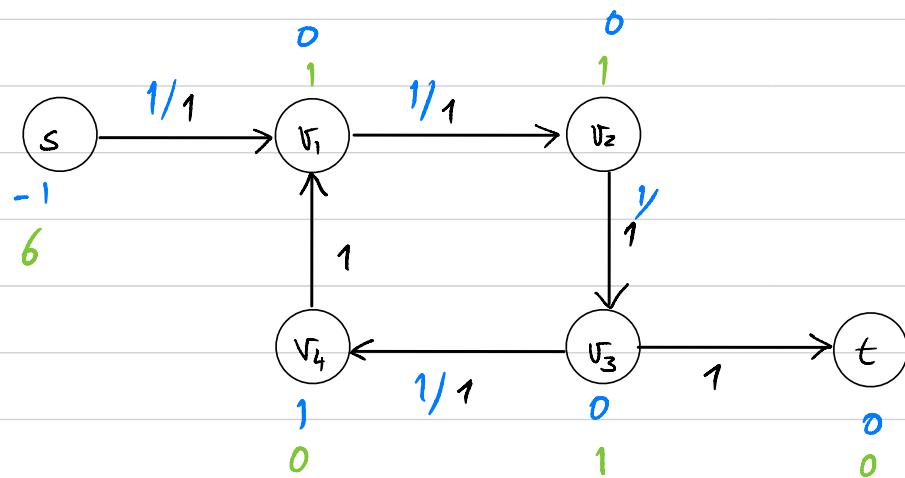


IV

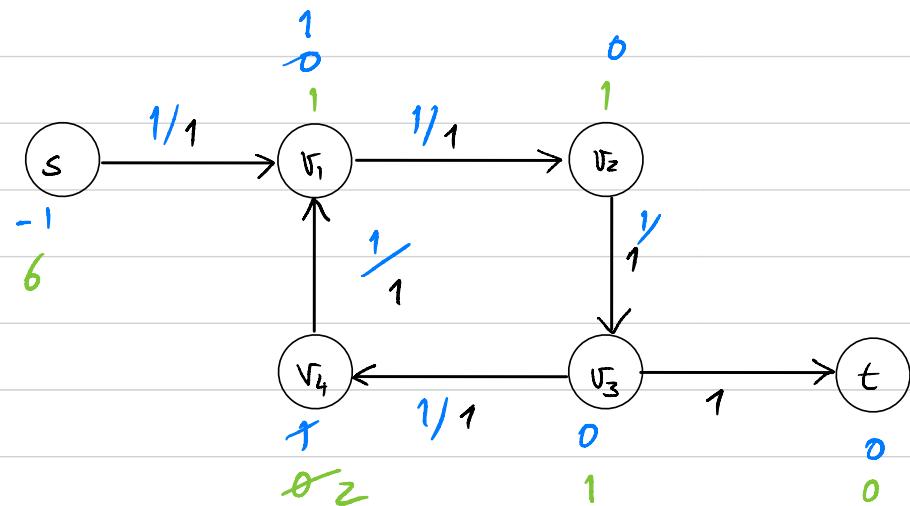


Método de Push-Relabel - Exemplo

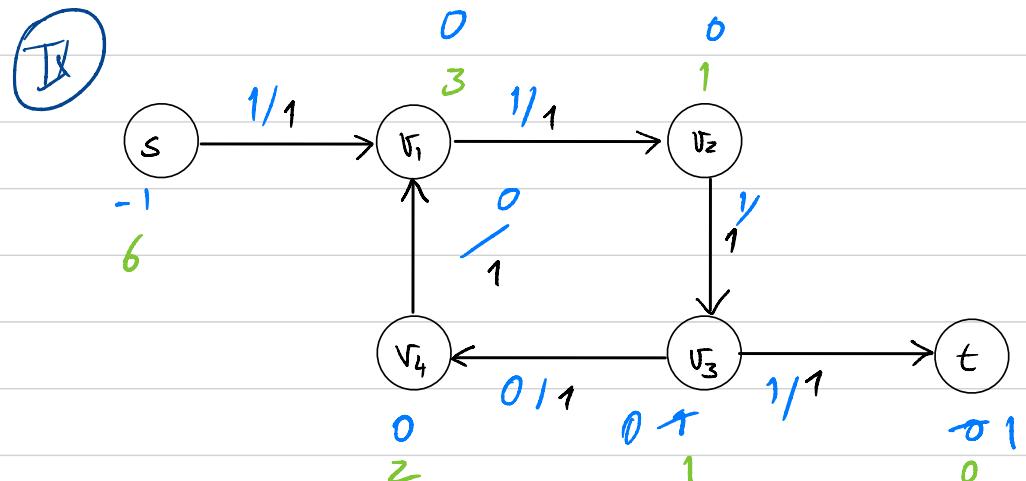
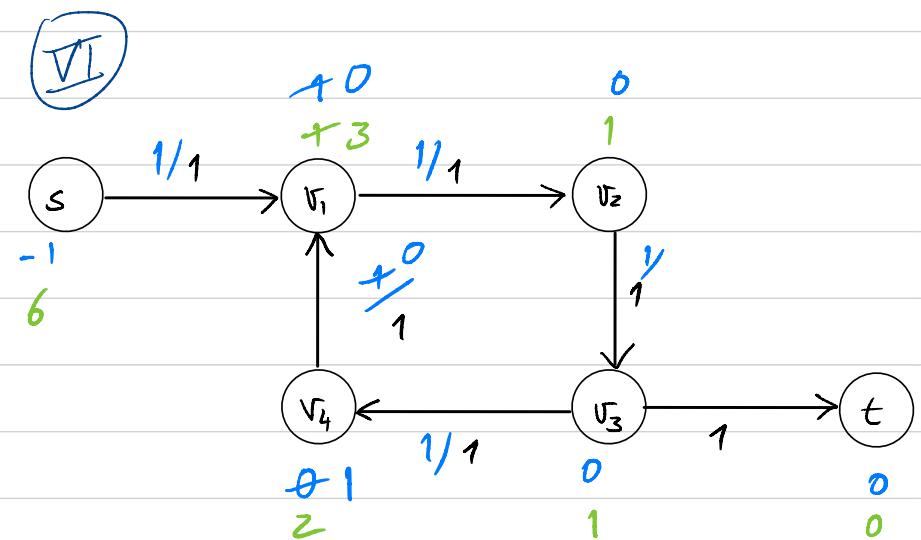
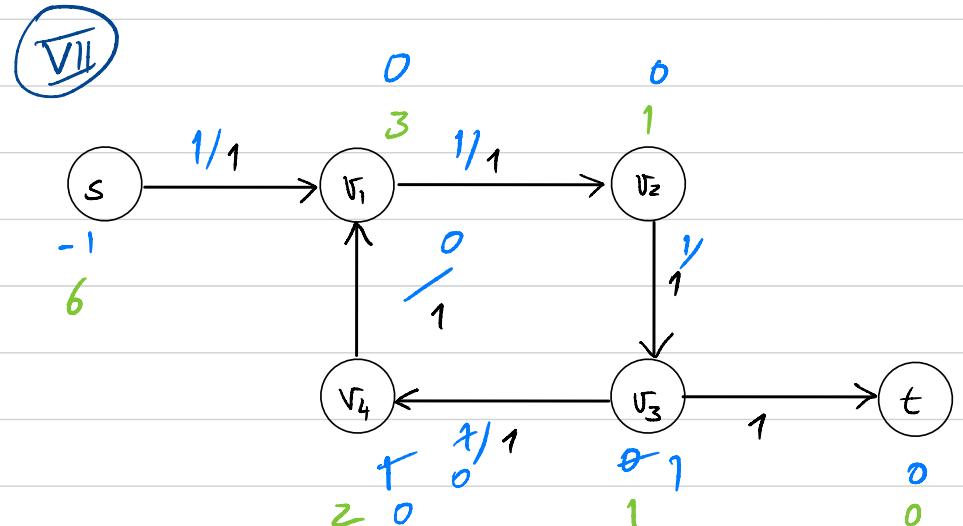
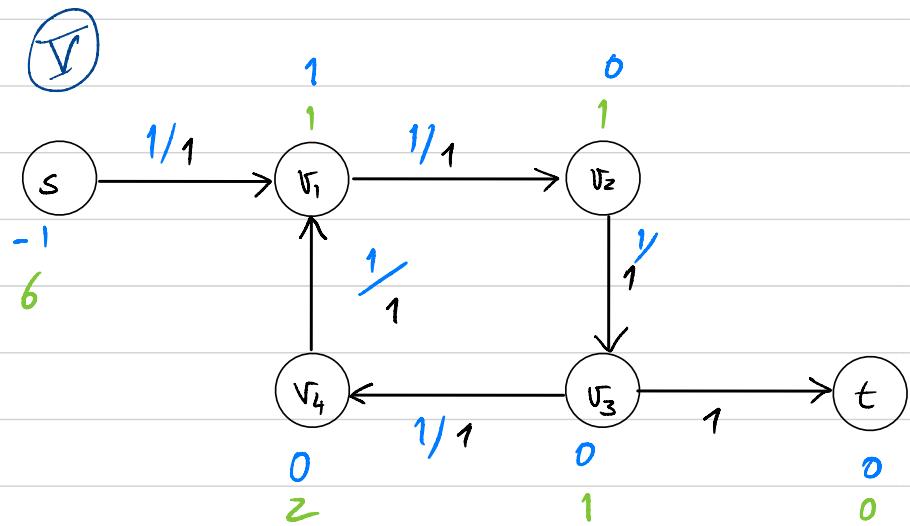
IV



V



Método de Push-Relabel - Exemplo



Push-Relabel

Push Relabel(G, s, t)

Initialize(G, s)

while $\exists u \in V \setminus \{s, t\} . u.e > 0$

let u be a vertex st. $u.e > 0$

let v be a vertex st. $(u, v) \in \bar{E} \wedge u.h = v.h + 1$

if ($v \neq nil$)

Push(u, v)

else Relabel(u)

• Conecto?

• Complexeidade?

• Invariante:

I1

é um pré-fluxo

Empurramos fluxo por
declives bairros

I2

$\forall u, v \in V . (u, v) \in E \Rightarrow u.h \leq v.h + 1$



IZ: Invariante de Alturas

$$\forall u, v \in V. (u, v) \in E_f \Rightarrow u.h \leq v.h + 1$$



Lema [Corte & Função de Altura]

Seja f um pré-fluxo numa rede de fluxo $G = (V, E, \alpha, t, c)$

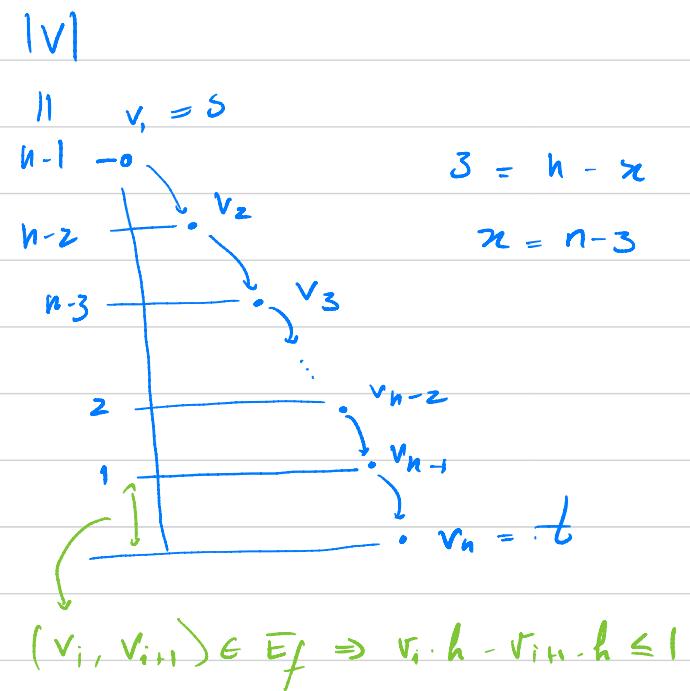
Se existir uma função de altura h consistente com f , então
não existe um caminho $s-t$ em G_f .

Prova:

- Suponhamos que f é um pré-fluxo em G , h uma função de altura consistente com f e, por contradição,
 $\langle v_1, \dots, v_n \rangle$ um caminho $s-t$ em G_f .
 s t

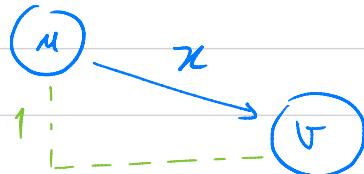
$$h \leq |V| \quad n-1 \geq |V| \Leftrightarrow n \geq |V|+1$$

∴



IZ: Invariante de Alturas

$$\forall u, v \in V. (u, v) \in E \Rightarrow u.h \leq v.h + 1$$



Lema [Corte & Função de Altura]

Seja f um pré-fluxo numa rede de fluxo $G = (V, E, \alpha, t, c)$

Se existir uma função de altura h consistente com f , então
não existe um caminho $s-t$ em G_f .

Prova:

- Suponhamos que f é um pré-fluxo em G , h uma função de altura consistente com f e, por contradição,
 $\langle v_1, \dots, v_n \rangle$ um caminho $s-t$ em G_f .
 s t

$$n \leq |V|$$

∴

$$\sum_{i=1}^{n-1} v_i.h - v_{i+1}.h = v_1.h - v_n.h = |V| \quad \left\{ \begin{matrix} |V| \leq n-1 \end{matrix} \right.$$

$$\sum_{i=1}^{n-1} \underbrace{v_i.h - v_{i+1}.h}_{\leq 1} \leq \sum_{i=1}^{n-1} 1 \leq n-1$$

Push-Relabel - Conexão

Push Relabel(G, s, t)

Initialize(G, s)

while $\exists u \in V \setminus \{s, t\} : d(u) > 0$

let u be a vertex s.t. $d(u) > 0$

let v be a vertex s.t. $(u, v) \in \bar{E}_f \wedge u \cdot h = v \cdot h + 1$

if ($v \neq t$)

Push(u, v)

else Relabel(u)

• Invariante:

I₁

f é um pré-fluxo

I_{2'}

Não existe um caminho s-t na rede residual

• No fim da execução do algoritmo:

- $\forall v \in V \setminus \{s, t\} : r_v = 0$

↓ I₁

• f é um fluxo

↓ I_{2'}

• f é um fluxo máximo