A $\frac{3}{2}$ dual approximation for the problem of minimazing makespan when scheduling independent tasks on heterogeneous processors

Marin Bougeret, Pierre-François Dutot, Erik Saule

Laboratoire LIG-ID

12 - 16 may 2008 CIRM - Marseille

Definition of the problem

Using the standard notation, this problem is $Q||C_{max}$

- Offline monocriteria optimization problem
- Input :
  - m machines of speed $s_1 \leq s_2 \leq ... \leq s_m$, $s_i > 0$
  - n jobs with processing requirement $p_1 \leq ... \leq p_n$, $p_j > 0$ (processing job j on machine i will take $\frac{p_j}{s_i}$ units of time, ie processors are related)
- Output : A schedule σ which assigns to each job one machine and one starting date
- Constraints :
  - A job can be processed on only one machine
  - No preemption, no restart
- Objective function : makespan

Definition of the problem

Using the standard notation, this problem is $Q||C_{max}$

- Offline monocriteria optimization problem
- Input :
  - m machines of speed $s_1 \leq s_2 \leq ... \leq s_m$, $s_i > 0$
  - n jobs with processing requirement $p_1 \leq ... \leq p_n$, $p_j > 0$ (processing job j on machine i will take $\frac{p_j}{s_i}$ units of time, ie processors are related)
- Output : A schedule σ which assigns to each job one machine and one starting date
- Constraints :
  - A job can be processed on only one machine
  - No preemption, no restart
- Objective function : makespan

Definition of the problem

Using the standard notation, this problem is $Q||C_{max}$

- Offline monocriteria optimization problem
- Input :
  - m machines of speed $s_1 \leq s_2 \leq ... \leq s_m, \; s_i > 0$
  - n jobs with processing requirement $p_1 \leq ... \leq p_n, \; p_j > 0$ (processing job j on machine i will take $\frac{p_j}{s_i}$ units of time, ie processors are related)
- Output : A schedule σ which assigns to each job one machine and one starting date
- Constraints :
  - A job can be processed on only one machine
  - No preemption, no restart
- Objective function : makespan

Definition of the problem

Using the standard notation, this problem is $Q||C_{max}$

- Offline monocriteria optimization problem
- Input :
  - m machines of speed $s_1 \leq s_2 \leq ... \leq s_m, \;\; s_i > 0$
  - n jobs with processing requirement $p_1 \leq ... \leq p_n, \;\; p_j > 0$ (processing job j on machine i will take $\frac{p_j}{s_i}$ units of time, ie processors are related)
- Output : A schedule σ which assigns to each job one machine and one starting date
- Constraints :
  - A job can be processed on only one machine
  - No preemption, no restart
- Objective function : makespan

Definition of the problem

Using the standard notation, this problem is $Q||C_{max}$

- Offline monocriteria optimization problem
- Input :
  - m machines of speed $s_1 \leq s_2 \leq ... \leq s_m, \;\; s_i > 0$
  - n jobs with processing requirement $p_1 \leq ... \leq p_n, \;\; p_j > 0$ (processing job j on machine i will take $\frac{p_j}{s_i}$ units of time, ie processors are related)
- Output : A schedule σ which assigns to each job one machine and one starting date
- Constraints :
  - A job can be processed on only one machine
  - No preemption, no restart
- Objective function : makespan

Results on this problem

Existing results :

- $Q||Cmax$ is unary NP hard
- Gonzalez, Ibarra and Sahni [SIAM JoC 77] provided a 2 approximation algorithm (a list algorithm)
- Hochbaum and Shmoys [SIAM JoC 88] provided a PTAS

Our result :

- A $\frac{3}{2}$ approximation algorithm

Results on this problem

Existing results :

- $Q||Cmax$ is unary NP hard
- Gonzalez, Ibarra and Sahni [SIAM JoC 77] provided a 2 approximation algorithm (a list algorithm)
- Hochbaum and Shmoys [SIAM JoC 88] provided a PTAS

Our result :

- A $\frac{3}{2}$ approximation algorithm

Results on this problem

Existing results :

- $Q||Cmax$ is unary NP hard
- Gonzalez, Ibarra and Sahni [SIAM JoC 77] provided a 2 approximation algorithm (a list algorithm)
- Hochbaum and Shmoys [SIAM JoC 88] provided a PTAS

Our result :

- A $\frac{3}{2}$ approximation algorithm

Why is this interesting ?

- $\longrightarrow$ the algorithmic complexity of the $PTAS$ is huge ($O(n^{43})$ to get a ratio of $\frac{3}{2}$)
- $\longrightarrow$ our algorithm, based on dual approximation and list technique, is simple and quick
- $\longrightarrow$ a naive implementation of the Gonzalez algorithm leads to $O(nlog(n) + mn)$ while our algorithm is in $O(nlog(n) + mlog(m) + mnlog(\sum_{i=1}^{n} p_i))$

Layout

1. Two classical list algorithms applied on $Q||Cmax$
   - Analysis of EFT without sorting the tasks
   - Analysis of EFT with decreasing sort
2. Our $\frac{3}{2}$ approximation for $Q||C_{max}$
   - The algorithm
   - Proof and tightness of the bound

1. Two classical list algorithms applied on $Q||Cmax$
   - Analysis of EFT without sorting the tasks
   - Analysis of EFT with decreasing sort
2. Our $\frac{3}{2}$ approximation for $Q||C_{max}$
   - The algorithm
   - Proof and tightness of the bound

Sum up of known results

- Applied on the $P||C_{max}$ problem
  - EFT is exactly the famous [Graham 66] algorithm
  - the approximation ratio is $2 - \frac{1}{m}$
- Applied on the $Q||C_{max}$ problem
  - the approximation ratio is $\theta(log(n))$ (Aspnes and al. [ACM 93])

  $\rightarrow$ schedule first the biggest tasks

Sum up of known results

- Applied on the $P||C_{max}$ problem
  - EFT is exactly the famous [Graham 66] algorithm
  - the approximation ratio is $2 - \frac{1}{m}$
- Applied on the $Q||C_{max}$ problem
  - the approximation ratio is $\theta(log(n))$ (Aspnes and al. [ACM 93])

  $\longrightarrow$ schedule first the biggest tasks

Sum up of known results

- Applied on the $P||C_{max}$ problem
  - EFT + decreasing sort is exactly the famous LPT algorithm
  - the approximation ratio is $\frac{4}{3} - \frac{1}{3m}$ *fixme*
- Applied on the $Q||C_{max}$ problem
  - the approximation ratio is 2, Gonzalez, Ibarra and Sahni [SIAM 77]

Sum up of known results

- Applied on the $P||C_{max}$ problem
  - EFT + decreasing sort is exactly the famous LPT algorithm
  - the approximation ratio is $\frac{4}{3} - \frac{1}{3m} fixme$
- Applied on the $Q||C_{max}$ problem
  - the approximation ratio is 2, Gonzalez, Ibarra and Sahni [SIAM 77]

The Gonzalez ratio of 2

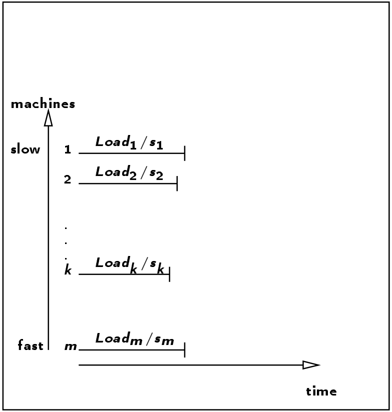
Before scheduling the last task n (the smallest) :

- let $Load_i$: total load of machine i before scheduling n
- completion time on machine i is $\frac{Load_i}{s_i}$
- let $W = \sum_{i=1}^{m} Load_i + p_n$ (total work)
- let $Q = \sum_{i=1}^{m} s_i$

Idea of the proof :

- $s_k C_{max} = Load_k + p_n$
  $s_i C_{max} \leq Load_i + p_n, \forall i \neq k$
- $\sum_{i=1}^{m} s_i \ C_{max} \leq \sum_{i=1}^{m} Load_i + mp_n$
  $QC_{max} \leq W + (m-1)p_n$
  $QC_{max} \leq W + \underbrace{(m-1)p_n}_{\leq W \ \text{if} \ n \geq m..}$

$C_{max} \leq 2\frac{W}{Q} \leq 2C^*_{max}$



The Gonzalez ratio of 2

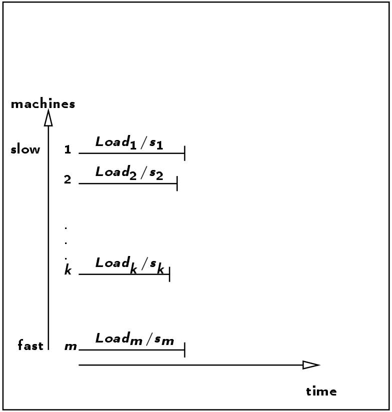
Before scheduling the last task n (the smallest) :

- let $Load_i$: total load of machine i before scheduling n
- completion time on machine i is $\frac{Load_i}{s_i}$
- let $W = \sum_{i=1}^{m} Load_i + p_n$ (total work)
- let $Q = \sum_{i=1}^{m} s_i$

Idea of the proof :

- $s_k C_{max} = Load_k + p_n$
  $s_i C_{max} \leq Load_i + p_n, \forall i \neq k$
- $\sum_{i=1}^{m} s_i \; C_{max} \leq \sum_{i=1}^{m} Load_i + mp_n$
  $QC_{max} \leq W + (m-1)p_n$
  $QC_{max} \leq W + \underbrace{(m-1)p_n}_{\leq W \text{ if } n \geq m..}$

$C_{max} \leq 2\frac{W}{Q} \leq 2C^*_{max}$



The Gonzalez ratio of 2

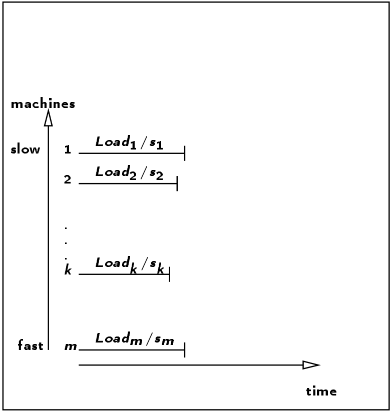
Before scheduling the last task n (the smallest) :

- let $Load_i$: total load of machine i before scheduling n
- completion time on machine i is $\frac{Load_i}{s_i}$
- let $W = \sum_{i=1}^{m} Load_i + p_n$ (total work)
- let $Q = \sum_{i=1}^{m} s_i$

Idea of the proof :

- $s_k C_{max} = Load_k + p_n$
  $s_i C_{max} \leq Load_i + p_n, \forall i \neq k$
- $\sum_{i=1}^{m} s_i \ C_{max} \leq \sum_{i=1}^{m} Load_i + mp_n$
  $QC_{max} \leq W + (m-1)p_n$
  $QC_{max} \leq W + \underbrace{(m-1)p_n}_{\leq W \text{ if } n \geq m..}$

$C_{max} \leq 2\frac{W}{Q} \leq 2C^*_{max}$



The Gonzalez ratio of 2

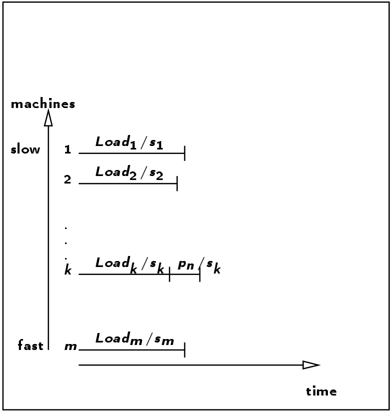
Before scheduling the last task n (the smallest) :

- let $Load_i$: total load of machine i before scheduling n
- completion time on machine i is $\frac{Load_i}{s_i}$
- let $W = \sum_{i=1}^{m} Load_i + p_n$ (total work)
- let $Q = \sum_{i=1}^{m} s_i$

Idea of the proof :

- $s_k C_{max} = Load_k + p_n$
  $s_i C_{max} \leq Load_i + p_n, \forall i \neq k$
- $\sum_{i=1}^{m} s_i \; C_{max} \leq \sum_{i=1}^{m} Load_i + mp_n$
  $QC_{max} \leq W + (m-1)p_n$
  $QC_{max} \leq W + \underbrace{(m-1)p_n}_{\leq W \text{ if } n \geq m..}$

$C_{max} \leq 2\frac{W}{Q} \leq 2C^*_{max}$



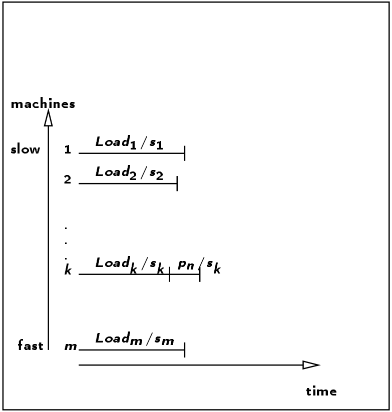
The Gonzalez ratio of 2

Before scheduling the last task n (the smallest) :

- let $Load_i$: total load of machine i before scheduling n
- completion time on machine i is $\frac{Load_i}{s_i}$
- let $W = \sum_{i=1}^{m} Load_i + p_n$ (total work)
- let $Q = \sum_{i=1}^{m} s_i$

Idea of the proof :

- $s_k C_{max} = Load_k + p_n$
  $s_i C_{max} \leq Load_i + p_n, \forall i \neq k$
- $\sum_{i=1}^{m} s_i \ C_{max} \leq \sum_{i=1}^{m} Load_i + mp_n$
  $QC_{max} \leq W + (m-1)p_n$
  $QC_{max} \leq W + \underbrace{(m-1)p_n}_{\leq W \ \text{if} \ n \geq m..}$
  $C_{max} \leq 2\frac{W}{Q} \leq 2C^*_{max}$



The Gonzalez ratio of 2

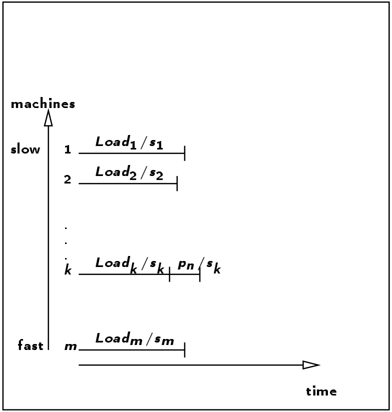
Before scheduling the last task n (the smallest) :

- let $Load_i$: total load of machine i before scheduling n
- completion time on machine i is $\frac{Load_i}{s_i}$
- let $W = \sum_{i=1}^{m} Load_i + p_n$ (total work)
- let $Q = \sum_{i=1}^{m} s_i$

Idea of the proof :

- $s_k C_{max} = Load_k + p_n$
  $s_i C_{max} \leq Load_i + p_n, \forall i \neq k$
- $\sum_{i=1}^{m} s_i \ C_{max} \leq \sum_{i=1}^{m} Load_i + mp_n$
  $QC_{max} \leq W + (m-1)p_n$
  $QC_{max} \leq W + \underbrace{(m-1)p_n}_{\leq W \ \ if \ \ n \geq m..}$

$C_{max} \leq 2\frac{W}{Q} \leq 2C^*_{max}$



A worst case for EFT + Sort

Lower bound for the ratio

There exist instances for which $\frac{C_{max}}{C^*_{max}} \approx \frac{3}{2}$, for $m \longrightarrow \infty$

Let $s_1 = 1 \leq s_2 \leq \ldots \leq s_m$ and
$p'_1 \leq p_1 \leq p_2 \leq p_3, \ldots \leq p_m$, such that :

- $p_1 = p'_1 = p$, (we fixed p) (1)
- $\frac{p_i + p}{s_i} = 2p$, for $2 \leq i \leq m$ (2)
- $\frac{2p}{s_m} = \frac{p_i}{s_{i-1}}$, for $2 \leq i \leq m$ (3)

Thus, we get :

- $C_{max} = 2p$ and $C^*_{max} = \frac{2p}{s_m} \Longrightarrow \frac{C_{max}}{C^*_{max}} = s_m$

In fact :

- (1) $\wedge$ (2) $\wedge$ (3) $\Longrightarrow 2s_m - \frac{3}{s_m} - \frac{1}{s_m^m} + \frac{2}{s_m^{m+1}} = 0$
- $\Longrightarrow s_m = \frac{3}{2} - \epsilon_m$
- $\epsilon_m > 0, \epsilon_m \longrightarrow_{m \to \infty} 0$

A worst case for EFT + Sort

Lower bound for the ratio

There exist instances for which $\frac{C_{max}}{C^*_{max}} \approx \frac{3}{2}$, for $m \to \infty$

Let $s_1 = 1 \leq s_2 \leq \ldots \leq s_m$ and
$p'_1 \leq p_1 \leq p_2 \leq p_3, \ldots \leq p_m$, such that :

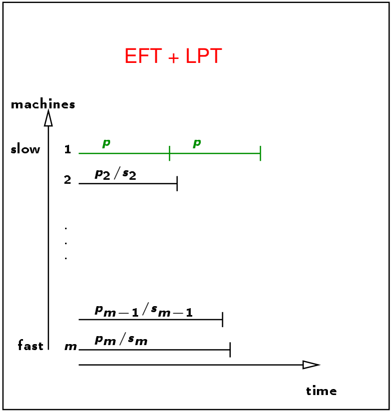
- $p_1 = p'_1 = p$, (we fixed p) (1)
- $\frac{p_i + p}{s_i} = 2p$, for $2 \leq i \leq m$ (2)
- $\frac{2p}{s_m} = \frac{p_i}{s_{i-1}}$, for $2 \leq i \leq m$ (3)

Thus, we get :

- $C_{max} = 2p$ and $C^*_{max} = \frac{2p}{s_m} \implies \frac{C_{max}}{C^*_{max}} = s_m$

In fact :

- $(1) \wedge (2) \wedge (3) \implies 2s_m - \frac{3}{s_m} - \frac{1}{s_m^m} + \frac{2}{s_m^{m+1}} = 0$
- $\implies s_m = \frac{3}{2} - \epsilon_m$
- $\epsilon_m > 0, \epsilon_m \to_{m\to\infty} 0$



A worst case for EFT + Sort

Lower bound for the ratio

There exist instances for which $\frac{C_{max}}{C^*_{max}} \approx \frac{3}{2}$, for $m \to \infty$

Let $s_1 = 1 \leq s_2 \leq \ldots \leq s_m$ and
$p'_1 \leq p_1 \leq p_2 \leq p_3, \ldots \leq p_m$, such that :

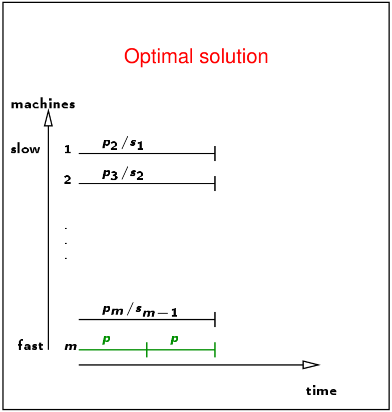
- $p_1 = p'_1 = p$, (we fixed p) (1)
- $\frac{p_i + p}{s_i} = 2p$, for $2 \leq i \leq m$ (2)
- $\frac{2p}{s_m} = \frac{p_i}{s_{i-1}}$, for $2 \leq i \leq m$ (3)

Thus, we get :

- $C_{max} = 2p$ and $C^*_{max} = \frac{2p}{s_m} \implies \frac{C_{max}}{C^*_{max}} = s_m$

In fact :

- (1) $\wedge$ (2) $\wedge$ (3) $\implies 2s_m - \frac{3}{s_m} - \frac{1}{s_m^m} + \frac{2}{s_m^{m+1}} = 0$
- $\implies s_m = \frac{3}{2} - \epsilon_m$
- $\epsilon_m > 0, \epsilon_m \to_{m \to \infty} 0$



A worst case for EFT + Sort

Lower bound for the ratio

There exist instances for which $\frac{C_{max}}{C^*_{max}} \approx \frac{3}{2}$, for $m \to \infty$

Let $s_1 = 1 \leq s_2 \leq \ldots \leq s_m$ and
$p'_1 \leq p_1 \leq p_2 \leq p_3, \ldots \leq p_m$, such that :

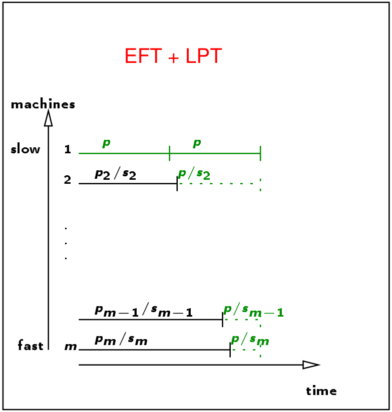
- $p_1 = p'_1 = p$, (we fixed p) (1)
- $\frac{p_i + p}{s_i} = 2p$, for $2 \leq i \leq m$ (2)
- $\frac{2p}{s_m} = \frac{p_i}{s_{i-1}}$, for $2 \leq i \leq m$ (3)

Thus, we get :

- $C_{max} = 2p$ and $C^*_{max} = \frac{2p}{s_m} \implies \frac{C_{max}}{C^*_{max}} = s_m$

In fact :

- $(1) \wedge (2) \wedge (3) \implies 2s_m - \frac{3}{s_m} - \frac{1}{s_m^m} + \frac{2}{s_m^{m+1}} = 0$
- $\implies s_m = \frac{3}{2} - \epsilon_m$
- $\epsilon_m > 0, \epsilon_m \to_{m\to\infty} 0$



A worst case for EFT + Sort

Lower bound for the ratio

There exist instances for which $\frac{C_{max}}{C^*_{max}} \approx \frac{3}{2}$, for $m \to \infty$

Let $s_1 = 1 \leq s_2 \leq \ldots \leq s_m$ and
$p'_1 \leq p_1 \leq p_2 \leq p_3, \ldots \leq p_m$, such that :

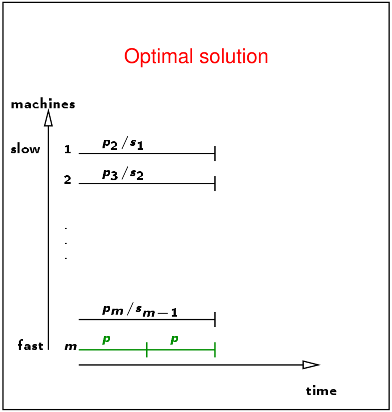
- $p_1 = p'_1 = p$, (we fixed p) (1)
- $\frac{p_i + p}{s_i} = 2p$, for $2 \leq i \leq m$ (2)
- $\frac{2p}{s_m} = \frac{p_i}{s_{i-1}}$, for $2 \leq i \leq m$ (3)

Thus, we get :

- $C_{max} = 2p$ and $C^*_{max} = \frac{2p}{s_m} \implies \frac{C_{max}}{C^*_{max}} = s_m$

In fact :

- $(1) \wedge (2) \wedge (3) \implies 2s_m - \frac{3}{s_m} - \frac{1}{s_m^m} + \frac{2}{s_m^{m+1}} = 0$
- $\implies s_m = \frac{3}{2} - \epsilon_m$
- $\epsilon_m > 0, \epsilon_m \to_{m \to \infty} 0$



A worst case for EFT + Sort

Lower bound for the ratio

There exist instances for which $\frac{C_{max}}{C^*_{max}} \approx \frac{3}{2}$, for $m \to \infty$

Let $s_1 = 1 \leq s_2 \leq \ldots \leq s_m$ and
$p'_1 \leq p_1 \leq p_2 \leq p_3, \ldots \leq p_m$, such that :

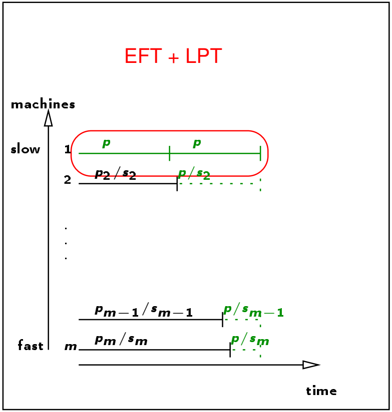
- $p_1 = p'_1 = p$, (we fixed p) (1)
- $\frac{p_i + p}{s_i} = 2p$, for $2 \leq i \leq m$ (2)
- $\frac{2p}{s_m} = \frac{p_i}{s_{i-1}}$, for $2 \leq i \leq m$ (3)

Thus, we get :

- $C_{max} = 2p$ and $C^*_{max} = \frac{2p}{s_m} \Longrightarrow \frac{C_{max}}{C^*_{max}} = s_m$

In fact :

- $(1) \wedge (2) \wedge (3) \Longrightarrow 2s_m - \frac{3}{s_m} - \frac{1}{s_m^m} + \frac{2}{s_m^{m+1}} = 0$
- $\Longrightarrow s_m = \frac{3}{2} - \epsilon_m$
- $\epsilon_m > 0, \epsilon_m \to_{m\to\infty} 0$



A worst case for EFT + Sort

Lower bound for the ratio

There exist instances for which $\frac{C_{max}}{C^*_{max}} \approx \frac{3}{2}$, for $m \to \infty$

Let $s_1 = 1 \leq s_2 \leq \ldots \leq s_m$ and
$p'_1 \leq p_1 \leq p_2 \leq p_3, \ldots \leq p_m$, such that :

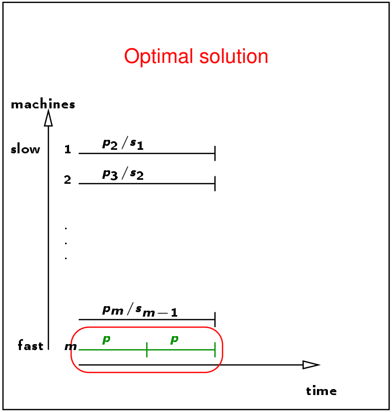
- $p_1 = p'_1 = p$, (we fixed p) (1)
- $\frac{p_i + p}{s_i} = 2p$, for $2 \leq i \leq m$ (2)
- $\frac{2p}{s_m} = \frac{p_i}{s_{i-1}}$, for $2 \leq i \leq m$ (3)

Thus, we get :

- $C_{max} = 2p$ and $C^*_{max} = \frac{2p}{s_m} \implies \frac{C_{max}}{C^*_{max}} = s_m$

In fact :

- (1) $\wedge$ (2) $\wedge$ (3) $\implies 2s_m - \frac{3}{s_m} - \frac{1}{s_m^m} + \frac{2}{s_m^{m+1}} = 0$
- $\implies s_m = \frac{3}{2} - \epsilon_m$
- $\epsilon_m > 0, \epsilon_m \to_{m\to\infty} 0$



A worst case for EFT + Sort

Lower bound for the ratio

There exist instances for which $\frac{C_{max}}{C^*_{max}} \approx \frac{3}{2}$, for $m \to \infty$

Let $s_1 = 1 \leq s_2 \leq \ldots \leq s_m$ and
$p'_1 \leq p_1 \leq p_2 \leq p_3, \ldots \leq p_m$, such that :

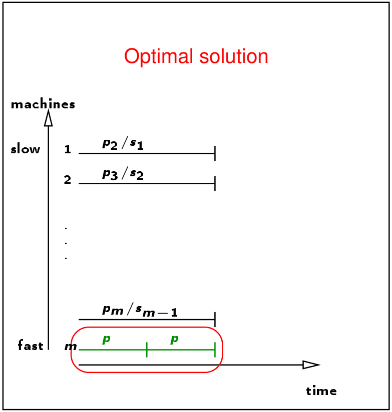
- $p_1 = p'_1 = p$, (we fixed p) (1)
- $\frac{p_i + p}{s_i} = 2p$, for $2 \leq i \leq m$ (2)
- $\frac{2p}{s_m} = \frac{p_i}{s_{i-1}}$, for $2 \leq i \leq m$ (3)

Thus, we get :

- $C_{max} = 2p$ and $C^*_{max} = \frac{2p}{s_m} \implies \frac{C_{max}}{C^*_{max}} = s_m$

In fact :

- (1) $\wedge$ (2) $\wedge$ (3) $\implies 2s_m - \frac{3}{s_m} - \frac{1}{s_m^m} + \frac{2}{s_m^{m+1}} = 0$
- $\implies s_m = \frac{3}{2} - \epsilon_m$
- $\epsilon_m > 0, \epsilon_m \to_{m \to \infty} 0$

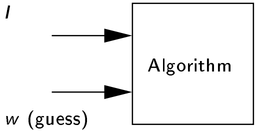


1. Two classical list algorithms applied on $Q||Cmax$
   - Analysis of EFT without sorting the tasks
   - Analysis of EFT with decreasing sort

2. Our $\frac{3}{2}$ approximation for $Q||C_{max}$
   - The algorithm
   - Proof and tightness of the bound

Presentation

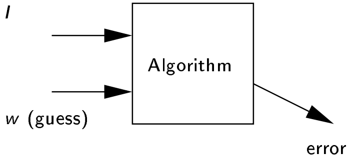
- our algorithm uses the dual approximation technique :



- the ratio is tight
- the complexity is $O(nlog(n) + mlog(m) + mnlog(\sum_{i=1}^{n} p_i))$

Presentation

- our algorithm uses the dual approximation technique :

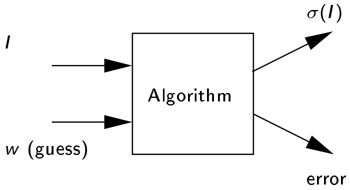


error $\Longrightarrow w < C^*_{max}$

- the ratio is tight
- the complexity is $O(nlog(n) + mlog(m) + mnlog(\sum_{i=1}^{n} p_i))$

Presentation

- our algorithm uses the dual approximation technique :



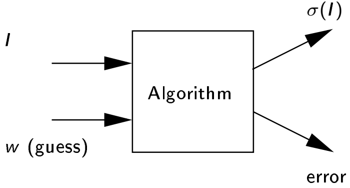
$C_{max} \leq \frac{3}{2} w$

error $\Longrightarrow w < C^*_{max}$

- the ratio is tight
- the complexity is $O(nlog(n) + mlog(m) + mnlog(\sum_{i=1}^{n} p_i))$

Presentation

- our algorithm uses the dual approximation technique :



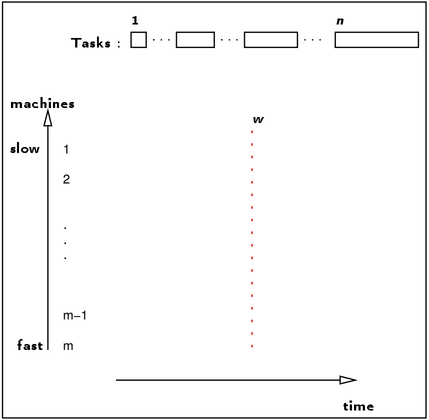
$C_{max} \leq \frac{3}{2} w$

error $\Longrightarrow w < C^*_{max}$

- the ratio is tight
- the complexity is $O(nlog(n) + mlog(m) + mnlog(\sum_{i=1}^{n} p_i))$

The algorithm

$Tasks = \{1, \ldots, n\}$
$C_i^1 = 0, i = 1..m$ /* completion time of each machine */

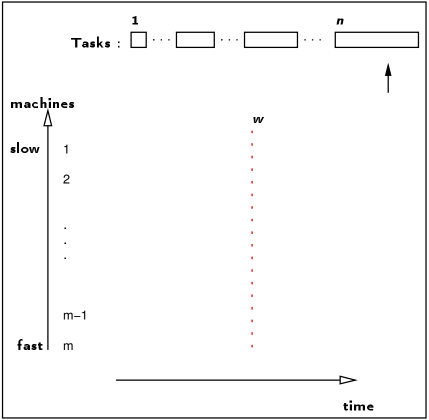


The algorithm

$Tasks = \{1, \ldots, n\}$
$C_i^1 = 0, i = 1..m$ /* completion time of each machine */

/************* Phase 1 *************/

for i *from* 1 *to* m
    for j *from* n *to* 1
        if $(j \in Tasks \wedge C_i^1 + \frac{p_j}{s_i} \leq w)$
            schedule j *on* i (*at time* C_i^1)
            $C_i^1 = C_i^1 + \frac{p_j}{s_i}$
            $Tasks = Tasks - \{j\}$

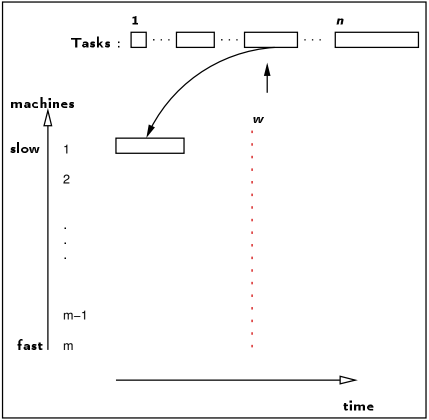


The algorithm

$Tasks = \{1, \ldots, n\}$
$C_i^1 = 0, i = 1..m$ /* completion time of each machine */

/************* Phase 1 *************/

for i *from* 1 *to* m
    for j *from* n *to* 1
        if $(j \in Tasks \wedge C_i^1 + \frac{p_j}{s_i} \leq w)$
            schedule j *on* i (*at time* C_i^1)
            $C_i^1 = C_i^1 + \frac{p_j}{s_i}$
            $Tasks = Tasks - \{j\}$

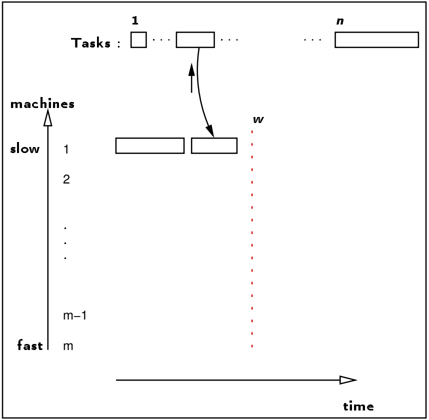


The algorithm

$Tasks = \{1, \dots, n\}$
$C_i^1 = 0, i = 1..m$ /* completion time of each machine */

/************* Phase 1 *************/

for i *from* 1 *to* m
  for j *from* n *to* 1
    if $(j \in Tasks \wedge C_i^1 + \frac{p_j}{s_i} \leq w)$
      schedule j *on* i (*at time* C_i^1)
      $C_i^1 = C_i^1 + \frac{p_j}{s_i}$
      $Tasks = Tasks - \{j\}$

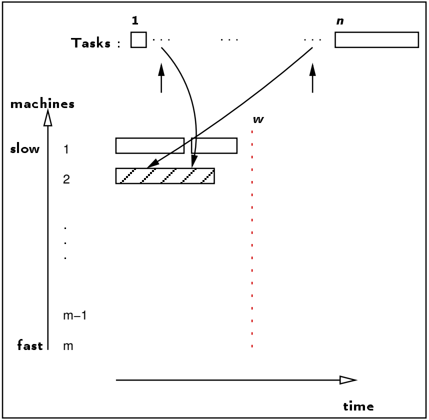


The algorithm

$Tasks = \{1, \dots, n\}$
$C_i^1 = 0, i = 1..m$ /* completion time of each machine */

/************* Phase 1 *************/

for i *from* 1 *to* m
    for j *from* n *to* 1
        if $(j \in Tasks \wedge C_i^1 + \frac{p_j}{s_i} \leq w)$
            schedule j *on* i (*at time* C_i^1)
            $C_i^1 = C_i^1 + \frac{p_j}{s_i}$
            $Tasks = Tasks - \{j\}$

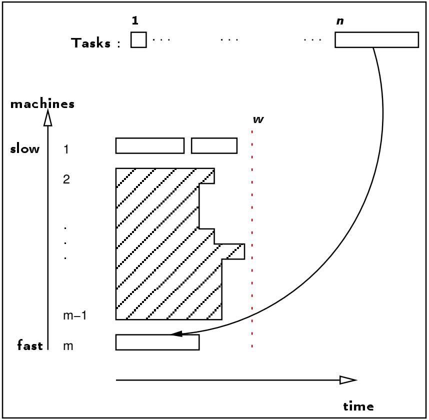


The algorithm

$Tasks = \{1, \ldots, n\}$
$C_i^1 = 0, i = 1..m$ /* completion time of each machine */

/************* Phase 1 *************/

for i *from* 1 *to* m
  for j *from* n *to* 1
    if $(j \in Tasks \wedge C_i^1 + \frac{p_j}{s_i} \leq w)$
      schedule j *on* i (*at time* C_i^1)
      $C_i^1 = C_i^1 + \frac{p_j}{s_i}$
      $Tasks = Tasks - \{j\}$



The algorithm

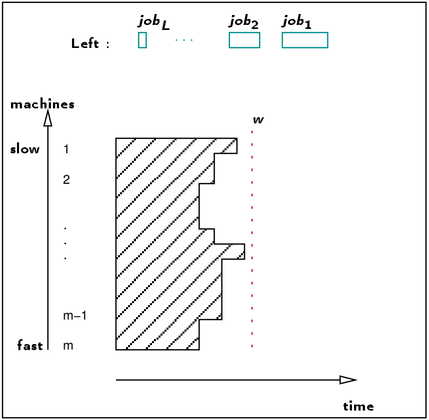
$Tasks = \{1, \ldots, n\}$
$C_i^1 = 0, i = 1..m$ /* completion time of each machine */

/************* Phase 1 *************/

for i *from* 1 *to* m
  for j *from* n *to* 1
    if $(j \in Tasks \wedge C_i^1 + \frac{p_j}{s_i} \leq w)$
      schedule j *on* i (*at time* C_i^1)
      $C_i^1 = C_i^1 + \frac{p_j}{s_i}$
      $Tasks = Tasks - \{j\}$

/************* Phase 2 *************/

/* Let's now use the following notation :
$Left = Tasks = \{job_1, \ldots, job_L\}$, $job_1 \geq \ldots \geq job_L$ */



The algorithm

$Tasks = \{1, \ldots, n\}$
$C_i^1 = 0, i = 1..m$ /* completion time of each machine */

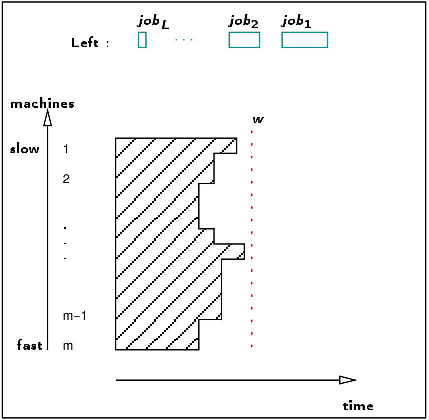
/************* Phase 1 *************/

for i *from* 1 *to* m
    for j *from* n *to* 1
        if $(j \in Tasks \wedge C_i^1 + \frac{p_j}{s_i} \leq w)$
            schedule j *on* i (*at time* C_i^1)
            $C_i^1 = C_i^1 + \frac{p_j}{s_i}$
            $Tasks = Tasks - \{j\}$

/************* Phase 2 *************/

/* Let's now use the following notation :
$Left = Tasks = \{job_1, \ldots, job_L\}$, $job_1 \geq \ldots \geq job_L$ */

if $L \geq m$
    return error /* 1^{st} reject condition */



The algorithm

$Tasks = \{1, \dots, n\}$
$C_i^1 = 0, i = 1..m$ /* completion time of each machine */

/************* Phase 1 *************/

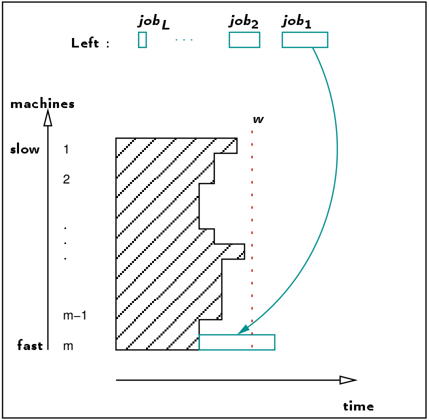
for i *from* 1 *to* m
    for j *from* n *to* 1
        if $(j \in Tasks \wedge C_i^1 + \frac{p_j}{s_i} \leq w)$
            schedule j *on* i *(at time* C_i^1)
            $C_i^1 = C_i^1 + \frac{p_j}{s_i}$
            $Tasks = Tasks - \{j\}$

/************* Phase 2 *************/

/* Let's now use the following notation :
$Left = Tasks = \{job_1, \dots, job_L\}$, $job_1 \geq \dots \geq job_L$ */

if $L \geq m$
    return error /* 1^{st} reject condition */

for j *from* 1 *to* L
    if $\frac{job_j}{s_{m-j+1}} > \frac{w}{2}$
        return error /* 2^{nd} reject condition */
    schedule job_j *on* $m - j + 1$ *(at time* C^1_{m-j+1})



The algorithm

$Tasks = \{1, \ldots, n\}$
$C_i^1 = 0, i = 1..m$ /* completion time of each machine */

/************* Phase 1 *************/

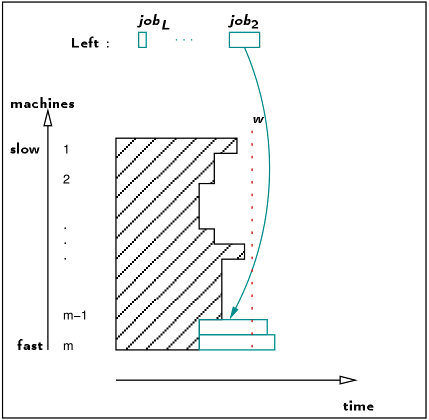
for i *from* 1 *to* m
  for j *from* n *to* 1
    if $(j \in Tasks \wedge C_i^1 + \frac{p_j}{s_i} \leq w)$
      schedule j *on* i (*at time* C_i^1)
      $C_i^1 = C_i^1 + \frac{p_j}{s_i}$
      $Tasks = Tasks - \{j\}$

/************* Phase 2 *************/

/* Let's now use the following notation :
$Left = Tasks = \{job_1, \ldots, job_L\}$, $job_1 \geq \ldots \geq job_L$ */

if $L \geq m$
  return error /* 1^{st} reject condition */

for j *from* 1 *to* L
  if $\frac{job_j}{s_{m-j+1}} > \frac{w}{2}$
    return error /* 2^{nd} reject condition */
  schedule job_j *on* $m - j + 1$ (*at time* C^1_{m-j+1})



The algorithm

$Tasks = \{1, \ldots, n\}$
$C_i^1 = 0, i = 1..m$ /* completion time of each machine */

/************* Phase 1 *************/

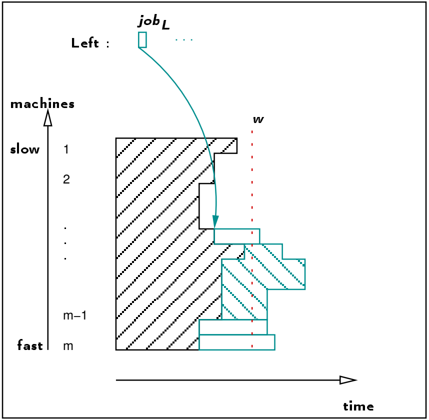
for i *from* 1 *to* m
    for j *from* n *to* 1
        if $(j \in Tasks \wedge C_i^1 + \frac{p_j}{s_i} \leq w)$
            schedule j *on* i *(at time* C_i^1*)*
            $C_i^1 = C_i^1 + \frac{p_j}{s_i}$
            $Tasks = Tasks - \{j\}$

/************* Phase 2 *************/

/* Let's now use the following notation :
$Left = Tasks = \{job_1, \ldots, job_L\}$, $job_1 \geq \ldots \geq job_L$ */

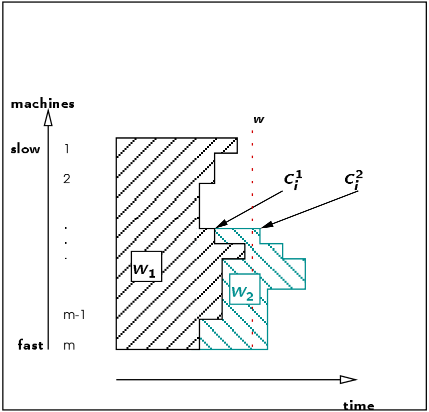
if $L \geq m$
    return error /* 1^{st} reject condition */

for j *from* 1 *to* L
    if $\frac{job_j}{s_{m-j+1}} > \frac{w}{2}$
        return error /* 2^{nd} reject condition */
    schedule job_j *on* $m - j + 1$ *(at time* C^1_{m-j+1}*)*



Notations

- let C_i^1 : completion time of machine i at the end of phase 1
- let C_i^2 : completion time of machine i at the end of phase 2
- $W_1 = \sum_{i=1}^{m} C_i^1 s_i$
- $W_2 = \sum_{j \in Left} p_{job_j}$
- $W = W_1 + W_2$



Proof of the 3/2 bound

Ratio of the algorithm

If the algorithm does not fail, $C_{max} \leq \frac{3}{2} w$

Proof :

- $L < m \implies$ all jobs are scheduled after phase 2
- $\forall i \in \{1, .., m - L\}, C_i^2 = C_i^1 \leq w$
- $\forall j \in \{1, .., L\}, \frac{job_j}{s_{m-j+1}} \leq \frac{w}{2} \implies$

$$\forall i \in \{m - L + 1, .., m\}, C_i^2 = C_i^1 + \frac{job_j}{s_{m-j+1}} \leq w + \frac{w}{2} = \frac{3}{2} w$$

Proof of the 3/2 bound

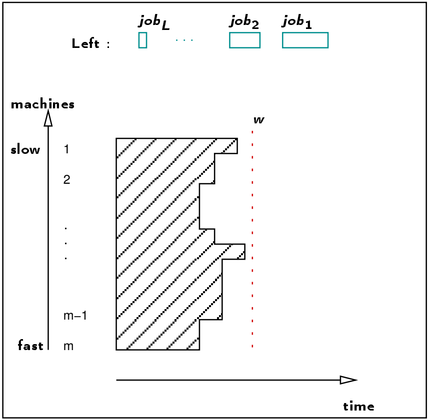
Ratio of the algorithm

If the algorithm does not fail, $C_{max} \leq \frac{3}{2} w$

Proof :

- $L < m \Longrightarrow$ all jobs are scheduled after phase 2
- $\forall i \in \{1, .., m - L\}, C_i^2 = C_i^1 \leq w$
- $\forall j \in \{1, .., L\}, \frac{job_j}{s_{m-j+1}} \leq \frac{w}{2} \Longrightarrow$

  $\forall i \in \{m - L + 1, .., m\}, C_i^2 = C_i^1 + \frac{job_j}{s_{m-j+1}}$
  $\leq w + \frac{w}{2} = \frac{3}{2} w$



Proof of the 3/2 bound

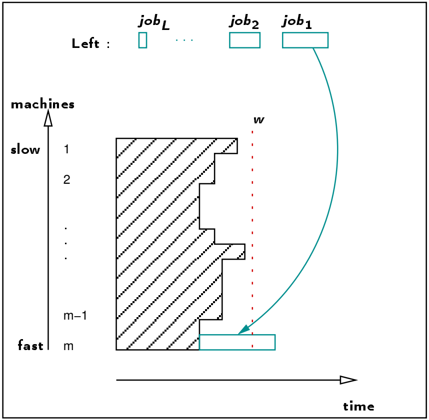
Ratio of the algorithm

If the algorithm does not fail, $C_{max} \leq \frac{3}{2} w$

Proof :

- $L < m \Longrightarrow$ all jobs are scheduled after phase 2
- $\forall i \in \{1, .., m-L\}, C_i^2 = C_i^1 \leq w$
- $\forall j \in \{1, .., L\}, \frac{job_j}{s_{m-j+1}} \leq \frac{w}{2} \Longrightarrow$

$$\forall i \in \{m-L+1, .., m\}, C_i^2 = C_i^1 + \frac{job_j}{s_{m-j+1}} \leq w + \frac{w}{2} = \frac{3}{2} w$$



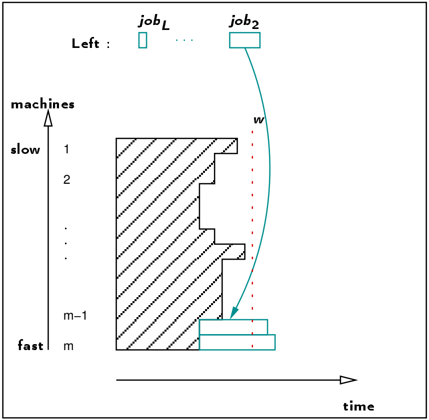
Proof of the 3/2 bound

Ratio of the algorithm

If the algorithm does not fail, $C_{max} \leq \frac{3}{2} w$

Proof :

- $L < m \Longrightarrow$ all jobs are scheduled after phase 2
- $\forall i \in \{1, .., m - L\}, C_i^2 = C_i^1 \leq w$
- $\forall j \in \{1, .., L\}, \frac{job_j}{s_{m-j+1}} \leq \frac{w}{2} \Longrightarrow$
  $\forall i \in \{m - L + 1, .., m\}, C_i^2 = C_i^1 + \frac{job_j}{s_{m-j+1}} \leq w + \frac{w}{2} = \frac{3}{2} w$



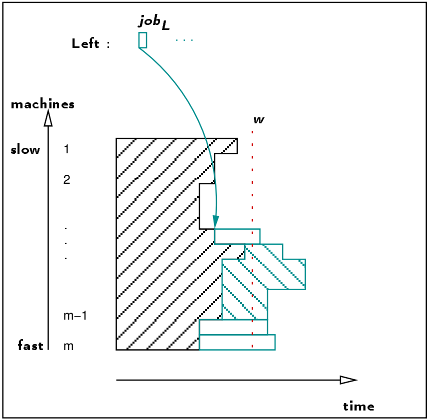
Proof of the 3/2 bound

Ratio of the algorithm

If the algorithm does not fail, $C_{max} \leq \frac{3}{2}w$

Proof :

- $L < m \Longrightarrow$ all jobs are scheduled after phase 2
- $\forall i \in \{1, .., m - L\}, C_i^2 = C_i^1 \leq w$
- $\forall j \in \{1, .., L\}, \frac{job_j}{s_{m-j+1}} \leq \frac{w}{2} \Longrightarrow$
  $\forall i \in \{m - L + 1, ..., m\}, C_i^2 = C_i^1 + \frac{job_j}{s_{m-j+1}}$
  $\leq w + \frac{w}{2} = \frac{3}{2}w$



Proof of the 3/2 bound

Ratio of the algorithm

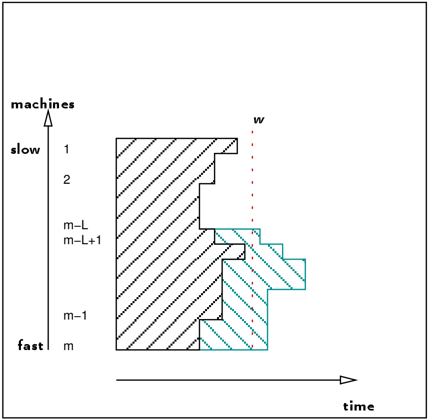
If the algorithm does not fail, $C_{max} \leq \frac{3}{2}w$

Proof :

- $L < m \Longrightarrow$ all jobs are scheduled after phase 2
- $\forall i \in \{1, .., m - L\}, C_i^2 = C_i^1 \leq w$
- $\forall j \in \{1, .., L\}, \frac{job_j}{s_{m-j+1}} \leq \frac{w}{2} \Longrightarrow$

  $\forall i \in \{m - L + 1, .., m\}, C_i^2 = C_i^1 + \frac{job_j}{s_{m-j+1}}$

  $\leq w + \frac{w}{2} = \frac{3}{2}w$



Proof of the 3/2 bound

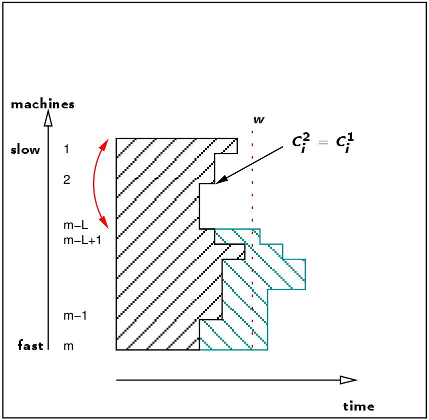
Ratio of the algorithm

If the algorithm does not fail, $C_{max} \leq \frac{3}{2} w$

Proof :

- $L < m \Longrightarrow$ all jobs are scheduled after phase 2
- $\forall i \in \{1, .., m - L\}, C_i^2 = C_i^1 \leq w$
- $\forall j \in \{1, .., L\}, \frac{job_j}{s_{m-j+1}} \leq \frac{w}{2} \Longrightarrow$

  $\forall i \in \{m - L + 1, .., m\}, C_i^2 = C_i^1 + \frac{job_j}{s_{m-j+1}}$
  $\leq w + \frac{w}{2} = \frac{3}{2} w$



Proof of the 3/2 bound

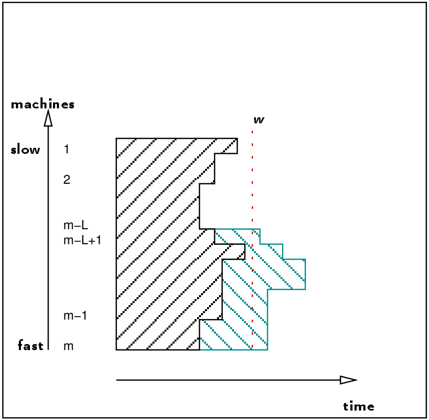
Ratio of the algorithm

If the algorithm does not fail, $C_{max} \leq \frac{3}{2} w$

Proof :

- $L < m \Longrightarrow$ all jobs are scheduled after phase 2
- $\forall i \in \{1, .., m-L\}, C_i^2 = C_i^1 \leq w$
- $\forall j \in \{1, .., L\}, \frac{job_j}{s_{m-j+1}} \leq \frac{w}{2} \Longrightarrow$

  $\forall i \in \{m-L+1, .., m\}, C_i^2 = C_i^1 + \frac{job_j}{s_{m-j+1}}$
  $\leq w + \frac{w}{2} = \frac{3}{2} w$



Proof of the 3/2 bound

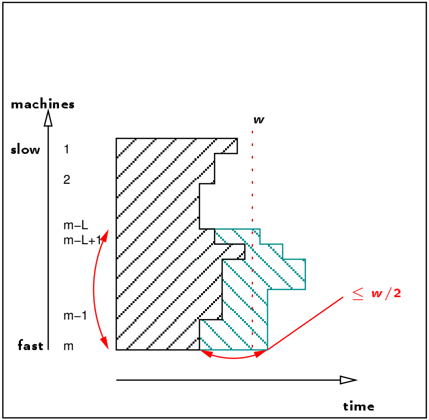
Ratio of the algorithm

If the algorithm does not fail, $C_{max} \leq \frac{3}{2} w$

Proof :

- $L < m \Longrightarrow$ all jobs are scheduled after phase 2
- $\forall i \in \{1, .., m - L\}, C_i^2 = C_i^1 \leq w$
- $\forall j \in \{1, .., L\}, \frac{job_j}{s_{m-j+1}} \leq \frac{w}{2} \Longrightarrow$

$\forall i \in \{m - L + 1, .., m\}, C_i^2 = C_i^1 + \frac{job_j}{s_{m-j+1}} \leq w + \frac{w}{2} = \frac{3}{2} w$



Proof of the 3/2 bound

Ratio of the algorithm

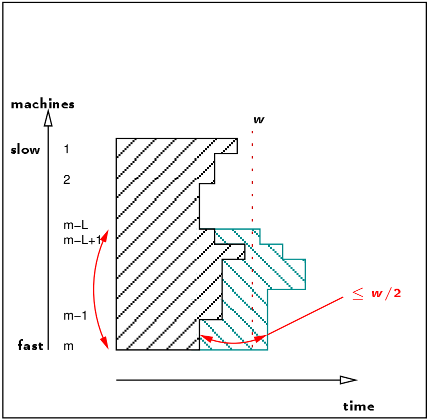
If the algorithm does not fail, $C_{max} \leq \frac{3}{2} w$

Proof :

- $L < m \Longrightarrow$ all jobs are scheduled after phase 2
- $\forall i \in \{1, .., m - L\}, C_i^2 = C_i^1 \leq w$
- $\forall j \in \{1, .., L\}, \frac{job_j}{s_{m-j+1}} \leq \frac{w}{2} \Longrightarrow$

  $\forall i \in \{m - L + 1, .., m\}, C_i^2 = C_i^1 + \frac{job_j}{s_{m-j+1}}$

  $\leq w + \frac{w}{2} = \frac{3}{2} w$



Proof of the 3/2 bound

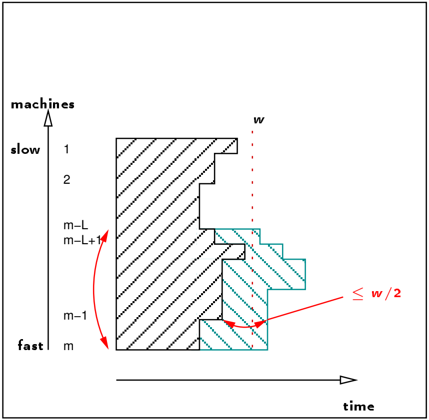
Ratio of the algorithm

If the algorithm does not fail, $C_{max} \leq \frac{3}{2}w$

Proof :

- $L < m \Longrightarrow$ all jobs are scheduled after phase 2
- $\forall i \in \{1, .., m-L\}, C_i^2 = C_i^1 \leq w$
- $\forall j \in \{1, .., L\}, \frac{job_j}{s_{m-j+1}} \leq \frac{w}{2} \Longrightarrow$

$$\forall i \in \{m-L+1, .., m\}, C_i^2 = C_i^1 + \frac{job_j}{s_{m-j+1}} \leq w + \frac{w}{2} = \frac{3}{2}w$$



Proof of the 3/2 bound

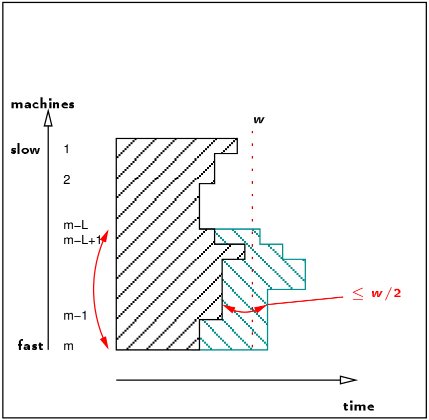
Ratio of the algorithm

If the algorithm does not fail, $C_{max} \leq \frac{3}{2} w$

Proof :

- $L < m \Longrightarrow$ all jobs are scheduled after phase 2
- $\forall i \in \{1, .., m - L\}, C_i^2 = C_i^1 \leq w$
- $\forall j \in \{1, .., L\}, \frac{job_j}{s_{m-j+1}} \leq \frac{w}{2} \Longrightarrow$

$$\forall i \in \{m - L + 1, .., m\}, C_i^2 = C_i^1 + \frac{job_j}{s_{m-j+1}} \leq w + \frac{w}{2} = \frac{3}{2} w$$



Proof of the 3/2 bound

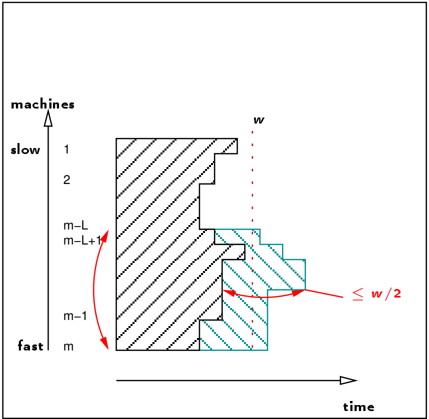
Ratio of the algorithm

If the algorithm does not fail, $C_{max} \leq \frac{3}{2} w$

Proof :

- $L < m \Longrightarrow$ all jobs are scheduled after phase 2
- $\forall i \in \{1, .., m - L\}, C_i^2 = C_i^1 \leq w$
- $\forall j \in \{1, .., L\}, \frac{job_j}{s_{m-j+1}} \leq \frac{w}{2} \Longrightarrow$

  $\forall i \in \{m - L + 1, .., m\}, C_i^2 = C_i^1 + \frac{job_j}{s_{m-j+1}} \leq w + \frac{w}{2} = \frac{3}{2} w$



Proof of the 3/2 bound

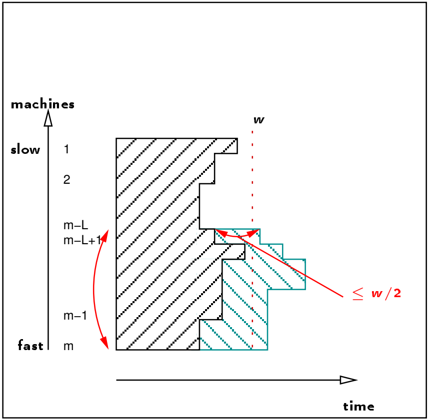
Ratio of the algorithm

If the algorithm does not fail, $C_{max} \leq \frac{3}{2} w$

Proof :

- $L < m \implies$ all jobs are scheduled after phase 2
- $\forall i \in \{1, .., m-L\}, C_i^2 = C_i^1 \leq w$
- $\forall j \in \{1, .., L\}, \frac{job_j}{s_{m-j+1}} \leq \frac{w}{2} \implies$

  $\forall i \in \{m-L+1, .., m\}, C_i^2 = C_i^1 + \frac{job_j}{s_{m-j+1}}$
  $\leq w + \frac{w}{2} = \frac{3}{2} w$



Proof of the 3/2 bound

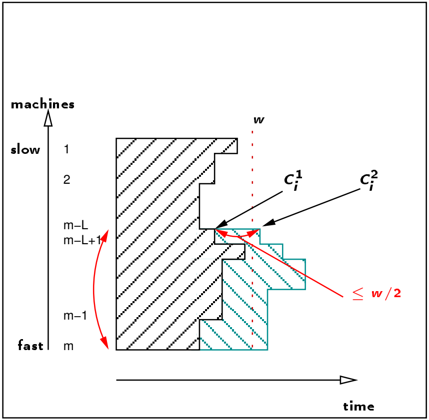
Ratio of the algorithm

If the algorithm does not fail, $C_{max} \leq \frac{3}{2}w$

Proof :

- $L < m \Longrightarrow$ all jobs are scheduled after phase 2
- $\forall i \in \{1, .., m-L\}, C_i^2 = C_i^1 \leq w$
- $\forall j \in \{1, .., L\}, \frac{job_j}{s_{m-j+1}} \leq \frac{w}{2} \Longrightarrow$

  $\forall i \in \{m-L+1, .., m\}, C_i^2 = C_i^1 + \frac{job_j}{s_{m-j+1}}$
  $\leq w + \frac{w}{2} = \frac{3}{2}w$



Proof of the first reject condition

First reject condition

$\neg(L < m) \Longrightarrow w < C^*_{max}$

Proof :

- By construction of *Left* :

  $\forall j \in Left, \forall i \in \{1, .., m\}, C_i^1 + \frac{job_j}{s_i} > w$ (1)

- Now suppose $L \geq m$

- using the classical Graham bound, we get $C^*_{max} \geq \frac{W}{\sum_{i=1}^{m} s_i} > w$

Proof of the first reject condition

First reject condition

$\neg(L < m) \implies w < C^*_{max}$

Proof :

- By construction of *Left* :
  $\forall j \in Left, \forall i \in \{1, .., m\}, C_i^1 + \frac{job_j}{s_i} > w$ (1)
- Now suppose $L \geq m$
- using the classical Graham bound, we get $C^*_{max} \geq \frac{W}{\sum_{i=1}^{m} s_i} > w$

Proof of the first reject condition

First reject condition

$\neg(L < m) \Longrightarrow w < C^*_{max}$

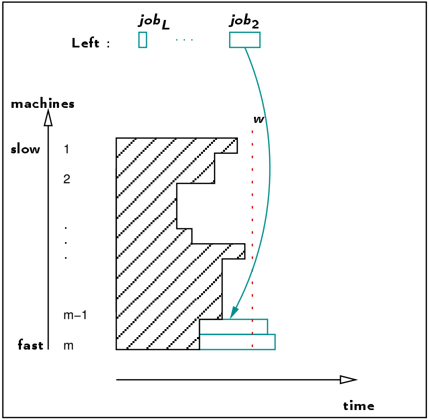
Proof :

- By construction of *Left* :

  $\forall j \in Left, \forall i \in \{1, .., m\}, C_i^1 + \frac{job_j}{s_i} > w$ (1)

- Now suppose $L \geq m$

- using the classical Graham bound, we get $C^*_{max} \geq \frac{W}{\sum_{i=1}^{m} s_i} > w$



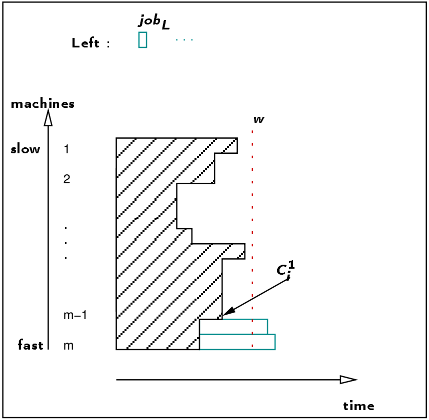
Proof of the first reject condition

First reject condition

$\neg(L < m) \Longrightarrow w < C^*_{max}$

Proof :

- By construction of *Left* :
  $\forall j \in Left, \forall i \in \{1, .., m\}, C_i^1 + \frac{job_j}{s_i} > w$ (1)
- Now suppose $L \geq m$
- using the classical Graham bound, we get $C^*_{max} \geq \frac{W}{\sum_{i=1}^{m} s_i} > w$



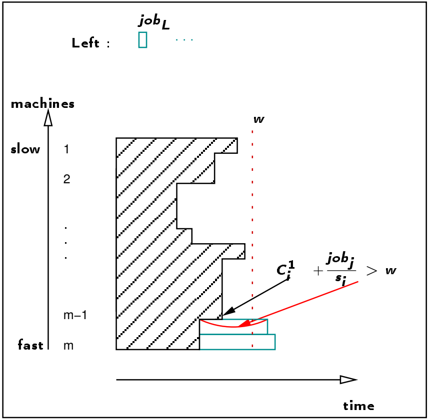
Proof of the first reject condition

First reject condition

$\neg(L < m) \implies w < C^*_{max}$

Proof :

- By construction of *Left* :
  $\forall j \in Left, \forall i \in \{1, .., m\}, C_i^1 + \frac{job_j}{s_i} > w$ (1)
- Now suppose $L \geq m$
- using the classical Graham bound, we get $C^*_{max} \geq \frac{W}{\sum_{i=1}^{m} s_i} > w$



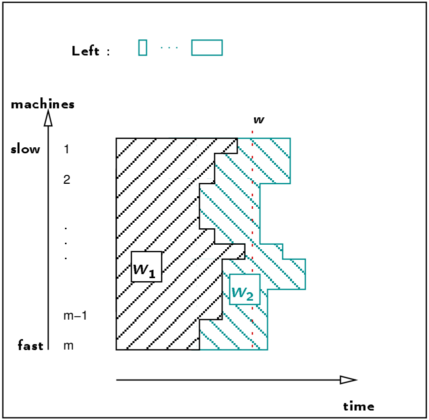
Proof of the first reject condition

First reject condition

$\neg(L < m) \implies w < C^*_{max}$

Proof :

- By construction of *Left* :
  $\forall j \in Left, \forall i \in \{1, .., m\}, C_i^1 + \frac{job_j}{s_i} > w$ (1)
- Now suppose $L \geq m$
- using the classical Graham bound, we get $C^*_{max} \geq \frac{W}{\sum_{i=1}^{m} s_i} > w$



Proof of the first reject condition

First reject condition

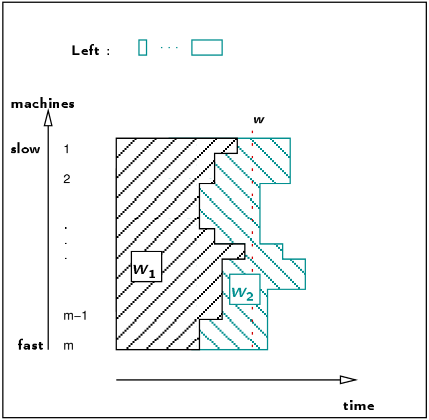
$\neg(L < m) \Longrightarrow w < C^*_{max}$

Proof :

- By construction of *Left* :
  $\forall j \in Left, \forall i \in \{1, .., m\}, C_i^1 + \frac{job_j}{s_i} > w$ (1)
- Now suppose $L \geq m$

$$\begin{aligned} W = W_1 + W_2 &\geq W_1 + m job_L \\ &= \sum_{i=1}^{m} (C_i^1 s_i + job_L) \\ &> (\sum_{i=1}^{m} s_i) w \quad (using\ (1)) \end{aligned}$$

- using the classical Graham bound, we get $C^*_{max} \geq \frac{W}{\sum_{i=1}^{m} s_i} > w$



Proof of the first reject condition

First reject condition

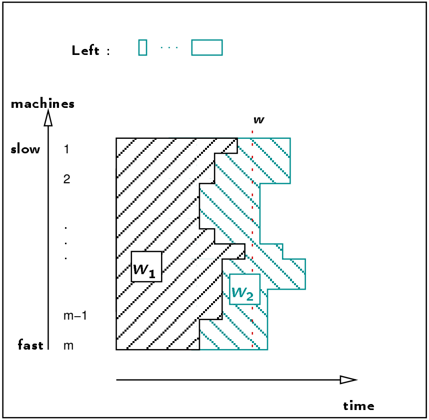
$\neg(L < m) \Longrightarrow w < C^*_{max}$

Proof :

- By construction of *Left* :
  $\forall j \in Left, \forall i \in \{1, .., m\}, C_i^1 + \frac{job_j}{s_i} > w$ (1)
- Now suppose $L \geq m$

$$
\begin{aligned}
W = W_1 + W_2 &\geq W_1 + m job_L \\
&= \sum_{i=1}^{m} (C_i^1 s_i + job_L) \\
&> (\sum_{i=1}^{m} s_i) w \quad \text{(using (1))}
\end{aligned}
$$

- using the classical Graham bound, we get
  $C^*_{max} \geq \frac{W}{\sum_{i=1}^{m} s_i} > w$



Proof of the second reject condition

Second reject condition

$$\neg(\forall j \in \{1,..,L\}, \frac{job_j}{s_{m-j+1}} \leq \frac{w}{2}) \Longrightarrow w < C^*_{max}$$

Proof :

- Suppose $\exists j_0, \frac{job_{j_0}}{s_{m-j_0+1}} > \frac{w}{2}$ (2) (let's take the first such j_0)
- Let $I' = \{j \in \{1,..,n\}, p_j \geq p_{job_{j_0}}\}$
- We will prove that scheduling tasks in I' is impossible in w units of time, proving thus that the original set of tasks $\{1,..,n\}$ is also impossible in w.

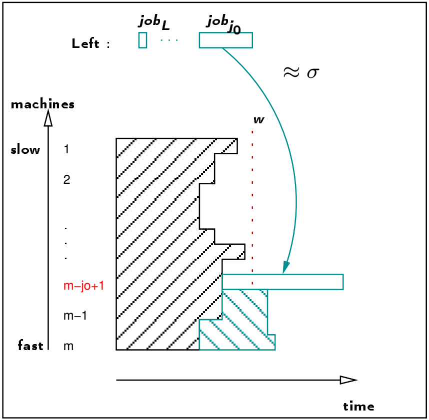
Proof of the second reject condition

Second reject condition

$\neg(\forall j \in \{1,..,L\}, \frac{job_j}{s_{m-j+1}} \leq \frac{w}{2}) \implies w < C^*_{max}$

Proof :

- Suppose $\exists j_0$, $\frac{job_{j_0}}{s_{m-j_0+1}} > \frac{w}{2}$ (2) (let's take the first such j_0)
- Let $I' = \{j \in \{1,..,n\}, p_j \geq p_{job_{j_0}}\}$
- We will prove that scheduling tasks in I' is impossible in w units of time, proving thus that the original set of tasks $\{1,..,n\}$ is also impossible in w.



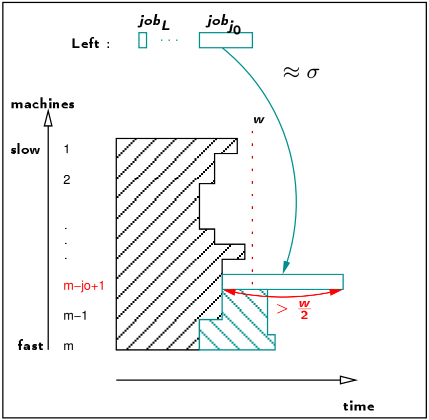
Proof of the second reject condition

Second reject condition

$\neg(\forall j \in \{1, .., L\}, \frac{job_j}{s_{m-j+1}} \leq \frac{w}{2}) \Longrightarrow w < C^*_{max}$

Proof :

- Suppose $\exists j_0$, $\frac{job_{j_0}}{s_{m-j_0+1}} > \frac{w}{2}$ (2) (let's take the first such j_0)
- Let $I' = \{j \in \{1, .., n\}, p_j \geq p_{job_{j_0}}\}$
- We will prove that scheduling tasks in I' is impossible in w units of time, proving thus that the original set of tasks $\{1, .., n\}$ is also impossible in w.



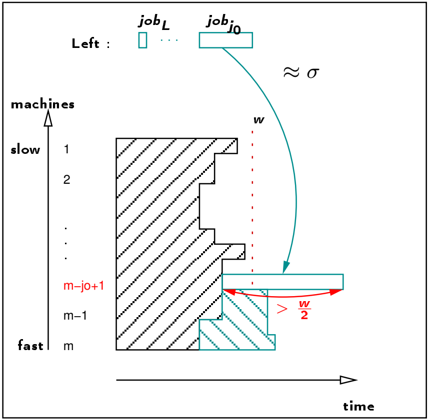
Proof of the second reject condition

Second reject condition

$\neg(\forall j \in \{1,..,L\}, \frac{job_j}{s_{m-j+1}} \leq \frac{w}{2}) \Longrightarrow w < C^*_{max}$

Proof :

- Suppose $\exists j_0, \frac{job_{j_0}}{s_{m-j_0+1}} > \frac{w}{2}$ (2) (let's take the first such j_0)
- Let $I' = \{j \in \{1,..,n\}, p_j \geq p_{job_{j_0}}\}$
- We will prove that scheduling tasks in I' is impossible in w units of time, proving thus that the original set of tasks $\{1,..,n\}$ is also impossible in w.



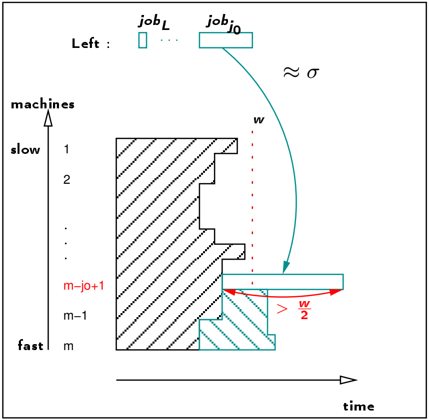
Proof of the second reject condition

Second reject condition

$$\neg(\forall j \in \{1, .., L\}, \frac{job_j}{s_{m-j+1}} \leq \frac{w}{2}) \Longrightarrow w < C^*_{max}$$

Proof :

- Suppose $\exists j_0, \frac{job_{j_0}}{s_{m-j_0+1}} > \frac{w}{2}$ (2) (let's take the first such j_0)
- Let $I' = \{j \in \{1, .., n\}, p_j \geq p_{job_{j_0}}\}$
- We will prove that scheduling tasks in I' is impossible in w units of time, proving thus that the original set of tasks $\{1, .., n\}$ is also impossible in w.



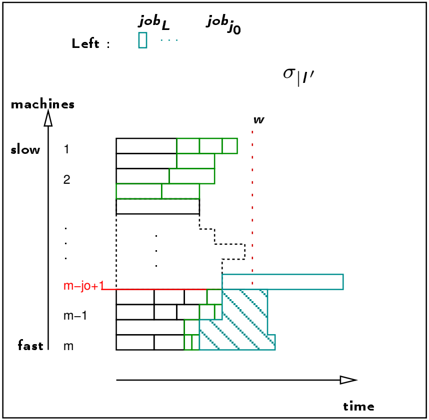
Proof of the second reject condition

Let's consider $\sigma_{|I'}$:

- On "slow" machines $i \in Slow = \{1, .., m - j_0 + 1\}$, (2) $\Longrightarrow$ 0 *or* 1 task per machine
- On "fast" machines $i \in Fast = \{m - j_0, .., m\}$, at least 1 task from phase 1, and 1 task from phase 2

To show that I' is impossible, we proved that :

- the total work W_{slow} scheduled on the "slow" machines is maximal (optimal)
- the remaining work scheduled on W_{fast} is too large ($W_{fast} > \sum_{i=m-j_0+1}^{m} s_i w$)



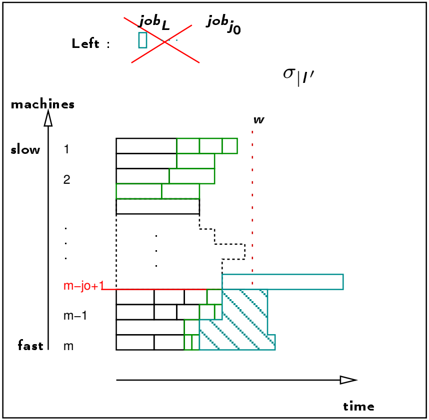
Proof of the second reject condition

Let's consider $\sigma_{|I'}$:

- On "slow" machines $i \in Slow = \{1, .., m - j_0 + 1\}$, (2) $\Longrightarrow$ 0 *or* 1 task per machine
- On "fast" machines $i \in Fast = \{m - j_0, .., m\}$, at least 1 task from phase 1, and 1 task from phase 2

To show that I' is impossible, we proved that :

- the total work W_{slow} scheduled on the "slow" machines is maximal (optimal)
- the remaining work scheduled on W_{fast} is too large ($W_{fast} > \sum_{i=m-j_0+1}^{m} s_i w$)



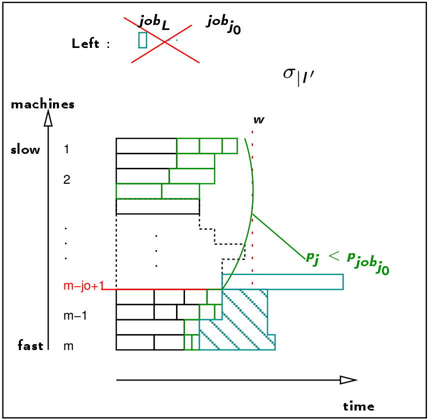
Proof of the second reject condition

Let's consider $\sigma_{|I'}$:

- On "slow" machines $i \in Slow = \{1, .., m - j_0 + 1\}$, (2) $\Longrightarrow$ 0 *or* 1 task per machine
- On "fast" machines $i \in Fast = \{m - j_0, .., m\}$, at least 1 task from phase 1, and 1 task from phase 2

To show that I' is impossible, we proved that :

- the total work W_{slow} scheduled on the "slow" machines is maximal (optimal)
- the remaining work scheduled on W_{fast} is too large ($W_{fast} > \sum_{i=m-j_0+1}^{m} s_i w$)



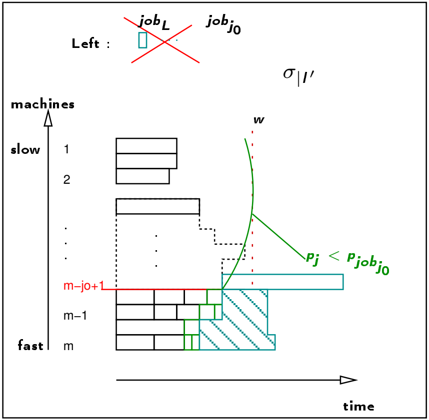
Proof of the second reject condition

Let's consider $\sigma_{|I'}$:

- On "slow" machines $i \in Slow = \{1, .., m - j_0 + 1\}$, (2) $\Longrightarrow$ 0 *or* 1 task per machine
- On "fast" machines $i \in Fast = \{m - j_0, .., m\}$, at least 1 task from phase 1, and 1 task from phase 2

To show that I' is impossible, we proved that :

- the total work W_{slow} scheduled on the "slow" machines is maximal (optimal)
- the remaining work scheduled on W_{fast} is too large ($W_{fast} > \sum_{i=m-j_0+1}^{m} s_i w$)



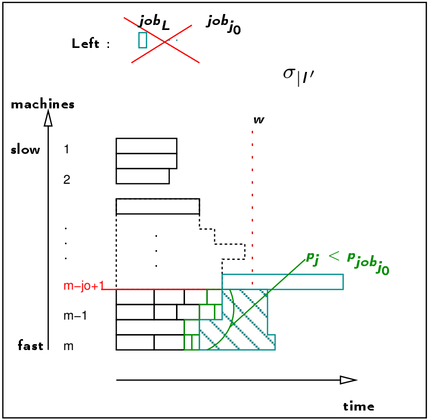
Proof of the second reject condition

Let's consider $\sigma_{|I'}$:

- On "slow" machines $i \in Slow = \{1, .., m - j_0 + 1\}$, (2) $\Longrightarrow$ 0 *or* 1 task per machine
- On "fast" machines $i \in Fast = \{m - j_0, .., m\}$, at least 1 task from phase 1, and 1 task from phase 2

To show that I' is impossible, we proved that :

- the total work W_{slow} scheduled on the "slow" machines is maximal (optimal)
- the remaining work scheduled on W_{fast} is too large ($W_{fast} > \sum_{i=m-j_0+1}^{m} s_i w$)



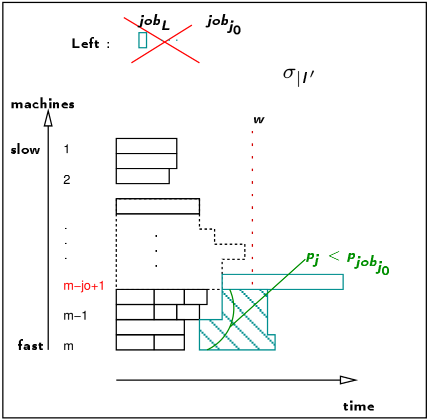
Proof of the second reject condition

Let's consider $\sigma_{|I'}$:

- On "slow" machines $i \in Slow = \{1, .., m - j_0 + 1\}$, (2) $\Longrightarrow$ 0 *or* 1 task per machine
- On "fast" machines $i \in Fast = \{m - j_0, .., m\}$, at least 1 task from phase 1, and 1 task from phase 2

To show that I' is impossible, we proved that :

- the total work W_{slow} scheduled on the "slow" machines is maximal (optimal)
- the remaining work scheduled on W_{fast} is too large ($W_{fast} > \sum_{i=m-j_0+1}^{m} s_i w$)



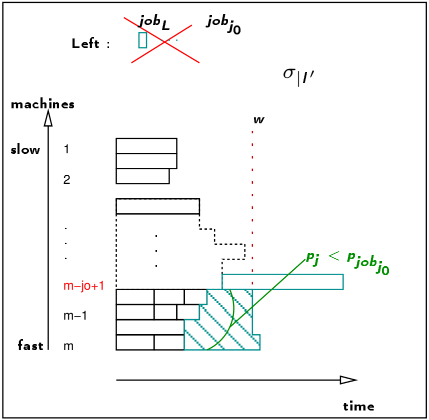
Proof of the second reject condition

Let's consider $\sigma_{|I'}$:

- On "slow" machines $i \in Slow = \{1, .., m - j_0 + 1\}$, (2) $\Longrightarrow$ 0 *or* 1 task per machine
- On "fast" machines $i \in Fast = \{m - j_0, .., m\}$, at least 1 task from phase 1, and 1 task from phase 2

To show that I' is impossible, we proved that :

- the total work W_{slow} scheduled on the "slow" machines is maximal (optimal)
- the remaining work scheduled on W_{fast} is too large ($W_{fast} > \sum_{i=m-j_0+1}^{m} s_i w$)



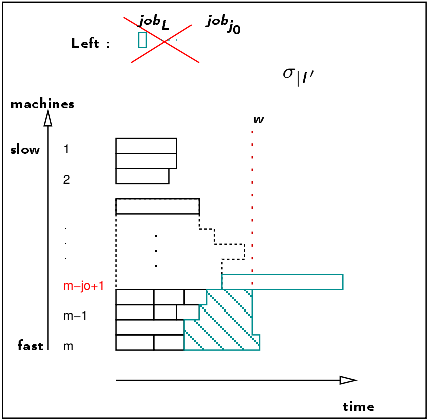
Proof of the second reject condition

Let's consider $\sigma_{|I'}$:

- On "slow" machines $i \in Slow = \{1, .., m - j_0 + 1\}$, (2) $\Longrightarrow$ 0 *or* 1 task per machine
- On "fast" machines $i \in Fast = \{m - j_0, .., m\}$, at least 1 task from phase 1, and 1 task from phase 2

To show that I' is impossible, we proved that :

- the total work W_{slow} scheduled on the "slow" machines is maximal (optimal)
- the remaining work scheduled on W_{fast} is too large ($W_{fast} > \sum_{i=m-j_0+1}^{m} s_i w$)



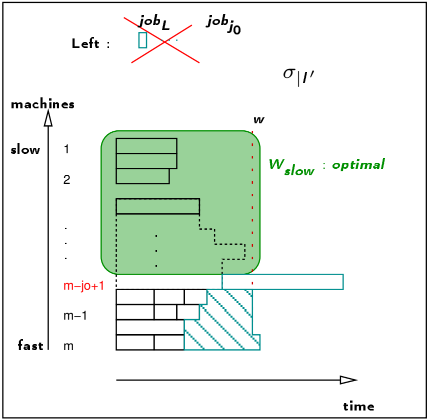
Proof of the second reject condition

Let's consider $\sigma_{|I'}$:

- On "slow" machines $i \in Slow = \{1, .., m - j_0 + 1\}$, (2) $\Longrightarrow$ 0 *or* 1 task per machine
- On "fast" machines $i \in Fast = \{m - j_0, .., m\}$, at least 1 task from phase 1, and 1 task from phase 2

To show that I' is impossible, we proved that :

- the total work W_{slow} scheduled on the "slow" machines is maximal (optimal)
- the remaining work scheduled on W_{fast} is too large ($W_{fast} > \sum_{i=m-j_0+1}^{m} s_i w$)



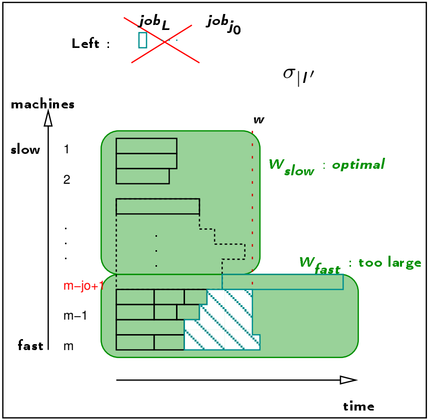
Proof of the second reject condition

Let's consider $\sigma_{|I'}$:

- On "slow" machines $i \in Slow = \{1, .., m - j_0 + 1\}$, (2) $\Longrightarrow$ 0 *or* 1 task per machine
- On "fast" machines $i \in Fast = \{m - j_0, .., m\}$, at least 1 task from phase 1, and 1 task from phase 2

To show that I' is impossible, we proved that :

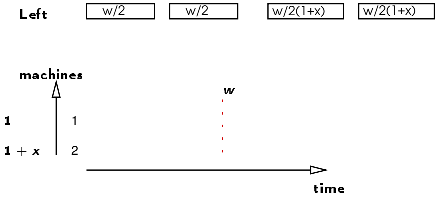
- the total work W_{slow} scheduled on the "slow" machines is maximal (optimal)
- the remaining work scheduled on W_{fast} is too large ($W_{fast} > \sum_{i=m-j_0+1}^{m} s_i w$)



Tightness of the bound

Theorem

$\forall \epsilon > 0, \exists I / C_{max} \geq (\frac{3}{2} - \epsilon) C^*_{max}$

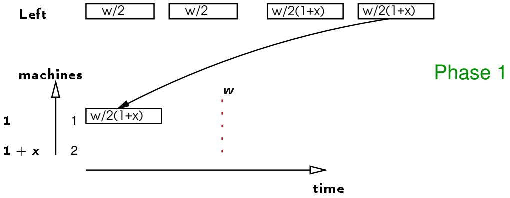


Notice that we could modify the phase 2 of our algorithm to avoid this case, but ...

Tightness of the bound

Theorem

$\forall \epsilon > 0, \exists I / C_{max} \geq (\frac{3}{2} - \epsilon) C^*_{max}$

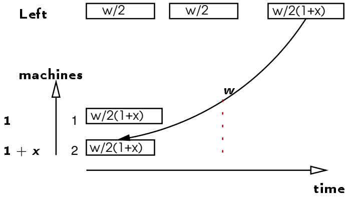


Notice that we could modify the phase 2 of our algorithm to avoid this case, but ...

Tightness of the bound

Theorem

$\forall \epsilon > 0, \exists I / C_{max} \geq (\frac{3}{2} - \epsilon) C^*_{max}$



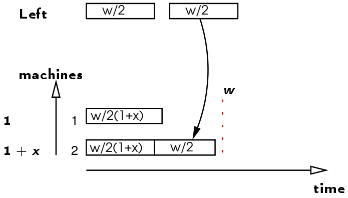
Phase 1

Notice that we could modify the phase 2 of our algorithm to avoid this case, but ...

Tightness of the bound

Theorem

$\forall \epsilon > 0, \exists I / C_{max} \geq (\frac{3}{2} - \epsilon) C^*_{max}$



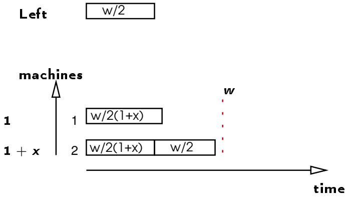
Phase 1

Notice that we could modify the phase 2 of our algorithm to avoid this case, but ...

Tightness of the bound

Theorem

$\forall \epsilon > 0, \exists I / C_{max} \geq (\frac{3}{2} - \epsilon) C^*_{max}$



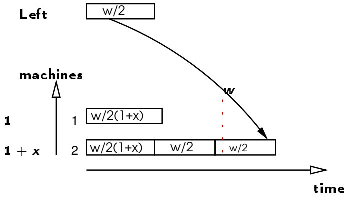
Phase 2

Notice that we could modify the phase 2 of our algorithm to avoid this case, but ...

Tightness of the bound

Theorem

$\forall\epsilon > 0, \exists I / C_{max} \geq (\frac{3}{2} - \epsilon) C^*_{max}$



Phase 2

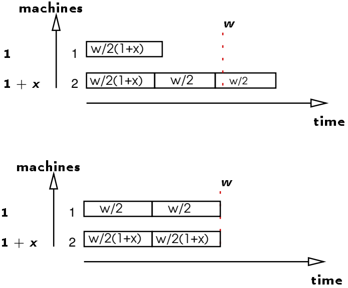
Notice that we could modify the phase 2 of our algorithm to avoid this case, but ...

Tightness of the bound

Theorem

$$\forall \epsilon > 0, \exists I / C_{max} \geq (\frac{3}{2} - \epsilon) C^*_{max}$$

Left



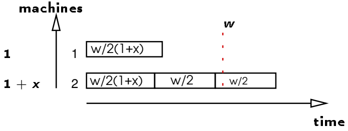
Notice that we could modify the phase 2 of our algorithm to avoid this case, but ...

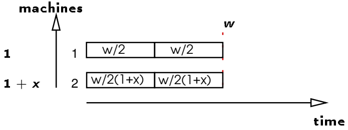
Tightness of the bound

Theorem

$\forall \epsilon > 0, \exists I / C_{max} \geq (\frac{3}{2} - \epsilon) C^*_{max}$

Left





$$\frac{C_{max}}{C^*_{max}} = \frac{(\frac{w}{2} + \frac{w}{1+x})}{w} = \frac{1}{2} + \frac{1}{1+x}$$

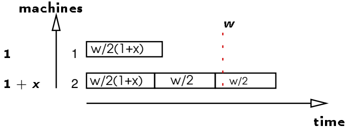
Notice that we could modify the phase 2 of our algorithm to avoid this case, but ...

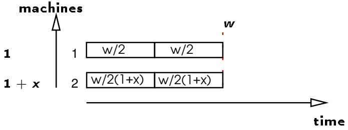
Tightness of the bound

Theorem

$$\forall \epsilon > 0, \exists I / C_{max} \geq (\tfrac{3}{2} - \epsilon) C^*_{max}$$

Left





$$\frac{C_{max}}{C^*_{max}} = \frac{\left(\frac{w}{2} + \frac{w}{1+x}\right)}{w} = \frac{1}{2} + \frac{1}{1+x}$$

Notice that we could modify the phase 2 of our algorithm to avoid this case, but ...

Conclusion

For the problem $Q||C_{max}$,

- there is a fast $\frac{3}{2}$ approximation
- there is the PTAS of Hochbaum and Shmoys [SIAM JoC 88]

Thank you for your attention !