

Algoritmos gulosos (*greedy*)

KT 4.2 e CLRS 16.4

Problema de escalonamento

Considere n tarefas indicadas pelos números $1, \dots, n$

t_i : duração da tarefa i

d_i : prazo de entrega da tarefa i

Problema de escalonamento

Considere n tarefas indicadas pelos números $1, \dots, n$

t_i : duração da tarefa i

d_i : prazo de entrega da tarefa i

Escalonamento: permutação de 1 a n
que indica a ordem em que as tarefas são executadas.

Problema de escalonamento

Considere n tarefas indicadas pelos números $1, \dots, n$

t_i : duração da tarefa i

d_i : prazo de entrega da tarefa i

Escalonamento: permutação de 1 a n
que indica a ordem em que as tarefas são executadas.

Para um escalonamento π , se $i = \pi(k)$,
então o tempo de início da tarefa i é

$$s_i = \sum_{j=1}^{k-1} t_{\pi(j)}$$

(soma da duração das tarefas escalonadas antes de i).

Problema de escalonamento

Considere n tarefas indicadas pelos números $1, \dots, n$

t_i : duração da tarefa i

d_i : prazo de entrega da tarefa i

Escalonamento: permutação de 1 a n
que indica a ordem em que as tarefas são executadas.

Para um escalonamento π , se $i = \pi(k)$,
então o tempo de início da tarefa i é

$$s_i = \sum_{j=1}^{k-1} t_{\pi(j)}$$

(soma da duração das tarefas escalonadas antes de i).

O tempo de término da tarefa i é $f_i = s_i + t_i$.

Problema de escalonamento

Para um escalonamento π , o tempo de início da tarefa i é a soma da duração das tarefas escalonadas antes de i .

O tempo de término da tarefa i é $f_i = s_i + t_i$.

Problema de escalonamento

Para um escalonamento π , o tempo de início da tarefa i é a soma da duração das tarefas escalonadas antes de i .

O tempo de término da tarefa i é $f_i = s_i + t_i$.

O atraso da tarefa i é o número

$$l_i = \begin{cases} 0 & \text{se } f_i \leq d_i \\ f_i - d_i & \text{se } f_i > d_i. \end{cases}$$

Problema de escalonamento

Para um escalonamento π , o tempo de início da tarefa i é a soma da duração das tarefas escalonadas antes de i .

O tempo de término da tarefa i é $f_i = s_i + t_i$.

O atraso da tarefa i é o número

$$l_i = \begin{cases} 0 & \text{se } f_i \leq d_i \\ f_i - d_i & \text{se } f_i > d_i. \end{cases}$$

Problema: Dados $t_1, \dots, t_n$ e $d_1, \dots, d_n$, encontrar um escalonamento com o menor atraso máximo.

Ou seja, que minimize $L = \max_i l_i$.

Exemplos

d_i : deadline da tarefa i

t_i : tempo de processamento da tarefa i

Dado escalonamento π ,

s_i : início do processamento da tarefa i

f_i : fim do processamento da tarefa i

ℓ_i : atraso da tarefa i

Exemplos

d_i : deadline da tarefa i

t_i : tempo de processamento da tarefa i

Dado escalonamento π ,

s_i : início do processamento da tarefa i

f_i : fim do processamento da tarefa i

ℓ_i : atraso da tarefa i

Problema: Dados d e t , encontrar π cujo atraso máximo é mínimo.

Exemplos

d_i : deadline da tarefa i

t_i : tempo de processamento da tarefa i

Dado escalonamento π ,

s_i : início do processamento da tarefa i

f_i : fim do processamento da tarefa i

ℓ_i : atraso da tarefa i

Problema: Dados d e t , encontrar π cujo atraso máximo é mínimo.

Exemplo 1: $t_1 = 1$ e $d_1 = 2$, $t_2 = 2$ e $d_2 = 4$, $t_3 = 3$ e $d_3 = 6$.

Exemples

d_i : deadline da tarefa i

t_j : tempo de processamento da tarefa i

Dado escalonamento π ,

s_i : início do processamento da tarefa i

f_i : fim do processamento da tarefa i

ℓ_i : atraso da tarefa i

Problema: Dados d e t , encontrar π cujo atraso máximo é mínimo.

Exemplo 1: $t_1 = 1$ e $d_1 = 2$, $t_2 = 2$ e $d_2 = 4$, $t_3 = 3$ e $d_3 = 6$.

Escalonamento com atraso mínimo: $(1, 2, 3)$



Atrasos: $\ell_1 = \ell_2 = \ell_3 = 0$.

Exemplos

d_i : deadline da tarefa i

t_i : tempo de processamento da tarefa i

Dado escalonamento π ,

s_i : início do processamento da tarefa i

f_i : fim do processamento da tarefa i

ℓ_i : atraso da tarefa i

Problema: Dados d e t , encontrar π cujo atraso máximo é mínimo.

Exemplo 2: $t_1 = 1$ e $d_1 = 4$, $t_2 = 2$ e $d_2 = 5$, $t_3 = 3$ e $d_3 = 3$.

Exemplos

d_i : deadline da tarefa i

t_i : tempo de processamento da tarefa i

Dado escalonamento π ,

s_i : início do processamento da tarefa i

f_i : fim do processamento da tarefa i

ℓ_i : atraso da tarefa i

Problema: Dados d e t , encontrar π cujo atraso máximo é mínimo.

Exemplo 2: $t_1 = 1$ e $d_1 = 4$, $t_2 = 2$ e $d_2 = 5$, $t_3 = 3$ e $d_3 = 3$.

Escalonamento com atraso mínimo: (3, 1, 2)



Atrasos: $\ell_1 = 0$, $\ell_2 = 1$, e $\ell_3 = 0$.

Possíveis critérios gulosos

- ▶ SPT - shortest processing time (menor t_i primeiro)

Funciona?

Possíveis critérios gulosos

- ▶ SPT - shortest processing time (menor t_i primeiro)

Funciona? Não...

Exemplo: $d_1 = 9$ e $t_1 = 1$, $d_2 = 8$ e $t_2 = 8$.

Possíveis critérios gulosos

- ▶ SPT - shortest processing time (menor t_i primeiro)

Funciona? Não...

Exemplo: $d_1 = 9$ e $t_1 = 1$, $d_2 = 8$ e $t_2 = 8$.



Possíveis critérios gulosos

- ▶ SPT - shortest processing time (menor t_i primeiro)

Funciona? Não...

Exemplo: $d_1 = 9$ e $t_1 = 1$, $d_2 = 8$ e $t_2 = 8$.



- ▶ LST - least slack time (menor $d_i - t_i$ primeiro)

Funciona?

Possíveis critérios gulosos

- ▶ SPT - shortest processing time (menor t_i primeiro)

Funciona? Não...

Exemplo: $d_1 = 9$ e $t_1 = 1$, $d_2 = 8$ e $t_2 = 8$.



- ▶ LST - least slack time (menor $d_i - t_i$ primeiro)

Funciona? Também não...

Exemplo: $d_1 = 2$ e $t_1 = 1$, $d_2 = 8$ e $t_2 = 8$.

Possíveis critérios gulosos

- ▶ SPT - shortest processing time (menor t_i primeiro)

Funciona? Não...

Exemplo: $d_1 = 9$ e $t_1 = 1$, $d_2 = 8$ e $t_2 = 8$.



- ▶ LST - least slack time (menor $d_i - t_i$ primeiro)

Funciona? Também não...

Exemplo: $d_1 = 2$ e $t_1 = 1$, $d_2 = 8$ e $t_2 = 8$.



Possíveis critérios gulosos

- ▶ LST - least slack time (menor $d_i - t_i$ primeiro)

Funciona? Também não...

Exemplo: $d_1 = 2$ e $t_1 = 1$, $d_2 = 8$ e $t_2 = 8$.



- ▶ EDD - earliest due date (menor d_i primeiro)

Funciona?

Exemplo

$$n = 5$$

i	1	2	3	4	5
t_i	6	2	1	2	4
d_i	8	3	5	12	10

Exemplo

$$n = 5$$

i	1	2	3	4	5
t_i	6	2	1	2	4
d_i	8	3	5	12	10

Para $\pi = (2, 3, 1, 5, 4)$, temos $L = 3$:

i	1	2	3	4	5
f_i	9	2	3	15	13
ℓ_i	1	0	0	3	3

Exemplo

$n = 5$

i	1	2	3	4	5
t_i	6	2	1	2	4
d_i	8	3	5	12	10

Para $\pi = (2, 3, 1, 5, 4)$, temos $L = 3$:

i	1	2	3	4	5
f_i	9	2	3	15	13
ℓ_i	1	0	0	3	3

Será que tem escalonamento melhor?

Algoritmo guloso

Devolve escalonamento com atraso máximo mínimo.

ESCALONAMENTO (t, d, n)

1 seja π a permutação de 1 a n tal que

$$d[\pi[1]] \leq d[\pi[2]] \leq \dots \leq d[\pi[n]]$$

2 devolva π

Algoritmo guloso

Devolve escalonamento com atraso máximo mínimo.

ESCALONAMENTO (t, d, n)

1 seja π a permutação de 1 a n tal que

$$d[\pi[1]] \leq d[\pi[2]] \leq \dots \leq d[\pi[n]]$$

2 devolva π

Consumo de tempo: $O(n \lg n)$.

Algoritmo guloso

Devolve escalonamento com atraso máximo mínimo.

ESCALONAMENTO (t, d, n)

1 seja π a permutação de 1 a n tal que

$$d[\pi[1]] \leq d[\pi[2]] \leq \dots \leq d[\pi[n]]$$

2 devolva π

Consumo de tempo: $O(n \lg n)$.

Na aula, vimos a prova de que este algoritmo está correto (devolve um escalonamento com atraso máximo mínimo).

Ingredientes da prova

Uma **inversão** em um escalonamento π é um par (i, j) tal que $i < j$ e $d[\pi[i]] > d[\pi[j]]$.

Ingredientes da prova

Uma **inversão** em um escalonamento π é um par (i, j) tal que $i < j$ e $d[\pi[i]] > d[\pi[j]]$.

Mostre que dois escalonamentos sem inversões têm o mesmo atraso máximo.

Ingredientes da prova

Uma **inversão** em um escalonamento π é um par (i, j) tal que $i < j$ e $d[\pi[i]] > d[\pi[j]]$.

Mostre que dois escalonamentos sem inversões têm o mesmo atraso máximo.

Mostre que se um escalonamento tem uma inversão, então ele tem uma inversão do tipo $(i, i + 1)$.

Ingredientes da prova

Uma **inversão** em um escalonamento π é um par (i, j) tal que $i < j$ e $d[\pi[i]] > d[\pi[j]]$.

Mostre que dois escalonamentos sem inversões têm o mesmo atraso máximo.

Mostre que se um escalonamento tem uma inversão, então ele tem uma inversão do tipo $(i, i + 1)$.

Mostre então que, se trocarmos $\pi[i]$ e $\pi[i + 1]$, obteremos um escalamento com atraso máximo não superior ao atraso máximo de π .

Primeiro passo

Afirmção: Dois escalonamentos sem inversões têm o mesmo atraso máximo.

Primeiro passo

Afirmção: Dois escalonamentos sem inversões têm o mesmo atraso máximo.

Prova: Escalonamento sem inversões:
sequência de blocos B_j de tarefas com o mesmo prazo P_j .

Primeiro passo

Afirmção: Dois escalonamentos sem inversões têm o mesmo atraso máximo.

Prova: Escalonamento sem inversões:
sequência de blocos B_j de tarefas com o mesmo prazo P_j .

O término da última tarefa do bloco B_j é o mesmo, digamos, F_j , independente da ordem das tarefas do bloco.

Assim a tarefa mais atrasada de cada bloco é a última, e seu atraso é o mesmo, independente da ordem das tarefas do bloco, pois todas as tarefas do bloco têm o mesmo prazo.

O atraso L_j da última tarefa em B_j é zero se $F_j \leq P_j$, e $F_j - P_j$ se $F_j > P_j$.

Primeiro passo

Afirmção: Dois escalonamentos sem inversões têm o mesmo atraso máximo.

Prova: Escalonamento sem inversões:
sequência de blocos B_j de tarefas com o mesmo prazo P_j .

O término da última tarefa do bloco B_j é o mesmo, digamos, F_j , independente da ordem das tarefas do bloco.

Assim a tarefa mais atrasada de cada bloco é a última, e seu atraso é o mesmo, independente da ordem das tarefas do bloco, pois todas as tarefas do bloco têm o mesmo prazo.

O atraso L_j da última tarefa em B_j é zero se $F_j \leq P_j$, e $F_j - P_j$ se $F_j > P_j$.

Assim, o atraso máximo L é o máximo dos atrasos L_j das últimas tarefas dos blocos.

Segundo passo

Uma **inversão** em um escalonamento π é um par (i, j) tal que $i < j$ e $d[\pi[i]] > d[\pi[j]]$.

Afirmção: Se um escalonamento tem uma inversão então ele tem uma inversão do tipo $(i, i + 1)$.

Segundo passo

Uma **inversão** em um escalonamento π é um par (i, j) tal que $i < j$ e $d[\pi[i]] > d[\pi[j]]$.

Afirmção: Se um escalonamento tem uma inversão então ele tem uma inversão do tipo $(i, i + 1)$.

Prova: Se $d[\pi[i]] \leq d[\pi[i + 1]]$ para todo i , então π está ordenado e não tem inversões.

Terceiro passo

Uma **inversão** em um escalonamento π é um par (i, j) tal que $i < j$ e $d[\pi[i]] > d[\pi[j]]$.

Afirmção: Se $(i, i + 1)$ é uma inversão e trocarmos $\pi[i]$ e $\pi[i + 1]$, obteremos um escalonamento com atraso máximo não superior ao atraso máximo de π .

Terceiro passo

Uma **inversão** em um escalonamento π é um par (i, j) tal que $i < j$ e $d[\pi[i]] > d[\pi[j]]$.

Afirmção: Se $(i, i + 1)$ é uma inversão e trocarmos $\pi[i]$ e $\pi[i + 1]$, obteremos um escalonamento com atraso máximo não superior ao atraso máximo de π .

Prova: Quando invertemos as tarefas i e $i + 1$ de ordem, isso não afeta o término de nenhuma das outras tarefas.

Terceiro passo

Uma **inversão** em um escalonamento π é um par (i, j) tal que $i < j$ e $d[\pi[i]] > d[\pi[j]]$.

Afirmção: Se $(i, i + 1)$ é uma inversão e trocarmos $\pi[i]$ e $\pi[i + 1]$, obteremos um escalonamento com atraso máximo não superior ao atraso máximo de π .

Prova: Quando invertemos as tarefas i e $i + 1$ de ordem, isso não afeta o término de nenhuma das outras tarefas.

Como $(i, i + 1)$ é uma inversão, $d[\pi[i]] > d[\pi[i + 1]]$.

Se $i + 1$ não está atrasada, então i também não está atrasada, e inverter a ordem de i e $i + 1$ não as torna atrasadas.

Terceiro passo

Uma **inversão** em um escalonamento π é um par (i, j) tal que $i < j$ e $d[\pi[i]] > d[\pi[j]]$.

Afirmção: Se $(i, i + 1)$ é uma inversão e trocarmos $\pi[i]$ e $\pi[i + 1]$, obteremos um escalonamento com atraso máximo não superior ao atraso máximo de π .

Prova: Quando invertemos as tarefas i e $i + 1$ de ordem, isso não afeta o término de nenhuma das outras tarefas.

Como $(i, i + 1)$ é uma inversão, $d[\pi[i]] > d[\pi[i + 1]]$.

Se $i + 1$ não está atrasada, então i também não está atrasada, e inverter a ordem de i e $i + 1$ não as torna atrasadas.

Se $i + 1$ está atrasada, então está mais atrasada que i . Ao invertermos i e $i + 1$, a tarefa $i + 1$ fica menos atrasada e a tarefa i fica menos atrasada do que $i + 1$ estava.

Matroides e o método guloso

U : conjunto finito arbitrário.

$\mathcal{C}$: família não vazia de subconjuntos de U hereditária.
(subconjuntos de conjuntos em $\mathcal{C}$ estão em $\mathcal{C}$)

Matroides e o método guloso

U : conjunto finito arbitrário.

$\mathcal{C}$: família não vazia de subconjuntos de U hereditária.

Problema 1: Dado um peso binário w_e para cada e de U , encontrar um conjunto de $\mathcal{C}$ de peso máximo.

Matroides e o método guloso

U : conjunto finito arbitrário.

$\mathcal{C}$: família não vazia de subconjuntos de U hereditária.

Problema 1: Dado um peso binário w_e para cada e de U , encontrar um conjunto de $\mathcal{C}$ de peso máximo.

GULOSO ($U, w, \mathcal{C}$)

- 1 $U_1 \leftarrow \{e \in U : w_e = 1\} \quad S \leftarrow \emptyset$
- 2 **enquanto** existe e em U_1 tal que $S \cup \{e\} \in \mathcal{C}$ **faça**
- 3 $S \leftarrow S \cup \{e\}$
- 4 **devolva** S

Matroides e o método guloso

U : conjunto finito arbitrário.

$\mathcal{C}$: família não vazia de subconjuntos de U hereditária.

Problema 1: Dado um peso binário w_e para cada e de U , encontrar um conjunto de $\mathcal{C}$ de peso máximo.

GULOSO ($U, w, \mathcal{C}$)

- 1 $U_1 \leftarrow \{e \in U : w_e = 1\} \quad S \leftarrow \emptyset$
- 2 **enquanto** existe e em U_1 tal que $S \cup \{e\} \in \mathcal{C}$ **faça**
- 3 $S \leftarrow S \cup \{e\}$
- 4 **devolva** S

GULOSO encontra um conjunto de $\mathcal{C}$ de peso **maximal**.

$\mathcal{C}$ é um **matroide** se
todo conjunto de $\mathcal{C}$ de peso **maximal** tem o mesmo tamanho.

Matroides e o método guloso

U : conjunto finito.

$\mathcal{C}$: família não vazia de subconjuntos de U hereditária.

Pesos positivos para os elementos de U .

Problema 2: Encontrar um conjunto de $\mathcal{C}$ de peso máximo.

Matroides e o método guloso

U : conjunto finito.

$\mathcal{C}$: família não vazia de subconjuntos de U hereditária.

Pesos positivos para os elementos de U .

Problema 2: Encontrar um conjunto de $\mathcal{C}$ de peso máximo.

GULOSO ($U, w, \mathcal{C}$)

- 1 $(e_1, \dots, e_n) \leftarrow \text{ORDENE}(U, w)$ $\triangleright$ ordem dos pesos
- 2 $S \leftarrow \emptyset$
- 3 **para** $i \leftarrow 1$ **até** n **faça**
- 4 **se** $S \cup \{e_i\} \in \mathcal{C}$
- 5 **então** $S \leftarrow S \cup \{e_i\}$
- 6 **devolva** S

Matroides e o método guloso

U : conjunto finito.

$\mathcal{C}$: família não vazia de subconjuntos de U hereditária.

Pesos positivos para os elementos de U .

Problema 2: Encontrar um conjunto de $\mathcal{C}$ de peso máximo.

GULOSO ($U, w, \mathcal{C}$)

- 1 $(e_1, \dots, e_n) \leftarrow \text{ORDENE}(U, w)$ $\triangleright$ ordem dos pesos
- 2 $S \leftarrow \emptyset$
- 3 **para** $i \leftarrow 1$ **até** n **faça**
- 4 **se** $S \cup \{e_i\} \in \mathcal{C}$
- 5 **então** $S \leftarrow S \cup \{e_i\}$
- 6 **devolva** S

Teorema: Se $\mathcal{C}$ é um matroide,
então o algoritmo acima resolve o Problema 2.

Exemplos de matroides

Dado um espaço vetorial, a coleção de todos os conjuntos de vetores LI deste espaço é um matroide.

Exemplos de matroides

Dado um espaço vetorial, a coleção de todos os conjuntos de vetores LI deste espaço é um matroide.

Dado um grafo G , a coleção de arestas de todas as florestas de G é um matroide.

Exemplos de matroides

Dado um espaço vetorial, a coleção de todos os conjuntos de vetores LI deste espaço é um matroide.

Dado um grafo G , a coleção de arestas de todas as florestas de G é um matroide.

Grafo bipartido $G = (U \times V, E)$.

M_U : todos os conjuntos S de arestas de G tq no máximo uma aresta de S é incidente a cada vértice de U .

M_U é um matroide.

Exemplos de matroides

Dado um espaço vetorial, a coleção de todos os **conjuntos de vetores LI** deste espaço é um matroide.

Dado um grafo G , a coleção de arestas de todas as **florestas** de G é um matroide.

Grafo bipartido $G = (U \times V, E)$.

M_U : todos os conjuntos S de arestas de G tq no máximo uma aresta de S é incidente a cada vértice de U .

M_U é um matroide.

M_V : a coleção análoga com V no lugar de U .

Claro que M_V também é um matroide.

Exemplos de matroides

Dado um espaço vetorial, a coleção de todos os **conjuntos de vetores LI** deste espaço é um matroide.

Dado um grafo G , a coleção de arestas de todas as **florestas** de G é um matroide.

Grafo bipartido $G = (U \times V, E)$.

M_U : todos os conjuntos S de arestas de G tq no máximo uma aresta de S é incidente a cada vértice de U .

M_U é um matroide.

M_V : a coleção análoga com V no lugar de U .

Claro que M_V também é um matroide.

E $M_U \cap M_V$? É ou não é um matroide?

Exemplos de matroides

Grafo bipartido $G = (U \times V, E)$.

M_U : todos os conjuntos S de arestas de G tq no máximo uma aresta de S é incidente a cada vértice de U .

M_V : a coleção analoga com V no lugar de U .

M_U e M_V são matroides.

Exemplos de matroides

Grafo bipartido $G = (U \times V, E)$.

M_U : todos os conjuntos S de arestas de G tq no máximo uma aresta de S é incidente a cada vértice de U .

M_V : a coleção analoga com V no lugar de U .

M_U e M_V são matroides.

O que é $M_U \cap M_V$?

O que é um conjunto da coleção $M_U \cap M_V$ em G ?

Exemplos de matroides

Grafo bipartido $G = (U \times V, E)$.

M_U : todos os conjuntos S de arestas de G tq no máximo uma aresta de S é incidente a cada vértice de U .

M_V : a coleção analoga com V no lugar de U .

M_U e M_V são matroides.

O que é $M_U \cap M_V$?

O que é um conjunto da coleção $M_U \cap M_V$ em G ?

Cada conjunto de $M_U \cap M_V$ é um **emparelhamento** em G .

Exemplos de matroides

Grafo bipartido $G = (U \times V, E)$.

M_U : todos os conjuntos S de arestas de G tq no máximo uma aresta de S é incidente a cada vértice de U .

M_V : a coleção analoga com V no lugar de U .

M_U e M_V são matroides.

O que é $M_U \cap M_V$?

O que é um conjunto da coleção $M_U \cap M_V$ em G ?

Cada conjunto de $M_U \cap M_V$ é um **emparelhamento** em G .

Esta coleção é ou não é um matroide?

Exemplos de matroides

Grafo bipartido $G = (U \times V, E)$.

M_U : todos os conjuntos S de arestas de G tq no máximo uma aresta de S é incidente a cada vértice de U .

M_V : a coleção analoga com V no lugar de U .

M_U e M_V são matroides.

O que é $M_U \cap M_V$?

O que é um conjunto da coleção $M_U \cap M_V$ em G ?

Cada conjunto de $M_U \cap M_V$ é um **emparelhamento** em G .

Esta coleção é ou não é um matroide?

Não é... (nem todo emparelhamento maximal é máximo)

Mas é a **interseção** de dois matroides.

Exemplos de matroides

Grafo bipartido $G = (U \times V, E)$.

M_U : todos os conjuntos S de arestas de G tq no máximo uma aresta de S é incidente a cada vértice de U .

M_V : a coleção analoga com V no lugar de U .

M_U e M_V são matroides.

$M_U \cap M_V$ é a coleção dos **emparelhamentos** de G .

Esta coleção não é um matroide.

Mas é a **interseção** de dois matroides.

Exemplos de matroides

Grafo bipartido $G = (U \times V, E)$.

M_U : todos os conjuntos S de arestas de G tq no máximo uma aresta de S é incidente a cada vértice de U .

M_V : a coleção analoga com V no lugar de U .

M_U e M_V são matroides.

$M_U \cap M_V$ é a coleção dos **emparelhamentos** de G .

Esta coleção não é um matroide.

Mas é a **interseção** de dois matroides.

Existe algoritmo polinomial para encontrar um conjunto (de peso) máximo na interseção de dois matroides.

Definição alternativa de matroides

U : conjunto finito arbitrário.

$\mathcal{C}$: família não vazia de subconjuntos de U hereditária.

Definição alternativa de matroides

U : conjunto finito arbitrário.

$\mathcal{C}$: família não vazia de subconjuntos de U hereditária.

$\mathcal{C}$ é um **matroide** se, para todo par A, B de conjuntos de $\mathcal{C}$ para os quais $|A| < |B|$, existe $e \in B \setminus A$ tal que $A \cup \{e\} \in \mathcal{C}$.

Definição alternativa de matroides

U : conjunto finito arbitrário.

$\mathcal{C}$: família não vazia de subconjuntos de U hereditária.

$\mathcal{C}$ é um **matroide** se, para todo par A, B de conjuntos de $\mathcal{C}$ para os quais $|A| < |B|$, existe $e \in B \setminus A$ tal que $A \cup \{e\} \in \mathcal{C}$.

Esta é a definição do CLRS.

Definição alternativa de matroides

U : conjunto finito arbitrário.

$\mathcal{C}$: família não vazia de subconjuntos de U hereditária.

$\mathcal{C}$ é um **matroide** se, para todo par A, B de conjuntos de $\mathcal{C}$ para os quais $|A| < |B|$, existe $e \in B \setminus A$ tal que $A \cup \{e\} \in \mathcal{C}$.

Esta é a definição do CLRS.

Exercício: Mostre que esta definição é análoga a anterior.