Algoritmos de Aproximação

Segundo Semestre de 2012

Escalonamento de uma máquina

Dados: n tarefas $([n] = \{1, \ldots, n\})$
duração $d[i]$ da tarefa i $(i = 1, \ldots, n)$
momento de chegada $r[i]$ da tarefa i $(i = 1, \ldots, n)$

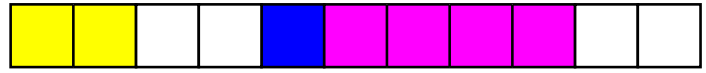
um escalonamento é uma ordenação de $[n]$.

Escalonamento de uma máquina

Dados: n tarefas $([n] = \{1, \ldots, n\})$
duração $d[i]$ da tarefa i $(i = 1, \ldots, n)$
momento de chegada $r[i]$ da tarefa i $(i = 1, \ldots, n)$

um escalonamento é uma ordenação de $[n]$.

Exemplo: $d_1 = 2$, $r_1 = 0$, $d_2 = 1$, $r_2 = 4$, $d_3 = 4$, $r_3 = 1$.

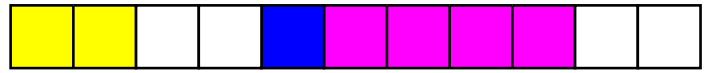


Escalonamento de uma máquina

Dados: n tarefas $([n] = \{1, \ldots, n\})$
duração $d[i]$ da tarefa i $(i = 1, \ldots, n)$
momento de chegada $r[i]$ da tarefa i $(i = 1, \ldots, n)$

um escalonamento é uma ordenação de $[n]$.

Exemplo: $d_1 = 2$, $r_1 = 0$, $d_2 = 1$, $r_2 = 4$, $d_3 = 4$, $r_3 = 1$.



$C_1 = 2$, $C_2 = 5$, $C_3 = 9$.

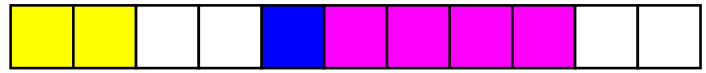
Encontrar escalonamento com
soma dos tempos de conclusão mínima.

Escalonamento de uma máquina

Dados: n tarefas $([n] = \{1, \ldots, n\})$
duração $d[i]$ da tarefa i $(i = 1, \ldots, n)$
momento de chegada $r[i]$ da tarefa i $(i = 1, \ldots, n)$

um escalonamento é uma ordenação de $[n]$.

Exemplo: $d_1 = 2$, $r_1 = 0$, $d_2 = 1$, $r_2 = 4$, $d_3 = 4$, $r_3 = 1$.



$C_1 = 2$, $C_2 = 5$, $C_3 = 9$.

Encontrar escalonamento com
soma dos tempos de conclusão mínima.

Mesmo que minimizar a média dos tempos de conclusão.

Versão com pesos

Dados: n tarefas $([n] = \{1, \ldots, n\})$
duração $d[i]$ da tarefa i $(i = 1, \ldots, n)$
momento de chegada $r[i]$ da tarefa i $(i = 1, \ldots, n)$
peso $w[i]$ da tarefa i $(i = 1, \ldots, n)$

Versão com pesos

Dados: n tarefas $([n] = \{1, \ldots, n\})$
duração $d[i]$ da tarefa i $(i = 1, \ldots, n)$
momento de chegada $r[i]$ da tarefa i $(i = 1, \ldots, n)$
peso $w[i]$ da tarefa i $(i = 1, \ldots, n)$

Um escalonamento é uma ordenação de $[n]$.

Encontrar escalonamento com
soma ponderada dos tempos de conclusão mínima.

Versão com pesos

Dados: n tarefas $([n] = \{1, \ldots, n\})$
duração $d[i]$ da tarefa i $(i = 1, \ldots, n)$
momento de chegada $r[i]$ da tarefa i $(i = 1, \ldots, n)$
peso $w[i]$ da tarefa i $(i = 1, \ldots, n)$

Um escalonamento é uma ordenação de $[n]$.

Encontrar escalonamento com
soma ponderada dos tempos de conclusão mínima.

Versão com interrupções com pesos é NP-difícil...

Versão com pesos

Dados: n tarefas $([n] = \{1, \ldots, n\})$
duração $d[i]$ da tarefa i $(i = 1, \ldots, n)$
momento de chegada $r[i]$ da tarefa i $(i = 1, \ldots, n)$
peso $w[i]$ da tarefa i $(i = 1, \ldots, n)$

Um escalonamento é uma ordenação de $[n]$.

Encontrar escalonamento com
soma ponderada dos tempos de conclusão mínima.

Versão com interrupções com pesos é NP-difícil...

Vimos uma 3-aproximação e vamos ver
uma 2-aproximação para esse problema.

Uma relaxação linear inteira

Seja $T = \max_j r[j] + \sum_{j=1}^{n} d[j]$.

Nenhuma tarefa pode terminar depois do instante T.

Uma relaxação linear inteira

Seja $T = \max_j r[j] + \sum_{j=1}^{n} d[j]$.

Nenhuma tarefa pode terminar depois do instante T.

Variáveis:

$y_{jt} = 1$ sse a tarefa j está executando no instante t.

Uma relaxação linear inteira

Seja $T = \max_j r[j] + \sum_{j=1}^{n} d[j]$.

Nenhuma tarefa pode terminar depois do instante T.

Variáveis:

$y_{jt} = 1$ sse a tarefa j está executando no instante t.

No máximo uma tarefa executa a cada instante t:

$$\sum_{j=1}^{n} y_{jt} \leq 1.$$

Uma relaxação linear inteira

Seja $T = \max_j r[j] + \sum_{j=1}^{n} d[j]$.

Nenhuma tarefa pode terminar depois do instante T.

Variáveis:
$y_{jt} = 1$ sse a tarefa j está executando no instante t.

No máximo uma tarefa executa a cada instante t:

$$\sum_{j=1}^{n} y_{jt} \leq 1.$$

A tarefa j executa por $d[j]$ unidades de tempo:

$$\sum_{t=1}^{T} y_{jt} = d[j].$$

Uma outra relaxação linear

Seja $T = \max_j r[j] + \sum_{j=1}^{n} d[j]$.

$y_{jt} = 1$ sse a tarefa j está executando no instante t.

No máximo uma tarefa executa a cada instante:

$$\sum_{j=1}^{n} y_{jt} \leq 1.$$

A tarefa j executa por $d[j]$ unidades de tempo:

$$\sum_{t=1}^{T} y_{jt} = d[j].$$

Uma outra relaxação linear

Seja $T = \max_j r[j] + \sum_{j=1}^{n} d[j]$.

$y_{jt} = 1$ sse a tarefa j está executando no instante t.

No máximo uma tarefa executa a cada instante:

$$\sum_{j=1}^{n} y_{jt} \;\leq\; 1.$$

A tarefa j executa por $d[j]$ unidades de tempo:

$$\sum_{t=1}^{T} y_{jt} \;=\; d[j].$$

A tarefa j só pode executar depois que estiver disponível:

$$y_{jt} \;=\; 0 \quad \text{para } t = 1, \ldots, r[j].$$

Tempo de conclusão

Num escalonamento sem interrupção,
se a tarefa j termina no instante D, então

Tempo de conclusão

Num escalonamento sem interrupção,
se a tarefa j termina no instante D, então

$$y_{jt} = 1 \quad \text{para } t = D - d[j] + 1, \ldots, D,$$

caso contrário, $y_{jt} = 0$.

Tempo de conclusão

Num escalonamento sem interrupção,
se a tarefa j termina no instante D, então

$$y_{jt} = 1 \quad \text{para } t = D - d[j] + 1, \ldots, D,$$

caso contrário, $y_{jt} = 0$.

A média de $t - \frac{1}{2}$ de cada instante satisfaz

$$\frac{1}{d[j]} \sum_{t=D-d[j]+1}^{D} \left(t - \frac{1}{2}\right) = D - \frac{d[j]}{2}.$$

Tempo de conclusão

Num escalonamento sem interrupção,
se a tarefa j termina no instante D, então

$$y_{jt} = 1 \quad \text{para } t = D - d[j] + 1, \ldots, D,$$

caso contrário, $y_{jt} = 0$.

A média de $t - \frac{1}{2}$ de cada instante satisfaz

$$\frac{1}{d[j]} \sum_{t=D-d[j]+1}^{D} \left(t - \frac{1}{2}\right) = D - \frac{d[j]}{2}.$$

Disso deduzimos que

$$C_j = \frac{1}{d[j]} \sum_{t=1}^{T} y_{jt}\left(t - \frac{1}{2}\right) + \frac{d[j]}{2}.$$

Relaxação linear inteira

Encontrar matriz y e vetor C que

minimizem $\sum_{j=1}^{n} w_j C_j$

sujeito a

$$\sum_{j=1}^{n} y_{jt} \leq 1 \qquad \text{para } t = 1, \ldots, T$$

$$\sum_{t=1}^{T} y_{jt} = d[j] \qquad \text{para } j = 1, \ldots, n$$

$$C_j = \frac{1}{d[j]} \sum_{t=1}^{T} y_{jt}\left(t - \tfrac{1}{2}\right) + \frac{d[j]}{2} \qquad \text{para } j = 1, \ldots, n$$

$$y_{jt} \in \{0, 1\} \qquad \text{para } j = 1, \ldots, n, \; t = 1, \ldots, T.$$

Relaxação linear inteira

Encontrar matriz y e vetor C que

minimizem $\sum_{j=1}^{n} w_j C_j$

sujeito a

$$\sum_{j=1}^{n} y_{jt} \leq 1 \qquad \text{para } t = 1, \ldots, T$$

$$\sum_{t=1}^{T} y_{jt} = d[j] \qquad \text{para } j = 1, \ldots, n$$

$$C_j = \frac{1}{d[j]} \sum_{t=1}^{T} y_{jt}\left(t - \frac{1}{2}\right) + \frac{d[j]}{2} \qquad \text{para } j = 1, \ldots, n$$

$$y_{jt} \in \{0, 1\} \qquad \text{para } j = 1, \ldots, n, \; t = 1, \ldots, T.$$

Todo y viável para esse programa corresponde a um escalonamento com interrupções.

Relaxação linear inteira

Encontrar matriz y e vetor C que

minimizem $\sum_{j=1}^{n} w_j C_j$

sujeito a

$$\sum_{j=1}^{n} y_{jt} \leq 1 \qquad \text{para } t = 1, \ldots, T$$

$$\sum_{t=1}^{T} y_{jt} = d[j] \qquad \text{para } j = 1, \ldots, n$$

$$C_j = \frac{1}{d[j]} \sum_{t=1}^{T} y_{jt}\left(t - \frac{1}{2}\right) + \frac{d[j]}{2} \qquad \text{para } j = 1, \ldots, n$$

$$y_{jt} \in \{0, 1\} \qquad \text{para } j = 1, \ldots, n, \; t = 1, \ldots, T.$$

Todo y viável para esse programa corresponde a um escalonamento com interrupções.

É uma relaxação pois...

Todo escalonamento sem interrupções corresponde a uma solução viável e...

Relaxação linear inteira

Encontrar matriz y e vetor C que

minimizem $\sum_{j=1}^{n} w_j C_j$

sujeito a

$$\sum_{j=1}^{n} y_{jt} \leq 1 \qquad \text{para } t = 1, \ldots, T$$

$$\sum_{t=1}^{T} y_{jt} = d[j] \qquad \text{para } j = 1, \ldots, n$$

$$C_j = \frac{1}{d[j]} \sum_{t=1}^{T} y_{jt}\left(t - \frac{1}{2}\right) + \frac{d[j]}{2} \qquad \text{para } j = 1, \ldots, n$$

$$y_{jt} \in \{0, 1\} \qquad \text{para } j = 1, \ldots, n, \; t = 1, \ldots, T.$$

Todo y viável para esse programa corresponde a um escalonamento com interrupções.

É uma relaxação pois...

Todo escalonamento sem interrupções corresponde a uma solução viável e...

o valor desta solução no programa e no problema é exatamente o mesmo.

2-aproximação

Suponha que temos solução ótima (y^*, C^*) da relaxação.

2-aproximação

Suponha que temos solução ótima (y^*, C^*) da relaxação.

X_j: variável aleatória que vale $t - \frac{1}{2}$ com probabilidade $\frac{y^*_{jt}}{d_j}$

2-aproximação

Suponha que temos solução ótima (y^*, C^*) da relaxação.

X_j: variável aleatória que vale $t - \frac{1}{2}$ com probabilidade $\frac{y^*_{jt}}{d_j}$

Note que $\sum_{t=1}^{T} \frac{y^*_{jt}}{d_j} = 1$.

2-aproximação

Suponha que temos solução ótima (y^*, C^*) da relaxação.

X_j: variável aleatória que vale $t - \frac{1}{2}$ com probabilidade $\frac{y^*_{jt}}{d_j}$

Note que $\sum_{t=1}^{T} \frac{y^*_{jt}}{d_j} = 1$.

Considere as tarefas ordenadas por $X_1 \leq X_2 \leq \cdots \leq X_n$,
e escalone-as nesta ordem, como no algoritmo anterior.

Este algoritmo é uma 2-aproximação!

Análise da razão de aproximação

Seja $\hat{C}_j$ o tempo de conclusão neste escalonamento.

Análise da razão de aproximação

Seja $\hat{C}_j$ o tempo de conclusão neste escalonamento.

Lema: $\mathrm{E}[\hat{C}_j \mid X_j = x] \leq d_j + 2x$.

Análise da razão de aproximação

Seja $\hat{C}_j$ o tempo de conclusão neste escalonamento.

Lema: $\mathrm{E}[\hat{C}_j \mid X_j = x] \leq d_j + 2x$.

Prova: Não há instante ocioso entre $\max_{k=1\,..\,j} r_k$ e $\hat{C}_j$.

Análise da razão de aproximação

Seja $\hat{C}_j$ o tempo de conclusão neste escalonamento.

Lema: $\mathrm{E}[\hat{C}_j \mid X_j = x] \leq d_j + 2x$.

Prova: Não há instante ocioso entre $\max_{k=1\,..\,j} r_k$ e $\hat{C}_j$.

Logo $\hat{C}_j \leq \max_{k=1\,..\,j} r_k + \sum_{k=1}^{j} d_k$.

Análise da razão de aproximação

Seja $\hat{C}_j$ o tempo de conclusão neste escalonamento.

Lema: $\mathrm{E}[\hat{C}_j \mid X_j = x] \leq d_j + 2x$.

Prova: Não há instante ocioso entre $\max_{k=1\,..\,j} r_k$ e $\hat{C}_j$.

Logo $\hat{C}_j \leq \max_{k=1\,..\,j} r_k + \sum_{k=1}^{j} d_k$.

Seja R o valor de $\max_{k=1\,..\,j} r_k$ e D o valor de $\sum_{k=1}^{j-1} d_k$.

Análise da razão de aproximação

Seja $\hat{C}_j$ o tempo de conclusão neste escalonamento.

Lema: $\mathrm{E}[\hat{C}_j \mid X_j = x] \leq d_j + 2x$.

Prova: Não há instante ocioso entre $\max_{k=1\,..\,j} r_k$ e $\hat{C}_j$.

Logo $\hat{C}_j \leq \max_{k=1\,..\,j} r_k + \sum_{k=1}^{j} d_k$.

Seja R o valor de $\max_{k=1\,..\,j} r_k$ e D o valor de $\sum_{k=1}^{j-1} d_k$.

Veja que $\hat{C}_j \leq d_j + R + D$.

Análise da razão de aproximação

Seja $\hat{C}_j$ o tempo de conclusão neste escalonamento.

Lema: $\mathrm{E}[\hat{C}_j \mid X_j = x] \leq d_j + 2x$.

Prova: Não há instante ocioso entre $\max_{k=1\,..\,j} r_k$ e $\hat{C}_j$.

Logo $\hat{C}_j \leq \max_{k=1\,..\,j} r_k + \sum_{k=1}^{j} d_k$.

Seja R o valor de $\max_{k=1\,..\,j} r_k$ e D o valor de $\sum_{k=1}^{j-1} d_k$.

Veja que $\hat{C}_j \leq d_j + R + D$.

Vamos mostrar que

- $R \leq x - \frac{1}{2}$ dado que $X_j = x$, e
- $\mathrm{E}[D \mid X_j = x] \leq x + \frac{1}{2}$.

Delimitação no R

Suponha que $X_j = x$.

Delimitação no R

Suponha que $X_j = x$.

Lembre-se que $y^*_{kt} = 0$ para $t \leq r_k$

Delimitação no R

Suponha que $X_j = x$.

Lembre-se que $y^*_{kt} = 0$ para $t \leq r_k$

Logo $X_k \geq r_k + \frac{1}{2}$.

Delimitação no R

Suponha que $X_j = x$.

Lembre-se que $y^*_{kt} = 0$ para $t \leq r_k$

Logo $X_k \geq r_k + \frac{1}{2}$.

Então

$$R \leq \max_{k:X_k \leq X_j} r_k \leq \max_{k:X_k \leq X_j} (X_k - \frac{1}{2}) \leq X_j - \frac{1}{2} = x - \frac{1}{2}.$$

Delimitação no D

$$\mathrm{E}[D \mid X_j = x] \;=\; \sum_{k:k\neq j} d_k \,\Pr[k \text{ vir antes de } j \mid X_j = x]$$

Delimitação no D

$$\begin{aligned}
\mathrm{E}[D \mid X_j = x] &= \sum_{k:k\neq j} d_k \Pr[k \text{ vir antes de } j \mid X_j = x] \\
&= \sum_{k:k\neq j} d_k \Pr[X_k \leq X_j \mid X_j = x]
\end{aligned}$$

Delimitação no D

$$
\begin{aligned}
\mathrm{E}[D \mid X_j = x] &= \sum_{k:k\neq j} d_k \, \Pr[k \text{ vir antes de } j \mid X_j = x] \\
&= \sum_{k:k\neq j} d_k \, \Pr[X_k \leq X_j \mid X_j = x] \\
&= \sum_{k:k\neq j} d_k \sum_{t=1}^{x+\frac{1}{2}} \Pr[X_k = t - \frac{1}{2}].
\end{aligned}
$$

Delimitação no D

$$\begin{aligned}
\mathrm{E}[D \mid X_j = x] &= \sum_{k:k\neq j} d_k \Pr[k \text{ vir antes de } j \mid X_j = x] \\
&= \sum_{k:k\neq j} d_k \Pr[X_k \leq X_j \mid X_j = x] \\
&= \sum_{k:k\neq j} d_k \sum_{t=1}^{x+\frac{1}{2}} \Pr[X_k = t - \frac{1}{2}].
\end{aligned}$$

Mas $\Pr[X_k = t - \frac{1}{2}] = y^*_{kt}/d_k$ e

Delimitação no D

$$\begin{aligned}
\mathrm{E}[D \mid X_j = x] &= \sum_{k:k\neq j} d_k \Pr[k \text{ vir antes de } j \mid X_j = x] \\
&= \sum_{k:k\neq j} d_k \Pr[X_k \leq X_j \mid X_j = x] \\
&= \sum_{k:k\neq j} d_k \sum_{t=1}^{x+\frac{1}{2}} \Pr[X_k = t - \frac{1}{2}].
\end{aligned}$$

Mas $\Pr[X_k = t - \frac{1}{2}] = y^*_{kt}/d_k$ e

$$\mathrm{E}[D \mid X_j = x] = \sum_{k:k\neq j} d_k \sum_{t=1}^{x+\frac{1}{2}} \frac{y^*_{kt}}{d_k} = \sum_{k:k\neq j} \sum_{t=1}^{x+\frac{1}{2}} y^*_{kt} = \sum_{t=1}^{x+\frac{1}{2}} \sum_{k:k\neq j} y^*_{kt}.$$

Delimitação no D

$$\mathrm{E}[D \mid X_j = x] = \sum_{k:k\neq j} d_k \sum_{t=1}^{x+\frac{1}{2}} \frac{y^*_{kt}}{d_k} = \sum_{k:k\neq j} \sum_{t=1}^{x+\frac{1}{2}} y^*_{kt} = \sum_{t=1}^{x+\frac{1}{2}} \sum_{k:k\neq j} y^*_{kt}.$$

Delimitação no D

$$\mathrm{E}[D \mid X_j = x] = \sum_{k:k\neq j} d_k \sum_{t=1}^{x+\frac{1}{2}} \frac{y^*_{kt}}{d_k} = \sum_{k:k\neq j} \sum_{t=1}^{x+\frac{1}{2}} y^*_{kt} = \sum_{t=1}^{x+\frac{1}{2}} \sum_{k:k\neq j} y^*_{kt}.$$

Mas $\sum_{k:k\neq j} y^*_{kt} \leq 1$ para todo t. Logo

Delimitação no D

$$\mathrm{E}[D \mid X_j = x] \;=\; \sum_{k:k\neq j} d_k \sum_{t=1}^{x+\frac{1}{2}} \frac{y^*_{kt}}{d_k} \;=\; \sum_{k:k\neq j} \sum_{t=1}^{x+\frac{1}{2}} y^*_{kt} \;=\; \sum_{t=1}^{x+\frac{1}{2}} \sum_{k:k\neq j} y^*_{kt}.$$

Mas $\sum_{k:k\neq j} y^*_{kt} \leq 1$ para todo t. Logo

$$\mathrm{E}[D \mid X_j = x] \;\leq\; \sum_{t=1}^{x+\frac{1}{2}} 1 \;=\; x + \frac{1}{2}.$$

Delimitação no D

$$\mathrm{E}[D \mid X_j = x] \;=\; \sum_{k:k\neq j} d_k \sum_{t=1}^{x+\frac{1}{2}} \frac{y^*_{kt}}{d_k} \;=\; \sum_{k:k\neq j} \sum_{t=1}^{x+\frac{1}{2}} y^*_{kt} \;=\; \sum_{t=1}^{x+\frac{1}{2}} \sum_{k:k\neq j} y^*_{kt}.$$

Mas $\sum_{k:k\neq j} y^*_{kt} \leq 1$ para todo t. Logo

$$\mathrm{E}[D \mid X_j = x] \;\leq\; \sum_{t=1}^{x+\frac{1}{2}} 1 \;=\; x + \frac{1}{2}.$$

E portanto o lema vale:

$$\mathrm{E}[\hat{C}_j \mid X_j = x] \;\leq\; d_j + (x - \frac{1}{2}) + (x + \frac{1}{2}) \;=\; d_j + 2x.$$

Conclusão da análise

Lema: $\mathrm{E}[\hat{C}_j \mid X_j = x] \leq d_j + 2x$.

Teorema: O algoritmo é uma 2-aproximação probabilística.

Conclusão da análise

Lema: $\mathrm{E}[\hat{C}_j \mid X_j = x] \leq d_j + 2x$.

Teorema: O algoritmo é uma 2-aproximação probabilística.

Prova:

$$\mathrm{E}[\hat{C}_j] = \sum_{t=1}^{T} \mathrm{E}[\hat{C}_j \mid X_j = t - \frac{1}{2}] \Pr[X_j = t - \frac{1}{2}]$$

Conclusão da análise

Lema: $\mathrm{E}[\hat{C}_j \mid X_j = x] \leq d_j + 2x$.

Teorema: O algoritmo é uma 2-aproximação probabilística.

Prova:

$$\begin{aligned}
\mathrm{E}[\hat{C}_j] &= \sum_{t=1}^{T} \mathrm{E}[\hat{C}_j \mid X_j = t - \frac{1}{2}] \Pr[X_j = t - \frac{1}{2}] \\
&\leq d_j + 2 \sum_{t=1}^{T} \left(t - \frac{1}{2}\right) \Pr[X_j = t - \frac{1}{2}]
\end{aligned}$$

Conclusão da análise

Lema: $\mathrm{E}[\hat{C}_j \mid X_j = x] \leq d_j + 2x$.

Teorema: O algoritmo é uma 2-aproximação probabilística.

Prova:

$$\begin{aligned}
\mathrm{E}[\hat{C}_j] &= \sum_{t=1}^{T} \mathrm{E}[\hat{C}_j \mid X_j = t - \frac{1}{2}] \Pr[X_j = t - \frac{1}{2}] \\
&\leq d_j + 2 \sum_{t=1}^{T} \left(t - \frac{1}{2}\right) \Pr[X_j = t - \frac{1}{2}] \\
&= d_j + 2 \sum_{t=1}^{T} \left(t - \frac{1}{2}\right) \frac{y^*_{jt}}{d_j}
\end{aligned}$$

Conclusão da análise

Lema: $\mathrm{E}[\hat{C}_j \mid X_j = x] \leq d_j + 2x$.

Teorema: O algoritmo é uma 2-aproximação probabilística.

Prova:

$$\begin{aligned}
\mathrm{E}[\hat{C}_j] &= \sum_{t=1}^{T} \mathrm{E}[\hat{C}_j \mid X_j = t - \frac{1}{2}] \Pr[X_j = t - \frac{1}{2}] \\
&= d_j + 2 \sum_{t=1}^{T} (t - \frac{1}{2}) \Pr[X_j = t - \frac{1}{2}] \\
&= d_j + 2 \sum_{t=1}^{T} (t - \frac{1}{2}) \frac{y^*_{jt}}{d_j} \\
&= 2 \left[\frac{d_j}{2} + \frac{1}{d_j} \sum_{t=1}^{T} (t - \frac{1}{2}) y^*_{jt}\right] = 2C^*_j.
\end{aligned}$$

Conclusão da análise

Lema: $\mathrm{E}[\hat{C}_j \mid X_j = x] \leq d_j + 2x$.

Teorema: O algoritmo é uma 2-aproximação probabilística.

Prova: Como $\mathrm{E}[\hat{C}_j] \leq 2C_j^*$, temos que

Conclusão da análise

Lema: $\mathrm{E}[\hat{C}_j \mid X_j = x] \leq d_j + 2x$.

Teorema: O algoritmo é uma 2-aproximação probabilística.

Prova: Como $\mathrm{E}[\hat{C}_j] \leq 2C_j^*$, temos que

$$\mathrm{E}[\sum_{j=1}^{n} w_j \hat{C}_j] = \sum_{j=1}^{n} w_j \mathrm{E}[\hat{C}_j]$$

Conclusão da análise

Lema: $\mathrm{E}[\hat{C}_j \mid X_j = x] \leq d_j + 2x$.

Teorema: O algoritmo é uma 2-aproximação probabilística.

Prova: Como $\mathrm{E}[\hat{C}_j] \leq 2C_j^*$, temos que

$$\begin{aligned} \mathrm{E}[\sum_{j=1}^{n} w_j \hat{C}_j] &= \sum_{j=1}^{n} w_j \mathrm{E}[\hat{C}_j] \\ &\leq 2 \sum_{j=1}^{n} w_j C_j^* \end{aligned}$$

Conclusão da análise

Lema: $\mathrm{E}[\hat{C}_j \mid X_j = x] \leq d_j + 2x$.

Teorema: O algoritmo é uma 2-aproximação probabilística.

Prova: Como $\mathrm{E}[\hat{C}_j] \leq 2C_j^*$, temos que

$$\begin{aligned}
\mathrm{E}[\sum_{j=1}^{n} w_j \hat{C}_j] &= \sum_{j=1}^{n} w_j \mathrm{E}[\hat{C}_j] \\
&\leq 2 \sum_{j=1}^{n} w_j C_j^* \\
&\leq 2\,\mathrm{OPT}.
\end{aligned}$$

Relaxação linear inteira

Como resolver o problema descrito por esse PLI?

Relaxação linear inteira

Como resolver o problema descrito por esse PLI?

Pelo método guloso!

Ordene as tarefas de modo que

$$\frac{w_1}{d_1} \geq \frac{w_2}{d_2} \geq \cdots \geq \frac{w_n}{d_n}.$$

Relaxação linear inteira

Como resolver o problema descrito por esse PLI?

Pelo método guloso!

Ordene as tarefas de modo que

$$\frac{w_1}{d_1} \geq \frac{w_2}{d_2} \geq \cdots \geq \frac{w_n}{d_n}.$$

Escalone sempre uma tarefa que está disponível, não foi completada ainda, e está na frente nesta ordem.

Relaxação linear inteira

Como resolver o problema descrito por esse PLI?

Pelo método guloso!

Ordene as tarefas de modo que

$$\frac{w_1}{d_1} \geq \frac{w_2}{d_2} \geq \cdots \geq \frac{w_n}{d_n}.$$

Escalone sempre uma tarefa que está disponível, não foi completada ainda, e está na frente nesta ordem.

A tarefa que está executando só é trocada caso tenha concluído, ou caso uma “melhor” ficou disponível.

Assim existem $2n$ eventos: os momentos r_j e os C_j.

Método guloso

Basta guardarmos no máximo $2n$ intervalos.

Método guloso

Basta guardarmos no máximo $2n$ intervalos.

Em cada intervalo, uma mesma tarefa está executando.

Método guloso

Basta guardarmos no máximo $2n$ intervalos.

Em cada intervalo, uma mesma tarefa está executando.

A cada evento, decidimos se há uma troca por outra tarefa.

Método guloso

Basta guardarmos no máximo $2n$ intervalos.

Em cada intervalo, uma mesma tarefa está executando.

A cada evento, decidimos se há uma troca por outra tarefa.

É possível mostrar que, devido à ordenação, o escalonamento produzido é ótimo.

Método guloso

Basta guardarmos no máximo $2n$ intervalos.

Em cada intervalo, uma mesma tarefa está executando.

A cada evento, decidimos se há uma troca por outra tarefa.

É possível mostrar que, devido à ordenação, o escalonamento produzido é ótimo.

Exercício: Escreva o algoritmo sugerido, mostre que sua implementação é polinomial e prove que o algoritmo resolve o problema em questão (descrito pelo PLI).

Método guloso

Basta guardarmos no máximo $2n$ intervalos.

Em cada intervalo, uma mesma tarefa está executando.

A cada evento, decidimos se há uma troca por outra tarefa.

É possível mostrar que, devido à ordenação, o escalonamento produzido é ótimo.

Exercício: Escreva o algoritmo sugerido, mostre que sua implementação é polinomial e prove que o algoritmo resolve o problema em questão (descrito pelo PLI).

Mas ele não produz uma solução (y, C).
Produz os intervalos e a tarefa associada a cada um.

Mas como conectar as coisas?

Como usar a saída do guloso para fazer o sorteio de X_j?

Mas como conectar as coisas?

Como usar a saída do guloso para fazer o sorteio de X_j?

Para cada j, escolha um valor $\alpha_j \in [0, 1]$,
e tome X_j como o ponto α_j da tarefa j:

o instante em que $\alpha_j \, d_j$ unidades de j foram processadas.

Mas como conectar as coisas?

Como usar a saída do guloso para fazer o sorteio de X_j?

Para cada j, escolha um valor $\alpha_j \in [0, 1]$,
e tome X_j como o ponto α_j da tarefa j:

o instante em que $\alpha_j\, d_j$ unidades de j foram processadas.

É fácil calcular qual é esse instante dos intervalos.

Mas como conectar as coisas?

Como usar a saída do guloso para fazer o sorteio de X_j?

Para cada j, escolha um valor $\alpha_j \in [0, 1]$,
e tome X_j como o ponto α_j da tarefa j:

o instante em que $\alpha_j\, d_j$ unidades de j foram processadas.

É fácil calcular qual é esse instante dos intervalos.

Para ver que isso é equivalente ao que queríamos,
calculemos $\Pr[X_j \in [t-1, t)]$.

Como fazer o sorteio de X_j

Escolha um valor $\alpha_j \in [0,1]$,
e tome X_j como o ponto α_j da tarefa j:
o instante em que $\alpha_j\, d_j$ unidades de j foram processadas.

Calculemos $\Pr[X_j \in [t-1, t)]$

Como fazer o sorteio de X_j

Escolha um valor $\alpha_j \in [0,1]$,
e tome X_j como o ponto α_j da tarefa j:
o instante em que $\alpha_j\, d_j$ unidades de j foram processadas.

Calculemos $\Pr[X_j \in [t-1, t)]$

Devemos ter $\alpha_j d_j$ unidade da tarefa j concluídas neste intervalo, que é a probabilidade de

$$\sum_{s=1}^{t-1} y_{js}^* \;\leq\; \alpha_j d_j \;\leq\; \sum_{s=1}^{t} y_{js}^*.$$

Como fazer o sorteio de X_j

Escolha um valor $\alpha_j \in [0,1]$,
e tome X_j como o ponto α_j da tarefa j:
o instante em que $\alpha_j\, d_j$ unidades de j foram processadas.

Calculemos $\Pr[X_j \in [t-1, t)]$

Devemos ter $\alpha_j d_j$ unidade da tarefa j concluídas neste intervalo, que é a probabilidade de

$$\sum_{s=1}^{t-1} y_{js}^* \;\leq\; \alpha_j d_j \;\leq\; \sum_{s=1}^{t} y_{js}^*.$$

Como α é escolhido uniformemente em $[0,1]$,
essa probabilidade é $\frac{y_{jt}^*}{d_j}$.

Comentários finais

Não sabemos desaleatorizar esse algoritmo...

Comentários finais

Não sabemos desaleatorizar esse algoritmo...

Por outro lado, pode-se mostrar que
basta sortear um mesmo α para todas as tarefas.

Comentários finais

Não sabemos desaleatorizar esse algoritmo...

Por outro lado, pode-se mostrar que
basta sortear um mesmo α para todas as tarefas.

E essa variante do algoritmo pode ser desaleatorizada!

Comentários finais

Não sabemos desaleatorizar esse algoritmo...

Por outro lado, pode-se mostrar que
basta sortear um mesmo α para todas as tarefas.

E essa variante do algoritmo pode ser desaleatorizada!

Ideia: Calcule, ao final de cada intervalo do escalonamento com interrupção, a fração concluída de cada tarefa (apenas a fração de uma tarefa muda ao final de cada intervalo).

Cada valor distinto de fração que aparece induz uma ordem nas tarefas.

Construa o escalonamento sem interrupção para cada uma destas ordens, e devolva um de menor soma ponderada dos tempos de conclusão.

Exemplo com 7 tarefas

j	r_j	d_j	w_j
1	5	1	8
2	12	1	6
3	16	2	10
4	3	4	16
5	2	3	9
6	0	5	10
7	10	1	1

Exemplo com 7 tarefas

j	r_j	d_j	w_j
1	5	1	8
2	12	1	6
3	16	2	10
4	3	4	16
5	2	3	9
6	0	5	10
7	10	1	1

Solução com interrupção:

6	6	5	4	4	1	4	4	5	5	6	6	2	6	7		3	3

Exemplo com 7 tarefas

j	r_j	d_j	w_j
1	5	1	8
2	12	1	6
3	16	2	10
4	3	4	16
5	2	3	9
6	0	5	10
7	10	1	1

6	6	5	4	4	1	4	4	5	5	6	6	2	6	7		3	3

Exemplo com 7 tarefas

j	r_j	d_j	w_j
1	5	1	8
2	12	1	6
3	16	2	10
4	3	4	16
5	2	3	9
6	0	5	10
7	10	1	1

6	6	5	4	4	1	4	4	5	5	6	6	2	6	7		3	3

Intervalos: $(1,6,2),(3,5,1),(4,4,2),(6,1,1),(7,4,2),(9,5,2),$
$(11,6,2),(13,2,1),(14,6,1),(15,7,1),(17,3,2)$.

Dos intervalos às ordens

intervalo	1	2	3	4	5	6	7	α
$(1,6,2)$	0	0	0	0	0	0	0	0
$(3,5,1)$	0	0	0	0	0	0,4	0	0,4
$(4,4,2)$	0	0	0	0	0,33	0,4	0	0,33
$(6,1,1)$	0	0	0	0,5	0,33	0,4	0	0,5
$(7,4,2)$	1	0	0	0,5	0,33	0,4	0	1
$(9,5,2)$	1	0	0	1	0,33	0,4	0	
$(11,6,2)$	1	0	0	1	1	0,4	0	
$(13,2,1)$	1	0	0	1	1	0,8	0	0,8
$(14,6,1)$	1	1	0	1	1	0,8	0	
$(15,7,1)$	1	1	0	1	1	1	0	
	1	1	0	1	1	1	1	
$(17,3,2)$	1	1	1	1	1	1	1	

Dos intervalos às ordens

intervalo	1	2	3	4	5	6	7	α	ordem
$(1,6,2)$	0	0	0	0	0	0	0		
$(3,5,1)$	0	0	0	0	0	$0{,}4$	0	$0{,}4$	(6,4,1,5,2,7,3)
$(4,4,2)$	0	0	0	0	$0{,}33$	$0{,}4$	0	$0{,}33$	(6,5,4,1,2,7,3)
$(6,1,1)$	0	0	0	$0{,}5$	$0{,}33$	$0{,}4$	0	$0{,}5$	(4,1,5,6,2,7,3)
$(7,4,2)$	1	0	0	$0{,}5$	$0{,}33$	$0{,}4$	0	1	(1,4,5,2,6,7,3)
$(9,5,2)$	1	0	0	1	$0{,}33$	$0{,}4$	0		
$(11,6,2)$	1	0	0	1	1	$0{,}4$	0		
$(13,2,1)$	1	0	0	1	1	$0{,}8$	0	$0{,}8$	(1,4,5,6,2,7,3)
$(14,6,1)$	1	1	0	1	1	$0{,}8$	0		
$(15,7,1)$	1	1	0	1	1	1	0		
	1	1	0	1	1	1	1		
$(17,3,2)$	1	1	1	1	1	1	1		

Exemplo com 7 tarefas

j	r_j	d_j	w_j
1	5	1	8
2	12	1	6
3	16	2	10
4	3	4	16
5	2	3	9
6	0	5	10
7	10	1	1

Ordens:

(6,4,1,5,2,7,3)

(6,5,4,1,2,7,3)

(4,1,5,6,2,7,3)

(1,4,5,2,6,7,3)

(1,4,5,6,2,7,3)

Exemplo com 7 tarefas

j	r_j	d_j	w_j
1	5	1	8
2	12	1	6
3	16	2	10
4	3	4	16
5	2	3	9
6	0	5	10
7	10	1	1

Ordens:

(6,4,1,5,2,7,3)

(6,5,4,1,2,7,3)

(4,1,5,6,2,7,3)

(1,4,5,2,6,7,3)

(1,4,5,6,2,7,3)

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
6	6	6	6	6	4	4	4	4	1	5	5	5	2	7		3	3

valor

$$= 5 \cdot 10 + 9 \cdot 16 + 10 \cdot 8 + 13 \cdot 9 + 14 \cdot 6 + 15 \cdot 1 + 18 \cdot 10 = 680$$

Exemplo com 7 tarefas

j	r_j	d_j	w_j
1	5	1	8
2	12	1	6
3	16	2	10
4	3	4	16
5	2	3	9
6	0	5	10
7	10	1	1

Ordens:

(6,4,1,5,2,7,3)

(6,5,4,1,2,7,3)

(4,1,5,6,2,7,3)

(1,4,5,2,6,7,3)

(1,4,5,6,2,7,3)

Exemplo com 7 tarefas

j	r_j	d_j	w_j
1	5	1	8
2	12	1	6
3	16	2	10
4	3	4	16
5	2	3	9
6	0	5	10
7	10	1	1

Ordens:

(6,4,1,5,2,7,3)

(6,5,4,1,2,7,3)

(4,1,5,6,2,7,3)

(1,4,5,2,6,7,3)

(1,4,5,6,2,7,3)

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
6	6	6	6	6	5	5	5	4	4	4	4	1	2	7		3	3

valor

$$= 5 \cdot 10 + 8 \cdot 9 + 12 \cdot 16 + 13 \cdot 8 + 14 \cdot 6 + 15 \cdot 1 + 18 \cdot 10 = 697$$

Exemplo com 7 tarefas

j	r_j	d_j	w_j
1	5	1	8
2	12	1	6
3	16	2	10
4	3	4	16
5	2	3	9
6	0	5	10
7	10	1	1

Ordens:

(6,4,1,5,2,7,3)

(6,5,4,1,2,7,3)

(4,1,5,6,2,7,3)

(1,4,5,2,6,7,3)

(1,4,5,6,2,7,3)

Exemplo com 7 tarefas

j	r_j	d_j	w_j
1	5	1	8
2	12	1	6
3	16	2	10
4	3	4	16
5	2	3	9
6	0	5	10
7	10	1	1

Ordens:

(6,4,1,5,2,7,3)

(6,5,4,1,2,7,3)

(4,1,5,6,2,7,3)

(1,4,5,2,6,7,3)

(1,4,5,6,2,7,3)

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
4	4	4	4	1	5	5	5	6	6	6	6	6	2	7		3	3

valor $= 4 \cdot 16 + 5 \cdot 8 + 8 \cdot 9 + 13 \cdot 10 + 14 \cdot 6 + 15 \cdot 1 + 18 \cdot 10 = 585$