Algoritmos gulosos (*greedy*)

CLRS 16.1 e mais...

Algoritmos gulosos

Algoritmo guloso

- procura ótimo local e acaba obtendo ótimo global

costuma ser

- muito simples e intuitivo
- muito eficiente
- difícil 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 disjuntos dois a dois.

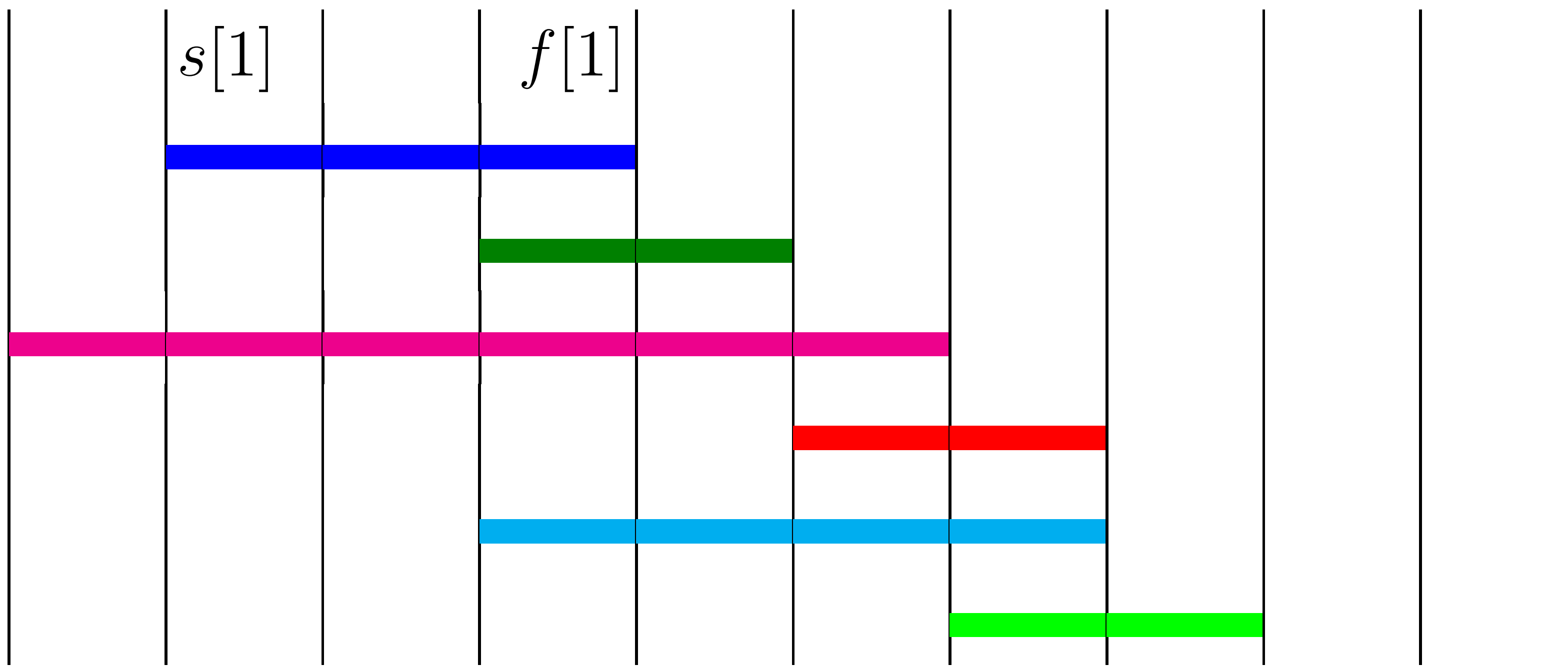
Solução é um subconjunto A de $\{1, \ldots, n\}$.

Problema dos intervalos disjuntos

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

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

Exemplo:

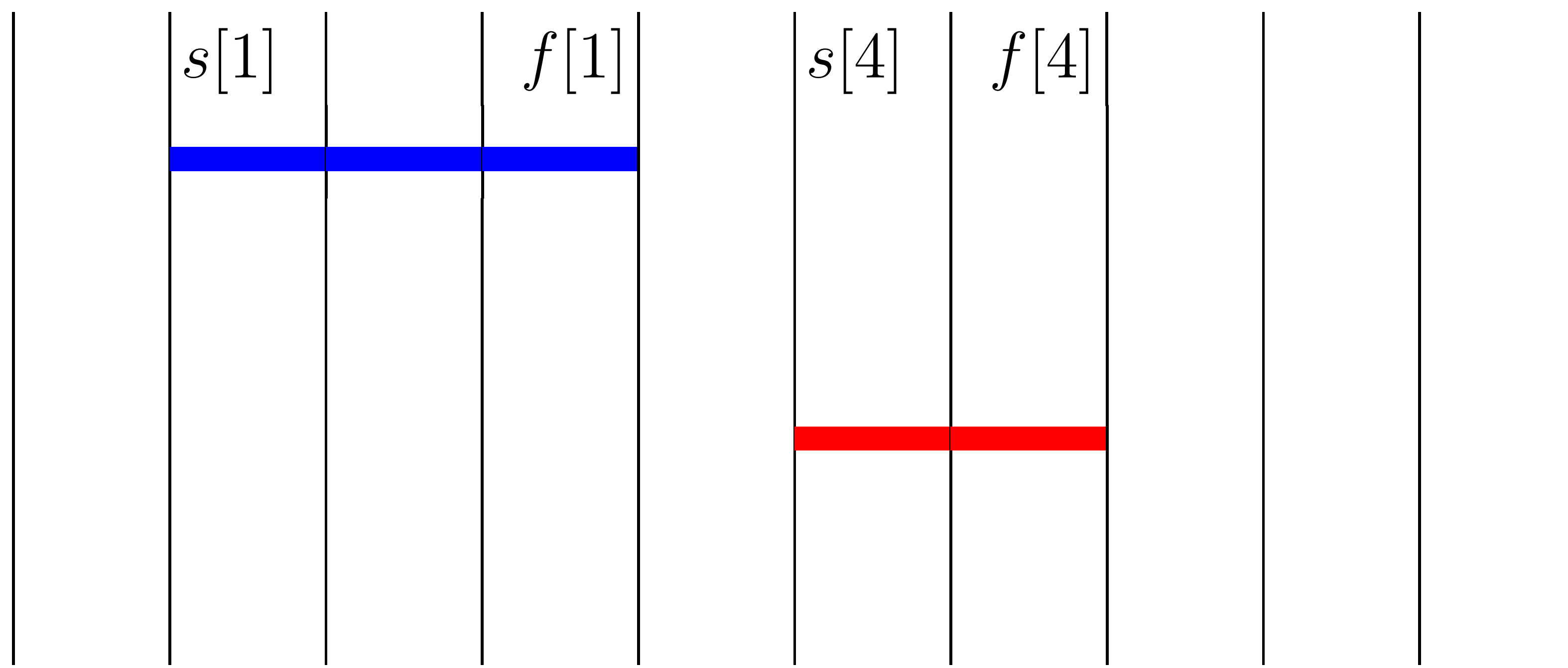


Problema dos intervalos disjuntos

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

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

Solução:

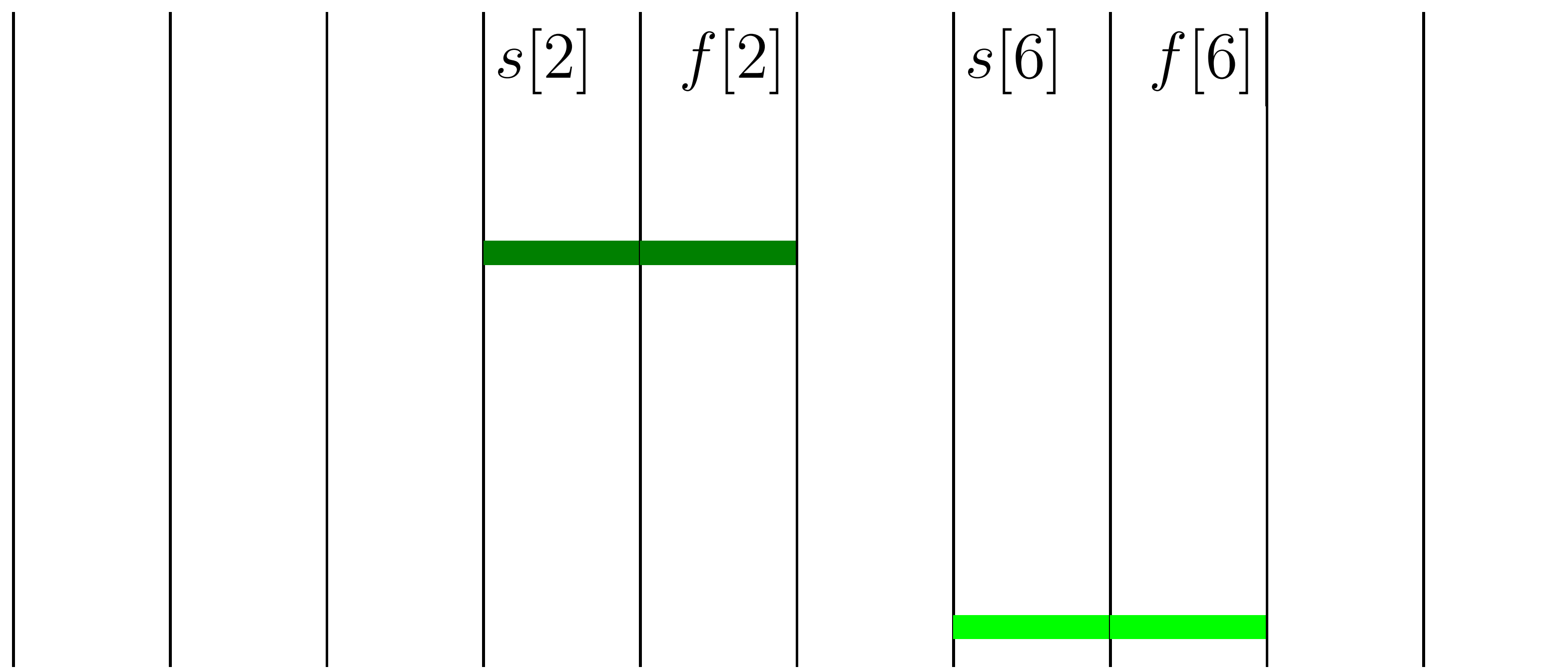


Problema dos intervalos disjuntos

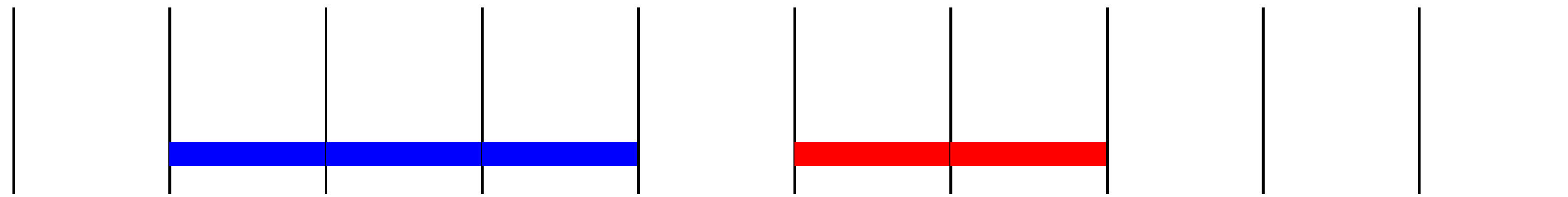
Problema: Dados intervalos $[s[1], f[1]), \ldots, [s[n], f[n])$, encontrar uma coleção máxima de intervalos disjuntos dois a dois.

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

Solução:



Motivação



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

Nome no CLRS: Activity Selection Problem

Subestrutura ótima

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

Suponha que A é coleção máxima de intervalos de S disjuntos dois a dois.

Se $i \in A$

então $A - \{i\}$ é coleção máxima de intervalos disjuntos de $S - \{k : [s[k], f[k]) \cap [s[i], f[i]) \neq \emptyset\}$.

senão A é coleção máxima de intervalos disjuntos de $S - \{i\}$.

Demonstre a propriedade.

Subestrutura ótima II

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

Suponha que A é coleção máxima de intervalos de S disjuntos dois a dois.

Se $i \in A$ é tal que $f[i]$ é mínimo

então $A - \{i\}$ é coleção máxima de intervalos disjuntos de $\{k : s[k] \geq f[i]\}$.

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

Demonstre a propriedade.

Algoritmo de programação dinâmica

Suponha $s[1] \leq s[2] \leq \cdots \leq s[n]$

$t[i] =$ tamanho de uma subcoleção disjunta máxima de $\{i, \ldots, n\}$

$$t[n] = 1$$

$$t[i] = \max\{t[i+1], 1 + t[k]\} \quad \text{para } i = 1, \ldots, n-1,$$

onde k é o menor índice tal que $s[k] \geq f[i]$.

Algoritmo de programação dinâmica

DYNAMIC-ACTIVITY-SELECTOR (s, f, n)

ordene s e f de tal forma que

$$s[1] \leq s[2] \leq \cdots \leq s[n]$$

$A[n+1] \leftarrow \emptyset$

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

  $A[i] \leftarrow A[i+1]$

  $k \leftarrow i + 1$

  enquanto $k \leq n$ e $s[k] < f[i]$ **faça**

    $k \leftarrow k + 1$

  se $|A[i]| < 1 + |A[k]|$

    então $A[i] \leftarrow \{i\} \cup A[k]$

devolva $A[1]$

Consumo de tempo é $\Theta(n^2)$.

Conclusão

Invariante: na linha 2 vale que

(i0) $A[k]$ é coleção disjunta máxima de $\{k, \ldots, n\}$ para $k = i + 1, \ldots, n$.

O consumo de tempo do algoritmo DYNAMIC-ACTIVITY-SELECTOR é $\Theta(n^2)$.

Escolha gulosa

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

Se $f[i]$ é mínimo em S,

então EXISTE uma solução ótima A tal que $i \in A$.

Demonstre a propriedade.

Algoritmo guloso

Devolve uma coleção máxima de intervalos disjuntos dois a dois.

INTERVALOS-DISJUNTOS (s, f, n)

0 ordene s e f de tal forma que

$$f[1] \leq f[2] \leq \cdots \leq f[n]$$

1 $A \leftarrow \{1\}$
2 $i \leftarrow 1$
3 **para** $j \leftarrow 2$ **até** n **faça**
4 **se** $s[j] \geq f[i]$
5 **então** $A \leftarrow A \cup \{j\}$
6 $i \leftarrow j$
7 **devolva** A

Consumo de tempo da linha 0 é $\Theta(n \lg n)$.
Consumo de tempo das linhas 1–7 é $\Theta(n)$.

Conclusão

Na linha 3 vale que

(i0) A é uma coleção máxima de intervalos disjuntos de $(s, f, j-1)$

$\boxed{\text{O consumo de tempo do algoritmo INTERVALOS-DISJUNTOS é } \Theta(n \lg n).}$

Coloração de intervalos

Problema: Dados intervalos de tempo $[s_1, f_2), \ldots, [s_n, f_n)$, encontrar uma coloração dos intervalos com o menor número possível de cores em que dois intervalos de mesma cor sempre sejam disjuntos.

Solução: uma partição de $\{1, \ldots, n\}$ em coleções de intervalos dois a dois disjuntos.

Motivação

Queremos distribuir um conjunto de atividades no menor número possivel de salas.

Cada atividade a_i ocupa um certo intervalo de tempo $[s_i, f_i)$ e duas atividades podem ser programadas para a mesma sala somente se os correspondentes intervalos são disjuntos.

Cada sala corresponde a uma cor. Queremos usar o menor número possível de cores para pintar todos os intervalos.

Coloração de intervalos

Estratégias gulosas:

- Encontre uma coleção disjunta máxima de intervalos, pinte com a próxima cor disponível e repita a idéia para os intervalos restantes.
- Ordene as atividades de maneira que $f[1] \leq f[2] \leq \cdots \leq f[n]$ e pinte uma a uma nesta ordem sempre usando a menor cor possível para aquela atividade.
- Ordene as atividades de maneira que $s[1] \leq s[2] \leq \cdots \leq s[n]$ e pinte uma a uma nesta ordem sempre usando a menor cor possível para aquela atividade.

Quais destas estratégias funcionam?
Quais não funcionam?