Tópicos de Análise de Algoritmos

Parte destes slides são adaptações de slides

do Prof. Paulo Feofiloff e do Prof. José Coelho de Pina.

AULA 10

Método guloso

Sec 4.1 do KT e 16.1 do CLRS

Algoritmos gulosos

"A *greedy algorithm* starts with a solution to a very small subproblem and augments it successively to a solution for the big problem.

Algoritmos gulosos

"A *greedy algorithm* starts with a solution to a very small subproblem and augments it successively to a solution for the big problem. The augmentation is done in a "greedy" fashion, that is, paying attention to short-term or local gain, without regard to whether it will lead to a good long-term or global solution.

Algoritmos gulosos

"A *greedy algorithm* starts with a solution to a very small subproblem and augments it successively to a solution for the big problem. The augmentation is done in a "greedy" fashion, that is, paying attention to short-term or local gain, without regard to whether it will lead to a good long-term or global solution. As in real life, greedy algorithms sometimes lead to the best solution, sometimes lead to pretty good solutions, and sometimes lead to lousy solutions.

Algoritmos gulosos

"A *greedy algorithm* starts with a solution to a very small subproblem and augments it successively to a solution for the big problem. The augmentation is done in a "greedy" fashion, that is, paying attention to short-term or local gain, without regard to whether it will lead to a good long-term or global solution. As in real life, greedy algorithms sometimes lead to the best solution, sometimes lead to pretty good solutions, and sometimes lead to lousy solutions. The trick is to determine when to be greedy."

Algoritmos gulosos

"A *greedy algorithm* starts with a solution to a very small subproblem and augments it successively to a solution for the big problem. The augmentation is done in a "greedy" fashion, that is, paying attention to short-term or local gain, without regard to whether it will lead to a good long-term or global solution. As in real life, greedy algorithms sometimes lead to the best solution, sometimes lead to pretty good solutions, and sometimes lead to lousy solutions. The trick is to determine when to be greedy."

"One thing you will notice about greedy algorithms is that they are usually easy to design, easy to implement, easy to analyse, and they are very fast, but they are *almost always difficult to prove correct*."

I. Parberry, *Problems on Algorithms*, Prentice Hall, 1995.

Algoritmos gulosos

Algoritmo guloso

- procura ótimo local e acaba obtendo ótimo global

Algoritmos gulosos

Algoritmo guloso

- procura ótimo local e acaba obtendo ótimo global

Costuma ser

- muito simples e intuitivo
- muito eficiente
- difícil de provar que está correto

Algoritmos gulosos

Algoritmo guloso

- procura ótimo local e acaba obtendo ótimo global

Costuma ser

- muito simples e intuitivo
- muito eficiente
- difícil de provar que está correto

Problema precisa ter

- subestrutura ótima (como na programação dinâmica)
- propriedade da escolha gulosa (*greedy-choice property*)

Problema dos intervalos disjuntos

Problema: Dados intervalos $[s[1], f[1]), \ldots, [s[n], f[n])$, encontrar uma coleção máxima de intervalos dois a dois disjuntos.

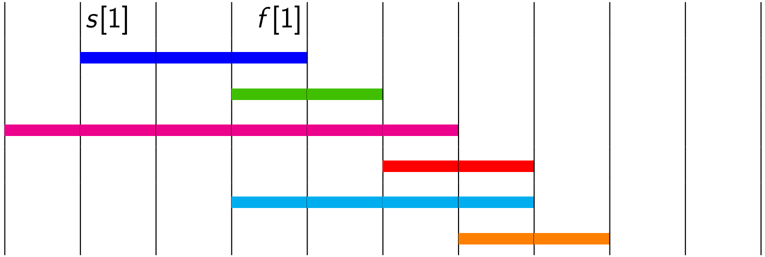
Solução é um subconjunto A de $\{1, \ldots, n\}$.

Problema dos intervalos disjuntos

Problema: Dados intervalos $[s[1], f[1]), \ldots, [s[n], f[n])$,
encontrar uma coleção máxima de intervalos dois a dois disjuntos.

Solução é um subconjunto A de $\{1, \ldots, n\}$.

Exemplo:

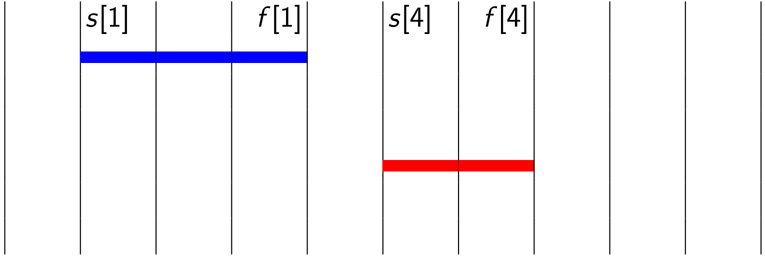


Problema dos intervalos disjuntos

Problema: Dados intervalos $[s[1], f[1]), \ldots, [s[n], f[n])$, encontrar uma coleção máxima de intervalos dois a dois disjuntos.

Solução é um subconjunto A de $\{1, \ldots, n\}$.

Solução:



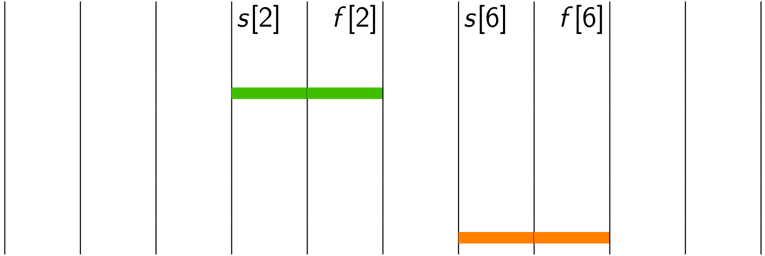
$A = \{1, 4\}$

Problema dos intervalos disjuntos

Problema: Dados intervalos $[s[1], f[1]), \ldots, [s[n], f[n])$, encontrar uma coleção máxima de intervalos dois a dois disjuntos.

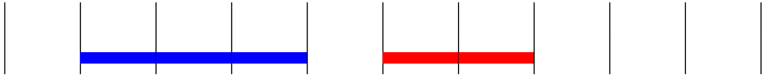
Solução é um subconjunto A de $\{1, \ldots, n\}$.

Solução:



$A = \{2, 6\}$

Motivação



Se cada intervalo é uma "atividade",
queremos coleção disjunta máxima de atividades compatíveis
(i e j com $s[i] \leq s[j]$ são compatíveis se $f[i] \leq s[j]$).

Nome no CLRS: Activity Selection Problem

Algoritmo guloso

GULOSO (s, f, n)

$A \leftarrow \emptyset$

$R \leftarrow \{1, \ldots, n\}$

enquanto $R \neq \emptyset$ **faça**

  escolha por um critério guloso um intervalo i de R

  $A \leftarrow A \cup \{i\}$

  $R \leftarrow R \setminus \{j \in R : j \text{ intersecta } i\}$

devolva A

Algoritmo guloso

GULOSO (s, f, n)

$A \leftarrow \emptyset$

$R \leftarrow \{1, \dots, n\}$

enquanto $R \neq \emptyset$ **faça**

  escolha por um critério guloso um intervalo i de R

  $A \leftarrow A \cup \{i\}$

  $R \leftarrow R \setminus \{j \in R : j \text{ intersecta } i\}$

devolva A

Invariante: todo intervalo de R é compatível com os de A.

Algoritmo guloso

GULOSO (s, f, n)

$A \leftarrow \emptyset$

$R \leftarrow \{1, \ldots, n\}$

enquanto $R \neq \emptyset$ **faça**

  escolha por um critério guloso um intervalo i de R

  $A \leftarrow A \cup \{i\}$

  $R \leftarrow R \setminus \{j \in R : j \text{ intersecta } i\}$

devolva A

Invariante: todo intervalo de R é compatível com os de A.

Claro que A é uma coleção de intervalos compatíveis.

Algoritmo guloso

GULOSO (s, f, n)
$A \leftarrow \emptyset$
$R \leftarrow \{1, \ldots, n\}$
enquanto $R \neq \emptyset$ **faça**
  escolha por um critério guloso um intervalo i de R
  $A \leftarrow A \cup \{i\}$
  $R \leftarrow R \setminus \{j \in R : j \text{ intersecta } i\}$
devolva A

Invariante: todo intervalo de R é compatível com os de A.

Claro que A é uma coleção de intervalos compatíveis.

Qual critério usamos para escolher um intervalo de R?

Possíveis critérios gulosos

- escolher o intervalo i em R com menor $s[i]$;

Funciona?

Possíveis critérios gulosos

- escolher o intervalo i em R com menor $s[i]$;
- escolher o intervalo i em R com menor $f[i] - s[i]$;

Funciona?

Possíveis critérios gulosos

- escolher o intervalo i em R com menor $s[i]$;
- escolher o intervalo i em R com menor $f[i] - s[i]$;
- escolher o intervalo i tal que

  $$|\{j \in R : j \text{ intersecta } \; i\}|$$

  é o menor possível;

Funciona?

Possíveis critérios gulosos

- escolher o intervalo i em R com menor $s[i]$;
- escolher o intervalo i em R com menor $f[i] - s[i]$;
- escolher o intervalo i tal que

  $$|\{j \in R : j \text{ intersecta } i\}|$$

  é o menor possível;
- escolher o intervalo i em R com menor $f[i]$;

Funciona?

Possíveis critérios gulosos

- escolher o intervalo i em R com menor $s[i]$;
- escolher o intervalo i em R com menor $f[i] - s[i]$;
- escolher o intervalo i tal que

  $$|\{j \in R : j \text{ intersecta } \ i\}|$$

  é o menor possível;
- escolher o intervalo i em R com menor $f[i]$;

Funciona?

Sim! Propriedade da escolha gulosa!

Algoritmo guloso

GULOSO (s, f, n)

ORDENE(s, f, n) $\qquad \triangleright\ f[1] \leq f[2] \leq \cdots \leq f[n]$

$A \leftarrow \emptyset$

$f_A \leftarrow 0$ $\qquad \triangleright$ fim do último intervalo em A

para $i \leftarrow 1$ **até** n **faça**

  se $s[i] \geq f_A$ $\qquad \triangleright$ intervalo i é compatível com A?

    então $A \leftarrow A \cup \{i\}$

      $f_A \leftarrow f[i]$

devolva A

Algoritmo guloso

GULOSO (s, f, n)

ORDENE(s, f, n) $\quad \triangleright\ f[1] \leq f[2] \leq \cdots \leq f[n]$

$A \leftarrow \emptyset$

$f_A \leftarrow 0$ $\quad \triangleright$ fim do último intervalo em A

para $i \leftarrow 1$ **até** n **faça**

  se $s[i] \geq f_A$ $\quad \triangleright$ intervalo i é compatível com A?

    então $A \leftarrow A \cup \{i\}$

      $f_A \leftarrow f[i]$

devolva A

Consumo de tempo: $\mathrm{O}(n \lg n)$

Algoritmo guloso

GULOSO (s, f, n)

ORDENE(s, f, n) $\triangleright$ $f[1] \leq f[2] \leq \cdots \leq f[n]$

$A \leftarrow \emptyset$

$f_A \leftarrow 0$ $\triangleright$ fim do último intervalo em A

para $i \leftarrow 1$ **até** n **faça**

  se $s[i] \geq f_A$ $\triangleright$ intervalo i é compatível com A?

    então $A \leftarrow A \cup \{i\}$

      $f_A \leftarrow f[i]$

devolva A

Consumo de tempo: $\mathrm{O}(n \lg n)$

Formalizando...

Por que A é coleção máxima de intervalos compatíveis?

Subestrutura ótima

Intervalos $S := \{1, \ldots, n\}$

Suponha que

A é coleção máxima de intervalos disjuntos de S.

Subestrutura ótima

Intervalos $S := \{1, \ldots, n\}$

Suponha que

A é coleção máxima de intervalos disjuntos de S.

Demonstre a seguinte propriedade:

Se $i \notin A$

então A é coleção máxima de intervalos disjuntos de $S \setminus \{i\}$.

Subestrutura ótima

Intervalos $S := \{1, \ldots, n\}$

Suponha que

A é coleção máxima de intervalos disjuntos de S.

Demonstre a seguinte propriedade:

Se $i \notin A$

então A é coleção máxima de intervalos disjuntos de $S \setminus \{i\}$.

senão $A \setminus \{i\}$ é coleção máxima de intervalos disjuntos

de $S \setminus \{k : [s[k], f[k]) \cap [s[i], f[i]) \neq \emptyset\}$.

Subestrutura ótima II

Intervalos $S := \{1, \ldots, n\}$

Suponha que

A é coleção máxima de intervalos disjunto de S.

Demonstre a seguinte propriedade:

Se $i \in A$ é tal que $f[i]$ é mínimo $\triangleright$ intervalo i é o mais à esquerda de A

Subestrutura ótima II

Intervalos $S := \{1, \ldots, n\}$

Suponha que

A é coleção máxima de intervalos disjunto de S.

Demonstre a seguinte propriedade:

Se $i \in A$ é tal que $f[i]$ é mínimo $\triangleright$ intervalo i é o mais à esquerda de A

então $A - \{i\}$ é coleção máxima de intervalos disjuntos

de $\{k : s[k] \geq f[i]\}$. $\triangleright$ intervalos compatíveis com i

$\{k : s[k] \geq f[i]\}$ = todos intervalos "à direita" de i.

Algoritmo guloso

GULOSO (s, f, n)

ORDENE(s, f, n) $\quad \triangleright\ f[1] \leq f[2] \leq \cdots \leq f[n]$

$A \leftarrow \emptyset$

$f_A \leftarrow 0$ $\quad \triangleright$ fim do último intervalo em A

para $i \leftarrow 1$ **até** n **faça**

  se $s[i] \geq f_A$ $\quad \triangleright$ intervalo i é compatível com A?

    então $A \leftarrow A \cup \{i\}$

      $f_A \leftarrow f[i]$

devolva A

Consumo de tempo: $\mathrm{O}(n \lg n)$

Do que argumentamos, vale a propriedade da escolha gulosa, logo o algoritmo devolve uma coleção máxima de intervalos disjuntos.

Coloração de intervalos

Dados intervalos $[s[1], f[1]), \ldots, [s[n], f[n])$ e um inteiro positivo k, uma *k-coloração* dos intervalos é uma partição deles em k coleções de intervalos disjuntos.

Coloração de intervalos

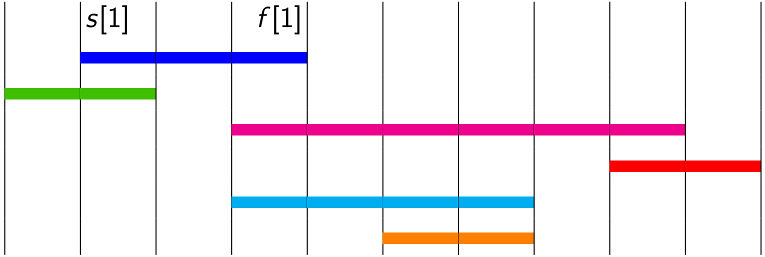
Dados intervalos $[s[1], f[1]), \ldots, [s[n], f[n])$ e um inteiro positivo k, uma k-coloração dos intervalos é uma partição deles em k coleções de intervalos disjuntos.

Problema: Dados intervalos $[s[1], f[1]), \ldots, [s[n], f[n])$, encontrar uma k-coloração dos intervalos com k o menor possível.

Coloração de intervalos

Dados intervalos $[s[1], f[1]), \ldots, [s[n], f[n])$ e um inteiro positivo k, uma *k*-coloração dos intervalos é uma partição deles em k coleções de intervalos disjuntos.

Problema: Dados intervalos $[s[1], f[1]), \ldots, [s[n], f[n])$, encontrar uma k-coloração dos intervalos com k o menor possível.

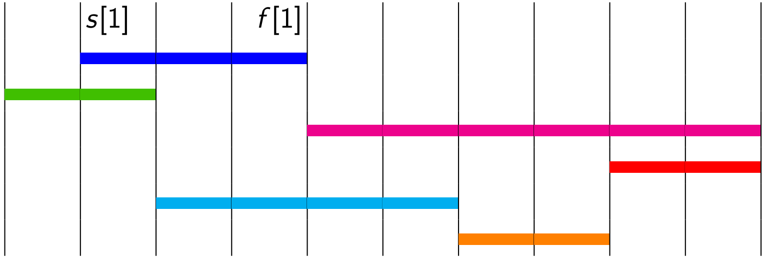


Quantas cores precisa?

Coloração de intervalos

Dados intervalos $[s[1], f[1]), \ldots, [s[n], f[n])$ e um inteiro positivo k, uma k-coloração dos intervalos é uma partição deles em k coleções de intervalos disjuntos.

Problema: Dados intervalos $[s[1], f[1]), \ldots, [s[n], f[n])$, encontrar uma k-coloração dos intervalos com k o menor possível.

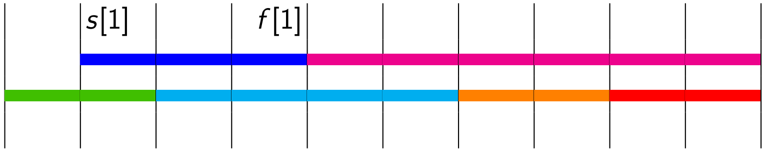


Quantas cores precisa?

Coloração de intervalos

Dados intervalos $[s[1], f[1]), \ldots, [s[n], f[n])$ e um inteiro positivo k, uma *k*-coloração dos intervalos é uma partição deles em k coleções de intervalos disjuntos.

Problema: Dados intervalos $[s[1], f[1]), \ldots, [s[n], f[n])$, encontrar uma k-coloração dos intervalos com k o menor possível.

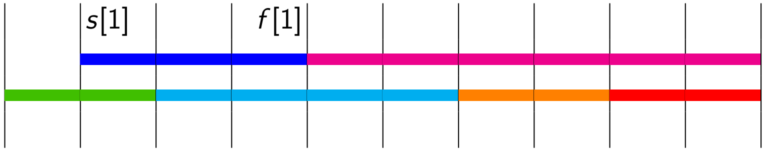


Solução: 2-coloração.

Coloração de intervalos

Dados intervalos $[s[1], f[1]), \ldots, [s[n], f[n])$ e um inteiro positivo k, uma *k*-coloração dos intervalos é uma partição deles em k coleções de intervalos disjuntos.

Problema: Dados intervalos $[s[1], f[1]), \ldots, [s[n], f[n])$, encontrar uma k-coloração dos intervalos com k o menor possível.



Solução: 2-coloração.

Como resolver esse problema?

Ideias?

Dados intervalos $[s[1], f[1]), \ldots, [s[n], f[n])$ e um inteiro positivo k, uma k-coloração dos intervalos é uma partição deles em k coleções de intervalos disjuntos.

Problema: Dados intervalos $[s[1], f[1]), \ldots, [s[n], f[n])$, encontrar uma k-coloração dos intervalos com k o menor possível.

Como resolver esse problema?

Ideias?

Dados intervalos $[s[1], f[1]), \ldots, [s[n], f[n])$ e um inteiro positivo k, uma k-coloração dos intervalos é uma partição deles em k coleções de intervalos disjuntos.

Problema: Dados intervalos $[s[1], f[1]), \ldots, [s[n], f[n])$, encontrar uma k-coloração dos intervalos com k o menor possível.

Como resolver esse problema?

Encontra uma coleção máxima de intervalos disjuntos.

Pinta de uma cor, remove e repete.

Isso funciona?

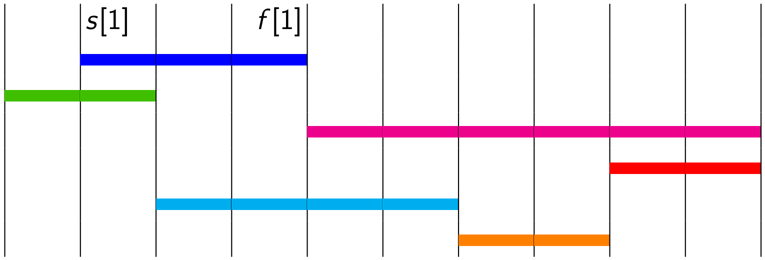
Ideias?

Encontra uma coleção máxima de intervalos disjuntos.

(ordena por f e vai escolhendo)

Pinta de uma cor, remove e repete.

Funciona nesse exemplo?



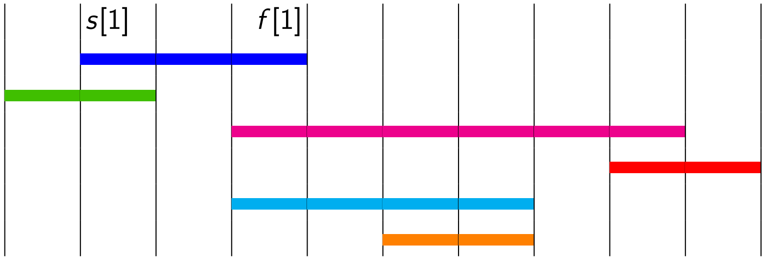
Ideias?

Encontra uma coleção máxima de intervalos disjuntos.

(ordena por f e vai escolhendo)

Pinta de uma cor, remove e repete.

Funciona nesse exemplo?



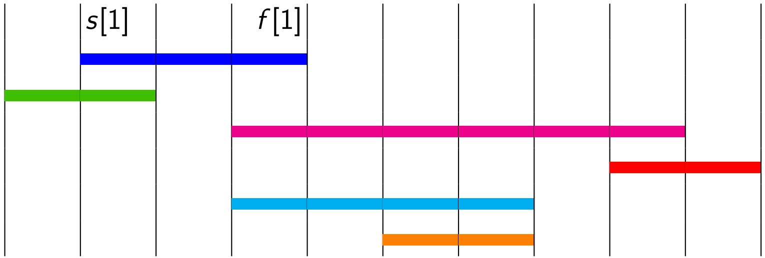
Ideias?

Encontra uma coleção máxima de intervalos disjuntos.

(ordena por f e vai escolhendo)

Pinta de uma cor, remove e repete.

Funciona nesse exemplo?



Dá para pintar com menos que 3 cores?

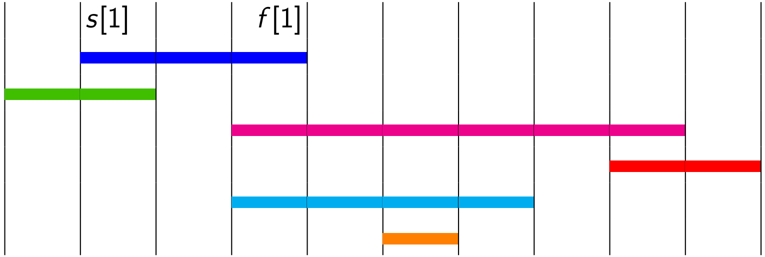
Ideias?

Encontra uma coleção máxima de intervalos disjuntos.

(ordena por f e vai escolhendo)

Pinta de uma cor, remove e repete.

Funciona nesse exemplo?



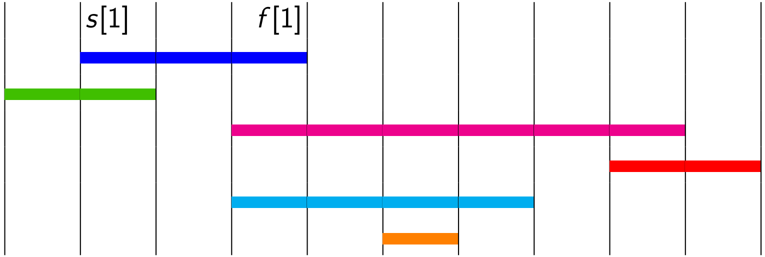
Ideias?

Encontra uma coleção máxima de intervalos disjuntos.

(ordena por f e vai escolhendo)

Pinta de uma cor, remove e repete.

Funciona nesse exemplo?



Não funciona... :-(

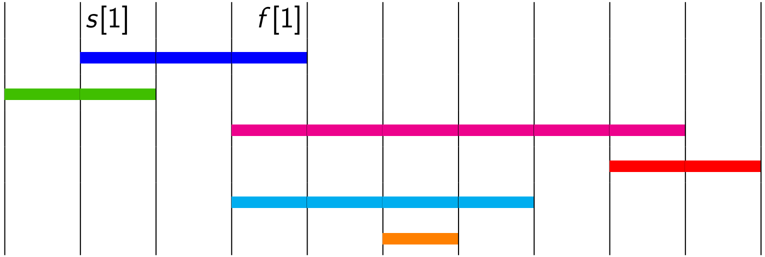
Ideias?

Encontra uma coleção máxima de intervalos disjuntos.

(ordena por f e vai escolhendo)

Pinta de uma cor, remove e repete.

Funciona nesse exemplo?



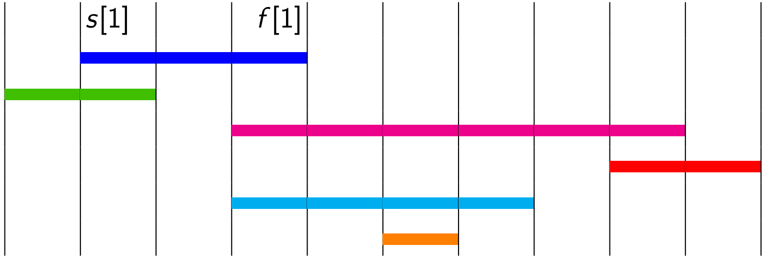
Não funciona... :-(

Essa ideia equivale a ordenar pelo f e pintar com a menor cor possível.

Ideias?

E ordenar pelo s e pintar com a menor cor possível?

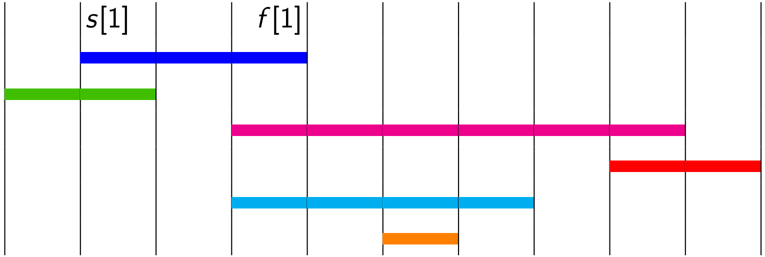
Funciona?



Ideias?

E ordenar pelo s e pintar com a menor cor possível?

Funciona?



Funciona! Produz um certificado!

Algoritmo guloso

```
GULOSO (s, f, n)
    ORDENE(s, f, n)                      ▷ s[1] ≤ s[2] ≤ ··· ≤ s[n]
    m ← 0                                ▷ número de cores usadas
    ℓ[1] ← 0                             ▷ término corrente daquela cor
    para k ← 1 até n faça
        j ← 1                            ▷ procura menor cor compatível para k
        enquanto ℓ[j] > s[k] faça
            j ← j + 1
        se j > m                         ▷ precisa de mais uma cor?
            então m ← j
                  ℓ[m + 1] ← 0
        ℓ[j] ← f[k]
        c[k] ← j
    devolva c
```

Exercícios

1. Mostre um exemplo para os três primeiros critérios gulosos apresentados para o primeiro problema que prove que o algoritmo obtido usando estes critérios pode produzir um conjunto A que não é máximo.

2. Considere o algoritmo do slide anterior para o segundo problema. Modifique-o para que, além de c, ele devolva um conjunto S de m intervalos e um instante t tal que $s[i] \leq t < f[i]$ para todo i em S.

3. Se, no algoritmo do slide anterior, os intervalos forem ordenados pelo valor de f em vez de pelo valor de s, isso faz diferença? Em particular, essa modificação também dá uma resposta correta? Prove ou dê um contra-exemplo.