Análise de Algoritmos

Estes slides são adaptações de slides

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

Mais um problema de escalonamento

d_i: deadline da tarefa i

t_i: tempo de processamento da tarefa i

Escalonamento: permutação π, onde
$\pi(i)$ denota a posição da tarefa i na ordem de execução.

Mais um problema de escalonamento

d_i: deadline da tarefa i

t_i: tempo de processamento da tarefa i

Escalonamento: permutação π, onde
$\pi(i)$ denota a posição da tarefa i na ordem de execução.

Dado π, o instante de início da tarefa i é

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

ou seja, é a soma das durações das tarefas anteriores a i.

Mais um problema de escalonamento

d_i: deadline da tarefa i

t_i: tempo de processamento da tarefa i

Escalonamento: permutação π, onde
$\pi(i)$ denota a posição da tarefa i na ordem de execução.

Dado π, o instante de início da tarefa i é

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

ou seja, é a soma das durações das tarefas anteriores a i.

Instante de término: $f_i = s_i + t_i$.

Mais um problema de escalonamento

d_i: deadline da tarefa i

t_i: tempo de processamento da tarefa i

Escalonamento: permutação π, onde
$\pi(i)$ denota a posição da tarefa i na ordem de execução.

Dado π, o instante de início da tarefa i é

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

ou seja, é a soma das durações das tarefas anteriores a i.

Instante de término: $f_i = s_i + t_i$.

Atraso ℓ_i: 0 se $f_i \leq d_i$ e $d_i - f_i$ c.c.

Mais um problema de escalonamento

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

Mais um problema de escalonamento

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.

Mais um problema de escalonamento

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

Mais um problema de escalonamento

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: $t_1 = 1$ 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: $\ell_1 = \ell_2 = \ell_3 = 0$.

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: $d_1 = 10$ 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: $d_1 = 10$ 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: $d_1 = 10$ 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 $t_1 = 1$, $d_2 = 10$ e $t_2 = 10$.

Possíveis critérios gulosos

- SPT - shortest processing time (menor t_i primeiro)

Funciona?

Não...
Exemplo: $d_1 = 10$ 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 $t_1 = 1$, $d_2 = 10$ e $t_2 = 10$.

- EDD - earliest due date (menor d_i primeiro)

Funciona?

Algoritmo resultante

GULOSO (d, t, n)

1 seja π permutação tq $d[\pi^{-1}(1)] \leq \cdots \leq d[\pi^{-1}(n)]$

2 **devolva** π

Nem olhamos o t... Será que isso funciona???

Algoritmo resultante

GULOSO (d, t, n)

1 seja π permutação tq $d[\pi^{-1}(1)] \leq \cdots \leq d[\pi^{-1}(n)]$

2 **devolva** π

Nem olhamos o t... Será que isso funciona???

Note que não há tempo ocioso em uma solução.

Dado um escalonamento, uma inversão é um par de índices i e j tais que $\pi(i) < \pi(j)$ e $d_i > d_j$.

Algoritmo resultante

GULOSO (d, t, n)

1 seja π permutação tq $d[\pi^{-1}(1)] \leq \cdots \leq d[\pi^{-1}(n)]$

2 **devolva** π

Nem olhamos o t... Será que isso funciona???

Note que não há tempo ocioso em uma solução.

Dado um escalonamento, uma inversão é um par de índices i e j tais que $\pi(i) < \pi(j)$ e $d_i > d_j$.

Afirmação 1: Dois escalonamentos sem inversão têm o mesmo atraso máximo.

Algoritmo resultante

GULOSO (d, t, n)

1 seja π permutação tq $d[\pi^{-1}(1)] \leq \cdots \leq d[\pi^{-1}(n)]$

2 **devolva** π

Nem olhamos o t... Será que isso funciona???

Note que não há tempo ocioso em uma solução.

Dado um escalonamento, uma inversão é um par de índices i e j tais que $\pi(i) < \pi(j)$ e $d_i > d_j$.

Afirmação 1: Dois escalonamentos sem inversão têm o mesmo atraso máximo.

Prova feita na aula.

Algoritmo resultante

GULOSO (d, t, n)

1 seja π permutação tq $d[\pi^{-1}(1)] \leq \cdots \leq d[\pi^{-1}(n)]$

2 **devolva** π

Nem olhamos o t... Será que isso funciona???

Note que não há tempo ocioso em uma solução.

Dado um escalonamento, uma inversão é um par de índices i e j tais que $\pi(i) < \pi(j)$ e $d_i > d_j$.

Afirmação 1: Dois escalonamentos sem inversão têm o mesmo atraso máximo.

Afirmação 2: Existe uma solução que não tem nenhuma inversão.

Provas feitas na aula.