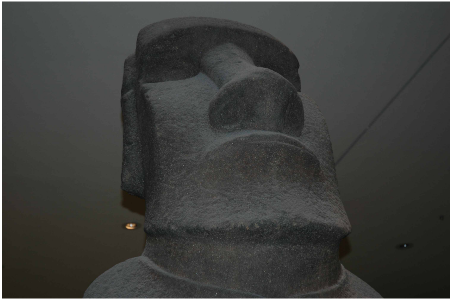
A fast 5/2 approximation for hierarchical scheduling

Marin Bougeret, Pierre-François Dutot, Christina Otte, Klaus Jansen and Denis Trystram

LIG laboratory (MOAIS Team), Grenoble, France



Context

Problem statement

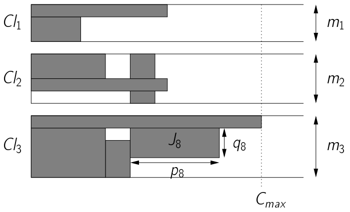
Multiple Cluster Scheduling Problem (MCSP):

Input:

- k clusters (cluster Cl_i owns m_i machines)
- n independent parallel jobs (job J_j requires q_j machines **on the same cluster** during p_j units of time)

Objective:

- schedule all the jobs minimizing the makespan C_{max}



Problem statement

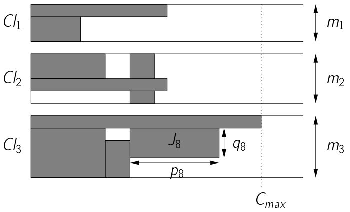
Multiple Cluster Scheduling Problem (MCSP):

Input:

- k clusters (cluster Cl_i owns m_i machines)
- n independent parallel jobs (job J_j requires q_j machines **on the same cluster** during p_j units of time)

Objective:

- schedule all the jobs minimizing the makespan C_{max}



Problem statement

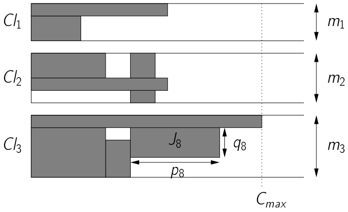
Multiple Cluster Scheduling Problem (MCSP):

Input:

- k clusters (cluster Cl_i owns m_i machines)
- n independent parallel jobs (job J_j requires q_j machines **on the same cluster** during p_j units of time)

Objective:

- schedule all the jobs minimizing the makespan C_{max}



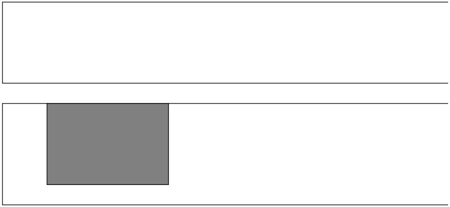
Motivation of the Multiple Cluster model

What means this model ?

- parallel job = schedule of sequential jobs with dependencies without communication costs

$\Rightarrow$ communication costs between machines of the same cluster are neglected

$\Rightarrow$ communication costs between machines of different clusters are unbounded, hence the constraint of using q_j machines of the same cluster when scheduling one job



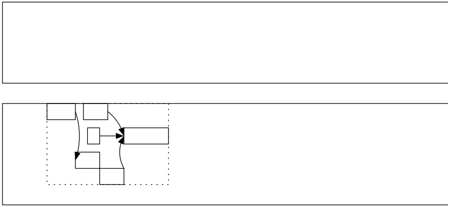
Motivation of the Multiple Cluster model

What means this model ?

- parallel job = schedule of sequential jobs with dependencies without communication costs

⇒ communication costs between machines of the same cluster are neglected

⇒ communication costs between machines of different clusters are unbounded, hence the constraint of using q_j machines of the same cluster when scheduling one job



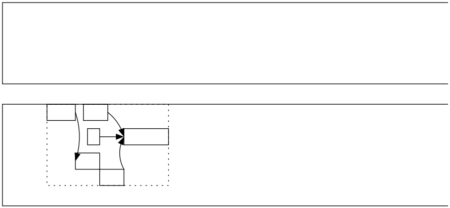
Motivation of the Multiple Cluster model

What means this model ?

- parallel job = schedule of sequential jobs with dependencies without communication costs

$\Rightarrow$ communication costs between machines of the same cluster are neglected

$\Rightarrow$ communication costs between machines of different clusters are unbounded, hence the constraint of using q_j machines of the same cluster when scheduling one job



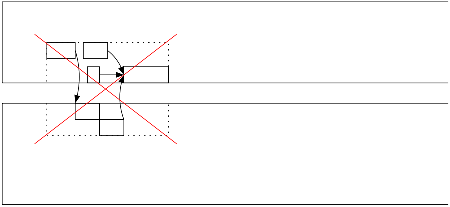
Motivation of the Multiple Cluster model

What means this model ?

- parallel job = schedule of sequential jobs with dependencies without communication costs

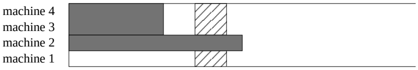
$\Rightarrow$ communication costs between machines of the same cluster are neglected

$\Rightarrow$ communication costs between machines of different clusters are unbounded, hence the constraint of using q_j machines of **the same** cluster when scheduling one job

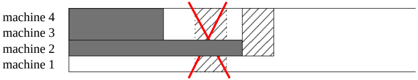


Related problems

- parallel (rigid) job scheduling



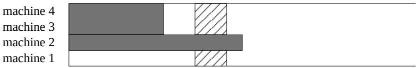
- Rectangle packing = parallel(rigid) job continuous scheduling



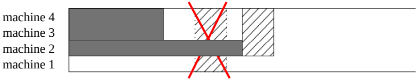
- Algorithms for non continuous case generally not apply to continuous case but ..
- Approximation ratios for continuous case may not apply to non contiguous case (as $C^*_{max} \leq C^{cont}_{max}$ *)
- However, approximation ratio for continuous case often apply to non continuous case

Related problems

- parallel (rigid) job scheduling



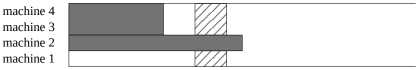
- Rectangle packing = parallel(rigid) job continuous scheduling



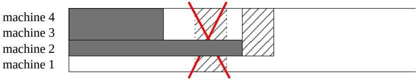
  - Algorithms for non continuous case generally not apply to continuous case but ..
  - Approximation ratios for continuous case may not apply to non contiguous case (as $C^*_{max} \leq C^{cont}_{max}{}^*$)
  - However, approximation ratio for continuous case often apply to non continuous case

Related problems

- parallel (rigid) job scheduling



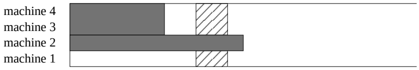
- Rectangle packing = parallel(rigid) job continuous scheduling



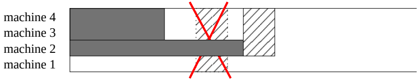
  - Algorithms for non continuous case generally not apply to continuous case but ..
  - Approximation ratios for continuous case may not apply to non contiguous case (as $C^*_{max} \leq C^{cont}_{max}$ *)
  - However, approximation ratio for continuous case often apply to non continuous case

Related problems

- parallel (rigid) job scheduling



- Rectangle packing = parallel(rigid) job continuous scheduling



  - Algorithms for non continuous case generally not apply to continuous case but ..
  - Approximation ratios for continuous case may not apply to non contiguous case (as $C^*_{max} \leq C^{cont}_{max}\ ^*$)
  - However, approximation ratio for continuous case often apply to non continuous case

Related problems

MCSP and strip packing

- MCSP + continuous constraint = Multiple Strip Packing
- For k = 1 Cluster :
  - MCSP is classical parallel job scheduling (List Scheduling ratio $2 - \frac{1}{m}$)
  - MCSP + continuous constraint = Strip Packing (the 2 ratio required more efforts!)

As shown in [STY08] List Scheduling is arbitrarily bad on MCSP

Related problems

MCSP and strip packing

- MCSP + continuous constraint = Multiple Strip Packing
- For k = 1 Cluster :
  - MCSP is classical parallel job scheduling
    (List Scheduling ratio $2 - \frac{1}{m}$)
  - MCSP + continuous constraint = Strip Packing
    (the 2 ratio required more efforts!)

As shown in [STY08] List Scheduling is arbitrarily bad on MCSP

Related problems

MCSP and strip packing

- MCSP + continuous constraint = Multiple Strip Packing
- For k = 1 Cluster :
  - MCSP is classical parallel job scheduling (List Scheduling ratio $2 - \frac{1}{m}$)
  - MCSP + continuous constraint = Strip Packing (the 2 ratio required more efforts!)

As shown in [STY08] List Scheduling is arbitrarily bad on MCSP

Related problems

MCSP and strip packing

- MCSP + continuous constraint = Multiple Strip Packing
- For k = 1 Cluster :
  - MCSP is classical parallel job scheduling (List Scheduling ratio $2 - \frac{1}{m}$)
  - MCSP + continuous constraint = Strip Packing (the 2 ratio required more efforts!)

As shown in [STY08] List Scheduling is arbitrarily bad on MCSP

Related problems

MCSP and strip packing

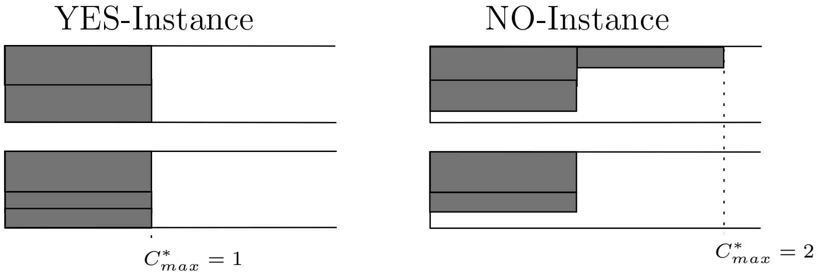
- MCSP + continuous constraint = Multiple Strip Packing
- For k = 1 Cluster :
  - MCSP is classical parallel job scheduling (List Scheduling ratio $2 - \frac{1}{m}$)
  - MCSP + continuous constraint = Strip Packing (the 2 ratio required more efforts!)

As shown in [STY08] List Scheduling is arbitrarily bad on MCSP

Negative result for MCSP

- MCSP is 2-inapproximable unless $P = NP$, even for $k = 2$ clusters having the same size

Using a gap reduction [Zhu06] from the 2-partition problem:



Main positive results for MCSP

Ratio	Remarks	Hypothesis	Source
3	decentralized (and fast) algorithm		[STY08]
$2+\epsilon$	requires solving $Q\|\|C_{max}$ with a ratio $1+\frac{\epsilon}{2}$	every job fits everywhere, adapted from [YHZ09]	this paper
$\frac{5}{2}$	fast, also applies for continuous case	every job fits everywhere	this talk

Remark: Solving $Q||C_{max}$ using a classical PTAS is in $\Omega(n^{60})$, as our $\frac{5}{2}$ approximation is in $\approx knlog^2(n)$

Main positive results for MCSP

Ratio	Remarks	Hypothesis	Source
3	decentralized (and fast) algorithm		[STY08]
$2+\epsilon$	requires solving $Q\|\|C_{max}$ with a ratio $1+\frac{\epsilon}{2}$	every job fits everywhere, adapted from [YHZ09]	this paper
$\frac{5}{2}$	fast, also applies for continuous case	every job fits everywhere	this talk

Remark: Solving $Q||C_{max}$ using a classical PTAS is in $\Omega(n^{60})$, as our $\frac{5}{2}$ approximation is in $\approx knlog^2(n)$

Main positive results for MCSP

Ratio	Remarks	Hypothesis	Source
3	decentralized (and fast) algorithm		[STY08]
$2+\epsilon$	requires solving $Q\|\|C_{max}$ with a ratio $1+\frac{\epsilon}{2}$	every job fits everywhere, adapted from [YHZ09]	this paper
$\frac{5}{2}$	fast, also applies for continuous case	every job fits everywhere	this talk

Remark: Solving $Q||C_{max}$ using a classical PTAS is in $\Omega(n^{60})$, as our $\frac{5}{2}$ approximation is in $\approx knlog^2(n)$

Main positive results for MCSP

Ratio	Remarks	Hypothesis	Source
3	decentralized (and fast) algorithm		[STY08]
$2+\epsilon$	requires solving $Q\|\|C_{max}$ with a ratio $1+\frac{\epsilon}{2}$	every job fits everywhere, adapted from [YHZ09]	this paper
$\frac{5}{2}$	fast, also applies for continuous case	every job fits everywhere	this talk

Remark: Solving $Q||C_{max}$ using a classical PTAS is in $\Omega(n^{60})$, as our $\frac{5}{2}$ approximation is in $\approx knlog^2(n)$

Main positive results for MCSP

Ratio	Remarks	Hypothesis	Source
3	decentralized (and fast) algorithm		[STY08]
$2+\epsilon$	requires solving $Q\|\|C_{max}$ with a ratio $1+\frac{\epsilon}{2}$	every job fits everywhere, adapted from [YHZ09]	this paper
$\frac{5}{2}$	fast, also applies for continuous case	every job fits everywhere	this talk

Remark: Solving $Q||C_{max}$ using a classical PTAS is in $\Omega(n^{60})$, as our $\frac{5}{2}$ approximation is in $\approx knlog^2(n)$

Using the dual approximation technique [HS87]

Specification

Given an input I, and a guess v of the optimal value, the algorithm:

- schedules I with $C_{max} \leq \frac{5v}{2}$
- or REJECTS v, implying then that $v < C^*_{max}$

Thus, using a binary search on v on the interval $[|0, np_{max}|] \ni C^*_{max}$ we get a $\frac{5}{2}$-approximation

Using the dual approximation technique [HS87]

Specification

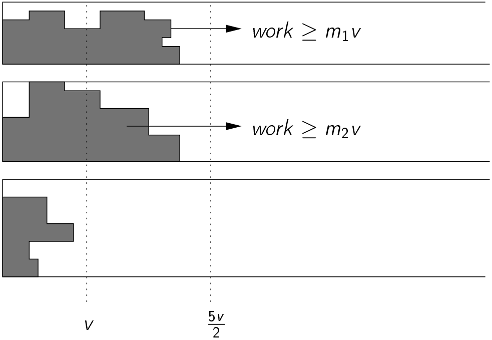
Given an input I, and a guess v of the optimal value, the algorithm:

- schedules I with $C_{max} \leq \frac{5v}{2}$
- or REJECTS v, implying then that $v < C^*_{max}$

Thus, using a binary search on v on the interval $[|0, np_{max}|] \ni C^*_{max}$ we get a $\frac{5}{2}$-approximation

Using the dual approximation technique

We fill the clusters from the smallest to the largest, trying to schedule an "optimal work" in every cluster

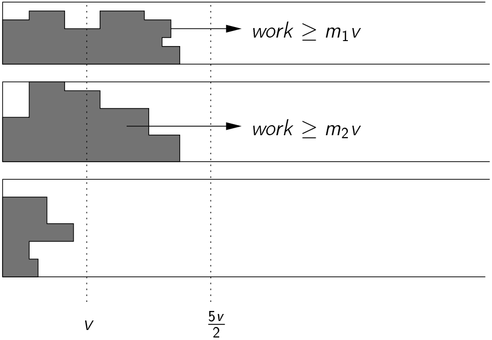


Notation

Let $W(X) = \sum_{j \in X} p_j q_j$ denote the *work* of the set of jobs X.

Using the dual approximation technique

We fill the clusters from the smallest to the largest, trying to schedule an "optimal work" in every cluster



Notation

Let $W(X) = \sum_{j \in X} p_j q_j$ denote the *work* of the set of jobs X.

Let us dream

A_{select}

Let us suppose that we have an algorithm A_{select} such that:

- if there is enough remaining jobs I' (I' is the set of unscheduled jobs)
- then A_{select} returns $X \subset I'$ such that
  - X can be scheduled in $\frac{5v}{2}$
  - $W(X) \geq m_i v$

Then we would directly have the $\frac{5}{2}$ approximation.

Let us dream

A_{select}

Let us suppose that we have an algorithm A_{select} such that:

- if there is enough remaining jobs I' (I' is the set of unscheduled jobs)
- then A_{select} returns $X \subset I'$ such that
  - X can be scheduled in $\frac{5v}{2}$
  - $W(X) \geq m_i v$

Then we would directly have the $\frac{5}{2}$ approximation.

Let us dream

A_{select}

Let us suppose that we have an algorithm A_{select} such that:

- if there is enough remaining jobs I' (I' is the set of unscheduled jobs)
- then A_{select} returns $X \subset I'$ such that
  - X can be scheduled in $\frac{5v}{2}$
  - $W(X) \geq m_i v$

Then we would directly have the $\frac{5}{2}$ approximation.

Let us dream

A_{select}

Let us suppose that we have an algorithm A_{select} such that:

- if there is enough remaining jobs I' (I' is the set of unscheduled jobs)
- then A_{select} returns $X \subset I'$ such that
  - X can be scheduled in $\frac{5v}{2}$
  - $W(X) \geq m_i v$

Then we would directly have the $\frac{5}{2}$ approximation.

Let us dream

A_{select}

Let us suppose that we have an algorithm A_{select} such that:

- if there is enough remaining jobs I' (I' is the set of unscheduled jobs)
- then A_{select} returns $X \subset I'$ such that
  - X can be scheduled in $\frac{5v}{2}$
  - $W(X) \geq m_i v$

Then we would directly have the $\frac{5}{2}$ approximation.

Let us dream

A_{select}

Let us suppose that we have an algorithm A_{select} such that:

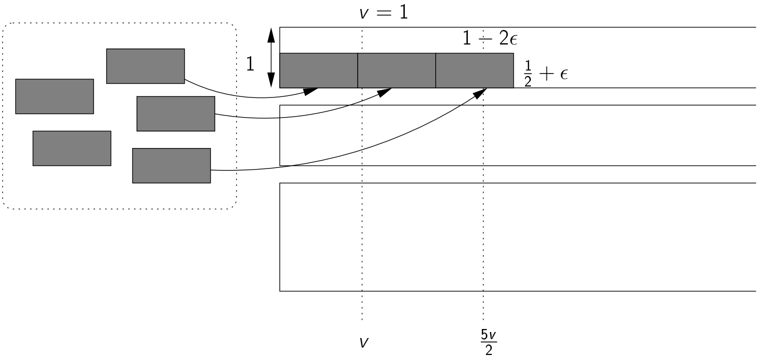
- if there is enough remaining jobs I' (I' is the set of unscheduled jobs)
- then A_{select} returns $X \subset I'$ such that
  - X can be scheduled in $\frac{5v}{2}$
  - $W(X) \geq m_i v$

Then we would directly have the $\frac{5}{2}$ approximation.

Bad news

The previous algorithm does not exist:

$I =$



Proposed solution

A weaker A_{select}

Thus, A_{select} requires constraints ensuring that such a X set exists.

- instead of: if there is enough remaining jobs I' then A_{select} returns X as specified before
- we have: if C_1 then A_{select} returns X as specified before

Road map

What we have to do is:

- define C_1
- if C_1 is true, prove that X can be scheduled in $\frac{5v}{2}$
- otherwise, prove that it is "obvious" to finish packing all the jobs of I' in all the remaining empty clusters

Proposed solution

A weaker A_{select}

Thus, A_{select} requires constraints ensuring that such a X set exists.

- instead of: if there is enough remaining jobs I' then A_{select} returns X as specified before
- we have: if C_1 then A_{select} returns X as specified before

Road map

What we have to do is:

- define C_1
- if C_1 is true, prove that X can be scheduled in $\frac{5v}{2}$
- otherwise, prove that it is "obvious" to finish packing all the jobs of I' in all the remaining empty clusters

Proposed solution

A weaker A_{select}

Thus, A_{select} requires constraints ensuring that such a X set exists.

- instead of: if there is enough remaining jobs I' then A_{select} returns X as specified before
- we have: if C_1 then A_{select} returns X as specified before

Road map

What we have to do is:

- define C_1
- if C_1 is true, prove that X can be scheduled in $\frac{5v}{2}$
- otherwise, prove that it is "obvious" to finish packing all the jobs of I' in all the remaining empty clusters

Proposed solution

A weaker A_{select}

Thus, A_{select} requires constraints ensuring that such a X set exists.

- instead of: if there is enough remaining jobs I' then A_{select} returns X as specified before
- we have: if C_1 then A_{select} returns X as specified before

Road map

What we have to do is:

- define C_1
- if C_1 is true, prove that X can be scheduled in $\frac{5v}{2}$
- otherwise, prove that it is "obvious" to finish packing all the jobs of I' in all the remaining empty clusters

Proposed solution

A weaker A_{select}

Thus, A_{select} requires constraints ensuring that such a X set exists.

- instead of: if there is enough remaining jobs I' then A_{select} returns X as specified before
- we have: if C_1 then A_{select} returns X as specified before

Road map

What we have to do is:

- define C_1
- if C_1 is true, prove that X can be scheduled in $\frac{5v}{2}$
- otherwise, prove that it is "obvious" to finish packing all the jobs of I' in all the remaining empty clusters

Proposed solution

A weaker A_{select}

Thus, A_{select} requires constraints ensuring that such a X set exists.

- instead of: if there is enough remaining jobs I' then A_{select} returns X as specified before
- we have: if C_1 then A_{select} returns X as specified before

Road map

What we have to do is:

- define C_1
- if C_1 is true, prove that X can be scheduled in $\frac{5v}{2}$
- otherwise, prove that it is "obvious" to finish packing all the jobs of I' in all the remaining empty clusters

Proposed solution

A weaker A_{select}

Thus, A_{select} requires constraints ensuring that such a X set exists.

- instead of: if there is enough remaining jobs I' then A_{select} returns X as specified before
- we have: if C_1 then A_{select} returns X as specified before

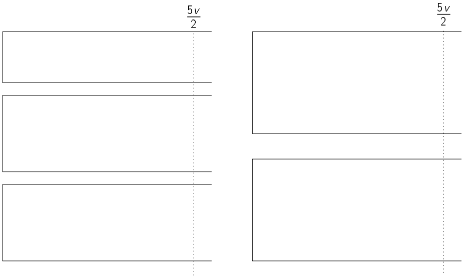
Road map

What we have to do is:

- define C_1
- if C_1 is true, prove that X can be scheduled in $\frac{5v}{2}$
- otherwise, prove that it is "obvious" to finish packing all the jobs of I' in all the remaining empty clusters

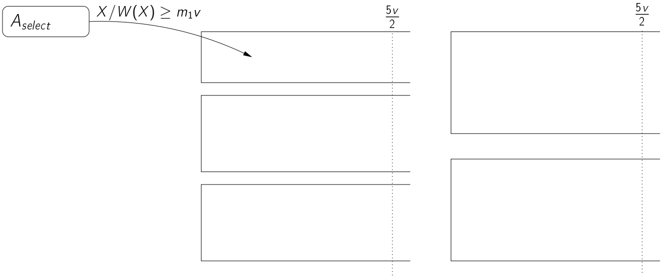
Example

All in all, the algorithm will behave as follows:



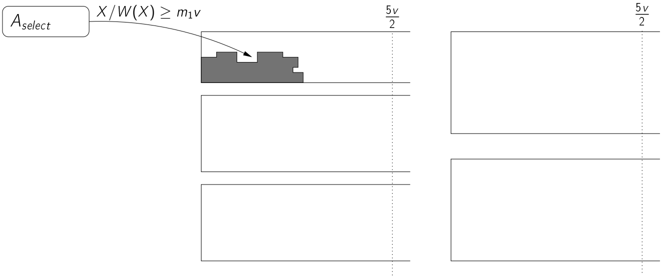
Example

All in all, the algorithm will behave as follows:



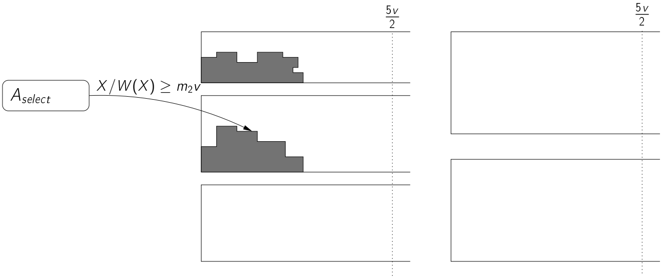
Example

All in all, the algorithm will behave as follows:



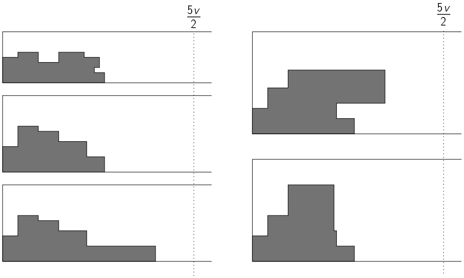
Example

All in all, the algorithm will behave as follows:



Example

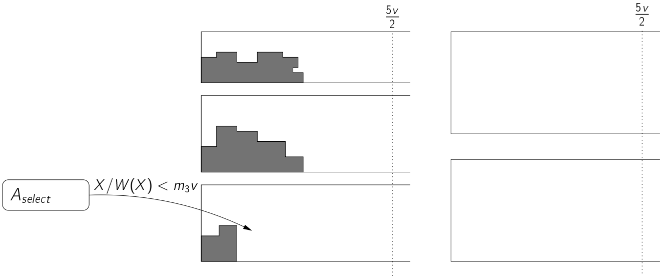
All in all, the algorithm will behave as follows:



REJECT v
as $W(I) > \sum_{i=1}^{k} m_i v$

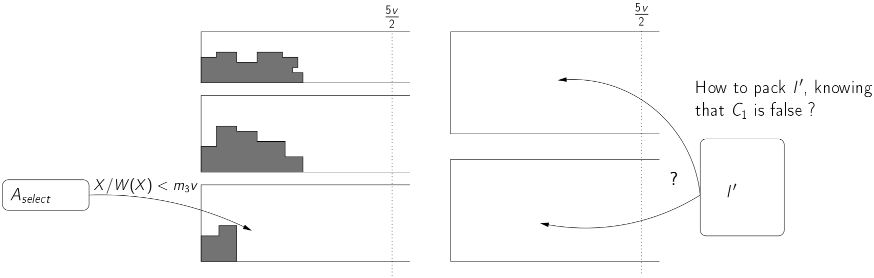
Example

All in all, the algorithm will behave as follows:



Example

All in all, the algorithm will behave as follows:



Toward a "good" A_{select}

An important tool for one strip:

Steinberg's theorem [Ste97]

Steinberg's theorem applied in our case:
if a set X is such that $W(X) \leq \alpha m_i v$, $(\alpha \geq 1)$, then X can be scheduled (continuously) in a rectangular box of size $m_i \times 2\alpha v$.

Remark: Can be proved using a List Scheduling algorithm (leading however to a non-continuous schedule)

Notation

We need the following definitions (given a cluster Cl_i):

- let $L = \{J_J | p_j > \frac{v}{2}\}$ (long jobs)
- let $H_i = \{J_J | q_j > \frac{m_i}{2}\}$ (high jobs)

Toward a "good" A_{select}

An important tool for one strip:

Steinberg's theorem [Ste97]

Steinberg's theorem applied in our case:
if a set X is such that $W(X) \leq \alpha m_i v$, $(\alpha \geq 1)$, then X can be scheduled (continuously) in a rectangular box of size $m_i \times 2\alpha v$.

Remark: Can be proved using a List Scheduling algorithm (leading however to a non-continuous schedule)

Notation

We need the following definitions (given a cluster Cl_i):

- let $L = \{J_J | p_j > \frac{v}{2}\}$ (long jobs)
- let $H_i = \{J_J | q_j > \frac{m_i}{2}\}$ (high jobs)

Toward a "good" A_{select}

An important tool for one strip:

Steinberg's theorem [Ste97]

Steinberg's theorem applied in our case:
if a set X is such that $W(X) \leq \alpha m_i v$, ($\alpha \geq 1$), then X can be scheduled (continuously) in a rectangular box of size $m_i \times 2\alpha v$.

Remark: Can be proved using a List Scheduling algorithm (leading however to a non-continuous schedule)

Notation

We need the following definitions (given a cluster Cl_i):

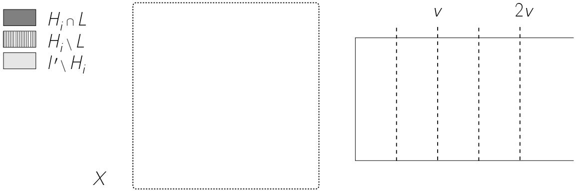
- let $L = \{J_J | p_j > \frac{v}{2}\}$ (long jobs)
- let $H_i = \{J_J | q_j > \frac{m_i}{2}\}$ (high jobs)

Toward a "good" A_{select}

Our A_{select}

Given Cl_i the current empty cluster (initially $X = \emptyset$):

- add to X one rectangle of $L \cap H_i$ (the highest..)
- if there are enough rectangles of $H_i \setminus L$ to reach $2v$, add them to X until reaching $2v$ (we will have $\sum_{j \in X} p_j \leq \frac{5v}{2}$)
- otherwise (we know that few jobs of $H_i \setminus L$ remain), add to X jobs of $I' \setminus H_i$ using a non-increasing work order, stops when $W(X) \geq m_i v$ or when no jobs of $I' \setminus H_i$ remain.

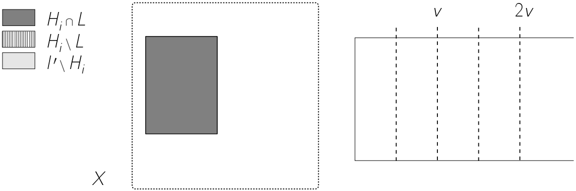


Toward a "good" A_{select}

Our A_{select}

Given Cl_i the current empty cluster (initially $X = \emptyset$):

- add to X one rectangle of $L \cap H_i$ (the highest..)
- if there are enough rectangles of $H_i \setminus L$ to reach $2v$, add them to X until reaching $2v$ (we will have $\sum_{j \in X} p_j \leq \frac{5v}{2}$)
- otherwise (we know that few jobs of $H_i \setminus L$ remain), add to X jobs of $I' \setminus H_i$ using a non-increasing work order, stops when $W(X) \geq m_i v$ or when no jobs of $I' \setminus H_i$ remain.

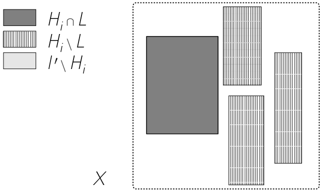


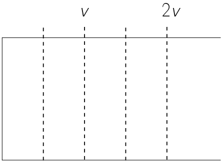
Toward a "good" A_{select}

Our A_{select}

Given Cl_i the current empty cluster (initially $X = \emptyset$):

- add to X one rectangle of $L \cap H_i$ (the highest..)
- if there are enough rectangles of $H_i \setminus L$ to reach $2v$, add them to X until reaching $2v$ (we will have $\sum_{j \in X} p_j \leq \frac{5v}{2}$)
- otherwise (we know that few jobs of $H_i \setminus L$ remain), add to X jobs of $I' \setminus H_i$ using a non-increasing work order, stops when $W(X) \geq m_i v$ or when no jobs of $I' \setminus H_i$ remain.



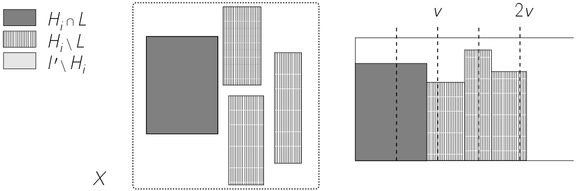


Toward a "good" A_{select}

Our A_{select}

Given Cl_i the current empty cluster (initially $X = \emptyset$):

- add to X one rectangle of $L \cap H_i$ (the highest..)
- if there are enough rectangles of $H_i \setminus L$ to reach $2v$, add them to X until reaching $2v$ (we will have $\sum_{j \in X} p_j \leq \frac{5v}{2}$)
- otherwise (we know that few jobs of $H_i \setminus L$ remain), add to X jobs of $I' \setminus H_i$ using a non-increasing work order, stops when $W(X) \geq m_i v$ or when no jobs of $I' \setminus H_i$ remain.

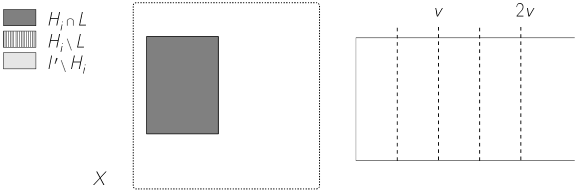


Toward a "good" A_{select}

Our A_{select}

Given Cl_i the current empty cluster (initially $X = \emptyset$):

- add to X one rectangle of $L \cap H_i$ (the highest..)
- if there are enough rectangles of $H_i \setminus L$ to reach $2v$, add them to X until reaching $2v$ (we will have $\sum_{j \in X} p_j \leq \frac{5v}{2}$)
- otherwise (we know that few jobs of $H_i \setminus L$ remain), add to X jobs of $I' \setminus H_i$ using a non-increasing work order, stops when $W(X) \geq m_i v$ or when no jobs of $I' \setminus H_i$ remain.

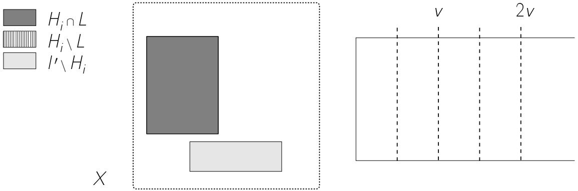


Toward a "good" A_{select}

Our A_{select}

Given Cl_i the current empty cluster (initially $X = \emptyset$):

- add to X one rectangle of $L \cap H_i$ (the highest..)
- if there are enough rectangles of $H_i \setminus L$ to reach $2v$, add them to X until reaching $2v$ (we will have $\sum_{j \in X} p_j \leq \frac{5v}{2}$)
- otherwise (we know that few jobs of $H_i \setminus L$ remain), add to X jobs of $I' \setminus H_i$ using a non-increasing work order, stops when $W(X) \geq m_i v$ or when no jobs of $I' \setminus H_i$ remain.

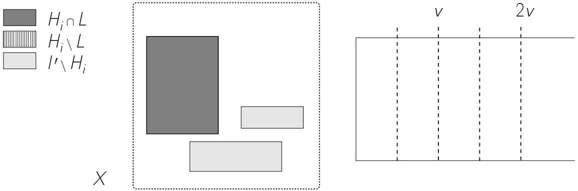


Toward a "good" A_{select}

Our A_{select}

Given Cl_i the current empty cluster (initially $X = \emptyset$):

- add to X one rectangle of $L \cap H_i$ (the highest..)
- if there are enough rectangles of $H_i \setminus L$ to reach $2v$, add them to X until reaching $2v$ (we will have $\sum_{j \in X} p_j \leq \frac{5v}{2}$)
- otherwise (we know that few jobs of $H_i \setminus L$ remain), add to X jobs of $I' \setminus H_i$ using a non-increasing work order, stops when $W(X) \geq m_i v$ or when no jobs of $I' \setminus H_i$ remain.

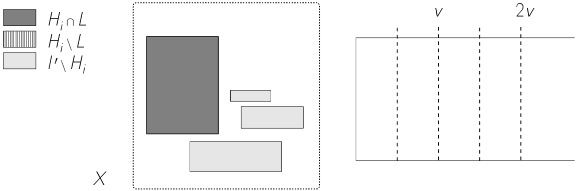


Toward a "good" A_{select}

Our A_{select}

Given Cl_i the current empty cluster (initially $X = \emptyset$):

- add to X one rectangle of $L \cap H_i$ (the highest..)
- if there are enough rectangles of $H_i \setminus L$ to reach $2v$, add them to X until reaching $2v$ (we will have $\sum_{j \in X} p_j \leq \frac{5v}{2}$)
- otherwise (we know that few jobs of $H_i \setminus L$ remain), add to X jobs of $I' \setminus H_i$ using a non-increasing work order, stops when $W(X) \geq m_i v$ or when no jobs of $I' \setminus H_i$ remain.



What is missing now?

Road map

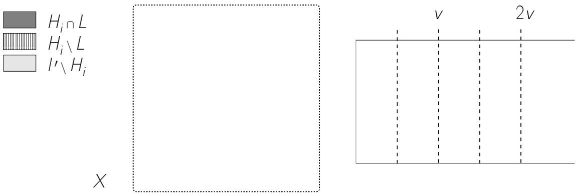
What we have to do is:

- define C_1: DONE
- if C_1 is true (i.e. when $W(X) \geq m_i v$), prove that X can be scheduled in $\frac{5v}{2}$
- otherwise, prove that it is "obvious" to finish packing all the jobs of I' in all the remaining empty clusters

Analyzing A_{select}

Our A_{select}

- if there are enough rectangles of $H_i \setminus L$ to reach $2v$, add them to X until reaching $2v$ (we will have $\sum_{j \in X} p_j \leq \frac{5v}{2}$)
- otherwise (we know that few jobs of $H_i \setminus L$ remain), add to X jobs of $I' \setminus H_i$ using a non-increasing work order, stops when $W(X) \geq m_i v$ or when no jobs of $I' \setminus H_i$ remain.

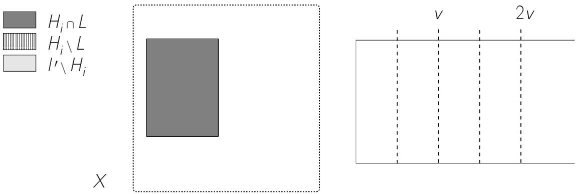


- if $m_i v \leq W(X) \leq \frac{5 m_i v}{4}$: use Steinberg's algorithm
- if $W(X) > \frac{5 m_i v}{4}$?

Analyzing A_{select}

Our A_{select}

- if there are enough rectangles of $H_i \setminus L$ to reach $2v$, add them to X until reaching $2v$ (we will have $\sum_{j\in X} p_j \leq \frac{5v}{2}$)
- otherwise (we know that few jobs of $H_i \setminus L$ remain), add to X jobs of $I' \setminus H_i$ using a non-increasing work order, stops when $W(X) \geq m_i v$ or when no jobs of $I' \setminus H_i$ remain.

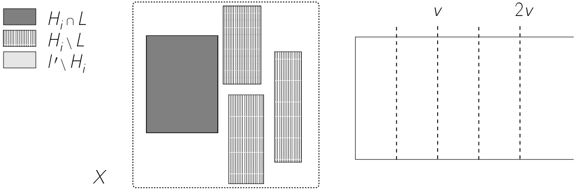


- if $m_i v \leq W(X) \leq \frac{5m_i v}{4}$: use Steinberg's algorithm
- if $W(X) > \frac{5m_i v}{4}$?

Analyzing A_{select}

Our A_{select}

- if there are enough rectangles of $H_i \setminus L$ to reach $2v$, add them to X until reaching $2v$ (we will have $\sum_{j \in X} p_j \leq \frac{5v}{2}$)
- otherwise (we know that few jobs of $H_i \setminus L$ remain), add to X jobs of $I' \setminus H_i$ using a non-increasing work order, stops when $W(X) \geq m_i v$ or when no jobs of $I' \setminus H_i$ remain.

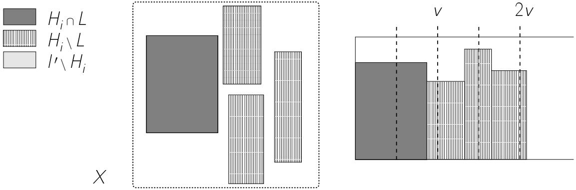


- if $m_i v \leq W(X) \leq \frac{5 m_i v}{4}$: use Steinberg's algorithm
- if $W(X) > \frac{5 m_i v}{4}$?

Analyzing A_{select}

Our A_{select}

- if there are enough rectangles of $H_i \setminus L$ to reach $2v$, add them to X until reaching $2v$ (we will have $\sum_{j \in X} p_j \leq \frac{5v}{2}$)
- otherwise (we know that few jobs of $H_i \setminus L$ remain), add to X jobs of $I' \setminus H_i$ using a non-increasing work order, stops when $W(X) \geq m_i v$ or when no jobs of $I' \setminus H_i$ remain.

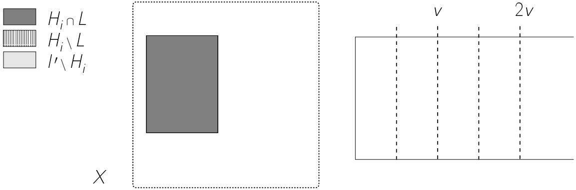


- if $m_i v \leq W(X) \leq \frac{5 m_i v}{4}$: use Steinberg's algorithm
- if $W(X) > \frac{5 m_i v}{4}$?

Analyzing A_{select}

Our A_{select}

- if there are enough rectangles of $H_i \setminus L$ to reach $2v$, add them to X until reaching $2v$ (we will have $\sum_{j \in X} p_j \leq \frac{5v}{2}$)
- otherwise (we know that few jobs of $H_i \setminus L$ remain), add to X jobs of $I' \setminus H_i$ using a non-increasing work order, stops when $W(X) \geq m_i v$ or when no jobs of $I' \setminus H_i$ remain.

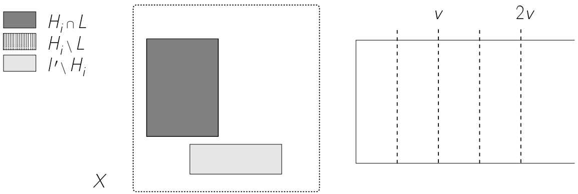


- if $m_i v \leq W(X) \leq \frac{5 m_i v}{4}$: use Steinberg's algorithm
- if $W(X) > \frac{5 m_i v}{4}$?

Analyzing A_{select}

Our A_{select}

- if there are enough rectangles of $H_i \setminus L$ to reach $2v$, add them to X until reaching $2v$ (we will have $\sum_{j \in X} p_j \leq \frac{5v}{2}$)
- otherwise (we know that few jobs of $H_i \setminus L$ remain), add to X jobs of $I' \setminus H_i$ using a non-increasing work order, stops when $W(X) \geq m_i v$ or when no jobs of $I' \setminus H_i$ remain.

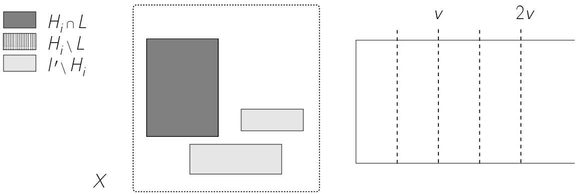


- if $m_i v \leq W(X) \leq \frac{5 m_i v}{4}$: use Steinberg's algorithm
- if $W(X) > \frac{5 m_i v}{4}$?

Analyzing A_{select}

Our A_{select}

- if there are enough rectangles of $H_i \setminus L$ to reach $2v$, add them to X until reaching $2v$ (we will have $\sum_{j \in X} p_j \leq \frac{5v}{2}$)
- otherwise (we know that few jobs of $H_i \setminus L$ remain), add to X jobs of $I' \setminus H_i$ using a non-increasing work order, stops when $W(X) \geq m_i v$ or when no jobs of $I' \setminus H_i$ remain.

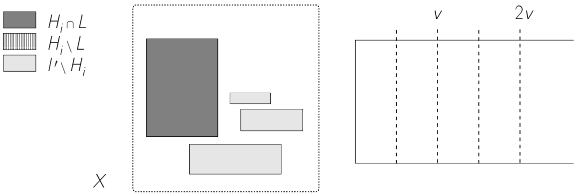


- if $m_i v \leq W(X) \leq \frac{5 m_i v}{4}$: use Steinberg's algorithm
- if $W(X) > \frac{5 m_i v}{4}$?

Analyzing A_{select}

Our A_{select}

- if there are enough rectangles of $H_i \setminus L$ to reach $2v$, add them to X until reaching $2v$ (we will have $\sum_{j \in X} p_j \leq \frac{5v}{2}$)
- otherwise (we know that few jobs of $H_i \setminus L$ remain), add to X jobs of $I' \setminus H_i$ using a non-increasing work order, stops when $W(X) \geq m_i v$ or when no jobs of $I' \setminus H_i$ remain.

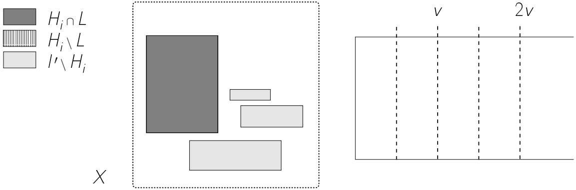


- if $m_i v \leq W(X) \leq \frac{5 m_i v}{4}$: use Steinberg's algorithm
- if $W(X) > \frac{5 m_i v}{4}$?

Analyzing A_{select}

Our A_{select}

- if there are enough rectangles of $H_i \setminus L$ to reach $2v$, add them to X until reaching $2v$ (we will have $\sum_{j \in X} p_j \leq \frac{5v}{2}$)
- otherwise (we know that few jobs of $H_i \setminus L$ remain), add to X jobs of $I' \setminus H_i$ using a non-increasing work order, stops when $W(X) \geq m_i v$ or when no jobs of $I' \setminus H_i$ remain.

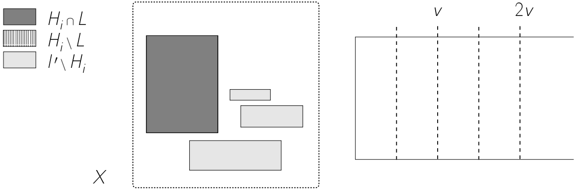


- if $m_i v \leq W(X) \leq \frac{5m_i v}{4}$: use Steinberg's algorithm
- if $W(X) > \frac{5m_i v}{4}$?

Analyzing A_{select}

Our A_{select}

- if there are enough rectangles of $H_i \setminus L$ to reach $2v$, add them to X until reaching $2v$ (we will have $\sum_{j \in X} p_j \leq \frac{5v}{2}$)
- otherwise (we know that few jobs of $H_i \setminus L$ remain), add to X jobs of $I' \setminus H_i$ using a non-increasing work order, stops when $W(X) \geq m_i v$ or when no jobs of $I' \setminus H_i$ remain.



- if $m_i v \leq W(X) \leq \frac{5 m_i v}{4}$: use Steinberg's algorithm
- if $W(X) > \frac{5 m_i v}{4}$?

Scheduling at most 4 big jobs

Let $X = \{J_0\} \cup \{J_1, \ldots, J_p\}$, with

- $J_0 \in H_i \cap L$ (if not empty)
- $W(J_j) \geq W(J_{j+1})$, for $1 \leq j \leq p-1$
- $W(X) > \frac{5 m_i v}{4}$ implies $J_j \in L$ for $1 \leq j \leq p$ and $p \leq 4$.

Scheduling at most 4 big jobs

Let $X = \{J_0\} \cup \{J_1, \ldots, J_p\}$, with

- $J_0 \in H_i \cap L$ (if not empty)
- $W(J_j) \geq W(J_{j+1})$, for $1 \leq j \leq p-1$
- $W(X) > \frac{5m_i v}{4}$ implies $J_j \in L$ for $1 \leq j \leq p$ and $p \leq 4$.

Scheduling at most 4 big jobs

Let $X = \{J_0\} \cup \{J_1, \ldots, J_p\}$, with

- $J_0 \in H_i \cap L$ (if not empty)
- $W(J_j) \geq W(J_{j+1})$, for $1 \leq j \leq p-1$
- $W(X) > \frac{5m_i v}{4}$ implies $J_j \in L$ for $1 \leq j \leq p$ and $p \leq 4$.

Scheduling at most 4 big jobs

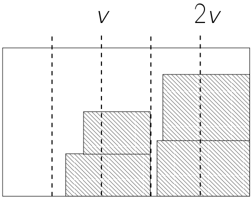
Let $X = \{J_0\} \cup \{J_1, \ldots, J_p\}$, with

- $J_0 \in H_i \cap L$ (if not empty)
- $W(J_j) \geq W(J_{j+1})$, for $1 \leq j \leq p-1$
- $W(X) > \frac{5m_i v}{4}$ implies $J_j \in L$ for $1 \leq j \leq p$ and $p \leq 4$.

Scheduling at most 4 big jobs

Let $X = \{J_0\} \cup \{J_1, \ldots, J_p\}$, with

- $J_0 \in H_i \cap L$ (if not empty)
- $W(J_j) \geq W(J_{j+1})$, for $1 \leq j \leq p-1$
- $W(X) > \frac{5m_i v}{4}$ implies $J_j \in L$ for $1 \leq j \leq p$ and $p \leq 4$.



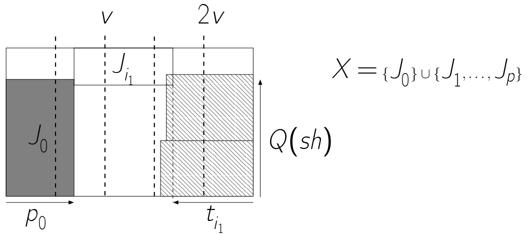
$X = \{J_0\} \cup \{J_1, \ldots, J_p\}$

If J_0 "does not exist", obvious! (as $\{J_1, \ldots, J_p\} \subset (L \setminus H_i)$).

Scheduling at most 4 big jobs

Let $X = \{J_0\} \cup \{J_1, \dots, J_p\}$, with

- $J_0 \in H_i \cap L$ (if not empty)
- $W(J_j) \geq W(J_{j+1})$, for $1 \leq j \leq p-1$
- $W(X) > \frac{5m_i v}{4}$ implies $J_j \in L$ for $1 \leq j \leq p$ and $p \leq 4$.



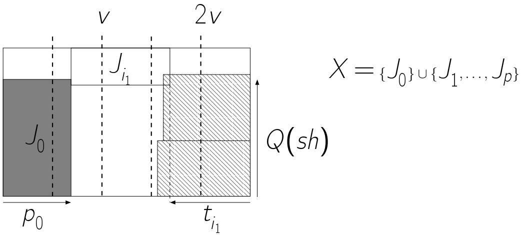
Otherwise, we even have $p \leq 3$, and the only non obvious case if for $p = 3$.

- let J_{i_1} be the rectangle of $\{J_1, J_2, J_3\}$ having the smallest q_j
- let $t_{i_1} = \frac{5v}{2} - p_0 - p_{i_1}$.

Scheduling at most 4 big jobs

Let $X = \{J_0\} \cup \{J_1, \dots, J_p\}$, with

- $J_0 \in H_i \cap L$ (if not empty)
- $W(J_j) \geq W(J_{j+1})$, for $1 \leq j \leq p-1$
- $W(X) > \frac{5m_i v}{4}$ implies $J_j \in L$ for $1 \leq j \leq p$ and $p \leq 4$.



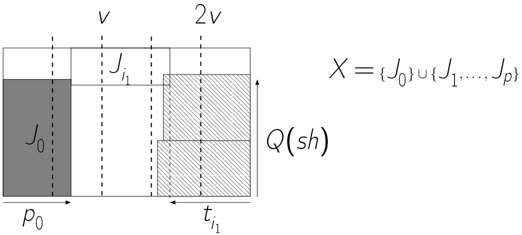
Otherwise, we even have $p \leq 3$, and the only non obvious case if for $p = 3$.

- let J_{i_1} be the rectangle of $\{J_1, J_2, J_3\}$ having the smallest q_j
- let $t_{i_1} = \frac{5v}{2} - p_0 - p_{i_1}$.

Scheduling at most 4 big jobs

Let $X = \{J_0\} \cup \{J_1, \ldots, J_p\}$, with

- $J_0 \in H_i \cap L$ (if not empty)
- $W(J_j) \geq W(J_{j+1})$, for $1 \leq j \leq p-1$
- $W(X) > \frac{5m_i v}{4}$ implies $J_j \in L$ for $1 \leq j \leq p$ and $p \leq 4$.



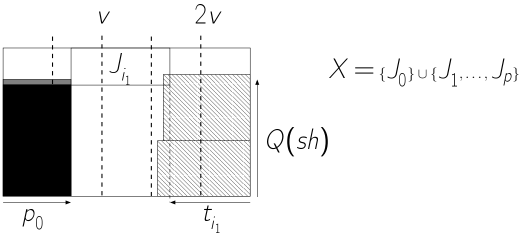
Otherwise, we even have $p \leq 3$, and the only non obvious case if for $p = 3$.

- let J_{i_1} be the rectangle of $\{J_1, J_2, J_3\}$ having the smallest q_j
- let $t_{i_1} = \frac{5v}{2} - p_0 - p_{i_1}$.

Scheduling at most 4 big jobs

Let $X = \{J_0\} \cup \{J_1, \ldots, J_p\}$, with

- $J_0 \in H_i \cap L$ (if not empty)
- $W(J_j) \geq W(J_{j+1})$, for $1 \leq j \leq p-1$
- $W(X) > \frac{5m_i v}{4}$ implies $J_j \in L$ for $1 \leq j \leq p$ and $p \leq 4$.



If J_{i_1} intersects the shelf we get:

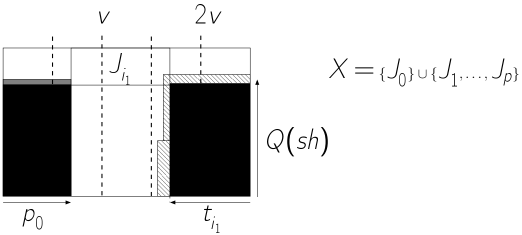
$$W(X \setminus J_1) > p_0(m_i - q_1) + t_{i_1}(m_i - q_1) + (Q(Sh) - (m_i - q_1))\frac{v}{2}$$

$$> m_i v \qquad \text{as } Q(sh) > 2q_{i_1} \text{ and } t_{i_1} > \frac{3v}{2} - p_0$$

Scheduling at most 4 big jobs

Let $X = \{J_0\} \cup \{J_1, \ldots, J_p\}$, with

- $J_0 \in H_i \cap L$ (if not empty)
- $W(J_j) \geq W(J_{j+1})$, for $1 \leq j \leq p-1$
- $W(X) > \frac{5m_i v}{4}$ implies $J_j \in L$ for $1 \leq j \leq p$ and $p \leq 4$.



If J_{i_1} intersects the shelf we get:

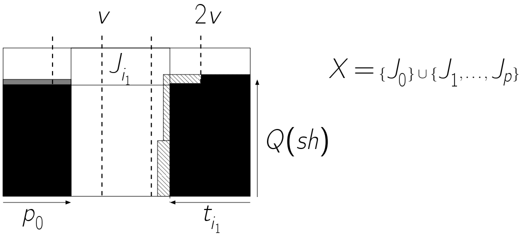
$$W(X \setminus J_1) > p_0(m_i - q_1) + t_{i_1}(m_i - q_1) + (Q(Sh) - (m_i - q_1))\frac{v}{2}$$

$$> m_i v \qquad \text{as } Q(sh) > 2q_{i_1} \text{ and } t_{i_1} > \frac{3v}{2} - p_0$$

Scheduling at most 4 big jobs

Let $X = \{J_0\} \cup \{J_1, \ldots, J_p\}$, with

- $J_0 \in H_i \cap L$ (if not empty)
- $W(J_j) \geq W(J_{j+1})$, for $1 \leq j \leq p-1$
- $W(X) > \frac{5m_i v}{4}$ implies $J_j \in L$ for $1 \leq j \leq p$ and $p \leq 4$.

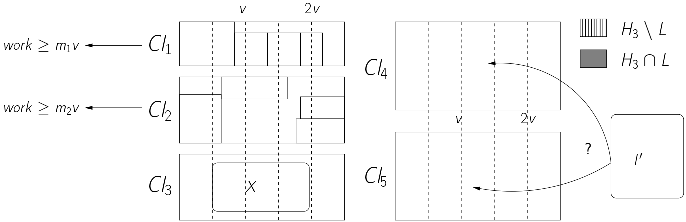


If J_{i_1} intersects the shelf we get:

$$W(X \setminus J_1) > p_0(m_i - q_1) + t_{i_1}(m_i - q_1) + (Q(Sh) - (m_i - q_1))\frac{v}{2}$$

$$> m_i v \quad \text{as } Q(sh) > 2q_{i_1} \text{ and } t_{i_1} > \frac{3v}{2} - p_0$$

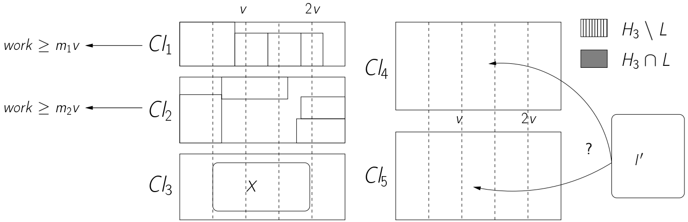
Finishing when C_1 is false



Let us suppose that A_{select} fails for Cl_3 (i.e. returns X with $W(X) < m_3 v$). Let I' be the remaining jobs after filling Cl_3.

- $I' \subset H_3$
- $P(I' \setminus L) = \sum_{I'} p_x \leq 2v$
- if $X \cap (H_3 \cap L) = \emptyset$ then $I' \subset H_3 \setminus L$.
- if $X \cap (H_3 \cap L) \neq \emptyset$: $P(I' \setminus L) \leq \frac{3v}{2}$
  - how to schedule $I' \cap L$?
  - how to schedule $I' \setminus L$?

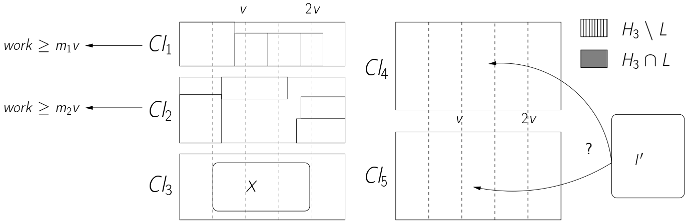
Finishing when C_1 is false



Let us suppose that A_{select} fails for Cl_3 (i.e. returns X with $W(X) < m_3 v$). Let I' be the remaining jobs after filling Cl_3.

- $I' \subset H_3$
- $P(I' \setminus L) = \sum_{I'} p_x \leq 2v$
- if $X \cap (H_3 \cap L) = \emptyset$ then $I' \subset H_3 \setminus L$.
- if $X \cap (H_3 \cap L) \neq \emptyset$: $P(I' \setminus L) \leq \frac{3v}{2}$
    - how to schedule $I' \cap L$?
    - how to schedule $I' \setminus L$?

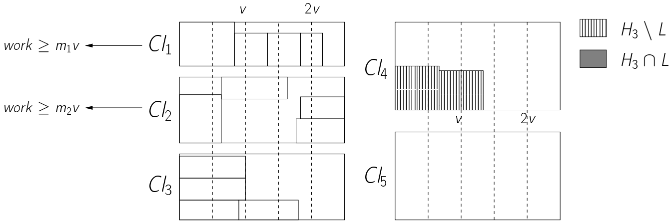
Finishing when C_1 is false



Let us suppose that A_{select} fails for Cl_3 (i.e. returns X with $W(X) < m_3 v$). Let I' be the remaining jobs after filling Cl_3.

- $I' \subset H_3$
- $P(I' \setminus L) = \sum_{I'} p_x \leq 2v$
- if $X \cap (H_3 \cap L) = \emptyset$ then $I' \subset H_3 \setminus L$.
- if $X \cap (H_3 \cap L) \neq \emptyset$: $P(I' \setminus L) \leq \frac{3v}{2}$
  - how to schedule $I' \cap L$?
  - how to schedule $I' \setminus L$?

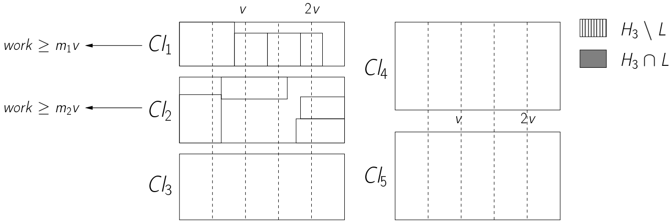
Finishing when C_1 is false



Let us suppose that A_{select} fails for Cl_3 (i.e. returns X with $W(X) < m_3 v$). Let I' be the remaining jobs after filling Cl_3.

- $I' \subset H_3$
- $P(I' \setminus L) = \sum_{I'} p_x \leq 2v$
- if $X \cap (H_3 \cap L) = \emptyset$ then $I' \subset H_3 \setminus L$.
- if $X \cap (H_3 \cap L) \neq \emptyset$: $P(I' \setminus L) \leq \frac{3v}{2}$
  - how to schedule $I' \cap L$?
  - how to schedule $I' \setminus L$?

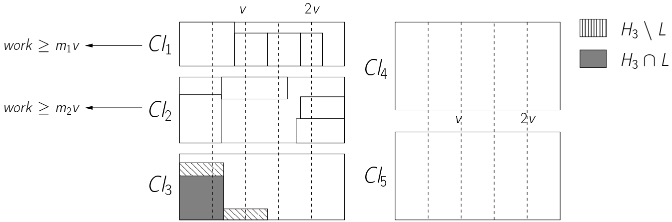
Finishing when C_1 is false



Let us suppose that A_{select} fails for Cl_3 (i.e. returns X with $W(X) < m_3 v$). Let I' be the remaining jobs after filling Cl_3.

- $I' \subset H_3$
- $P(I' \setminus L) = \sum_{I'} p_x \leq 2v$
- if $X \cap (H_3 \cap L) = \emptyset$ then $I' \subset H_3 \setminus L$.
- if $X \cap (H_3 \cap L) \neq \emptyset$: $P(I' \setminus L) \leq \frac{3v}{2}$
  - how to schedule $I' \cap L$?
  - how to schedule $I' \setminus L$?

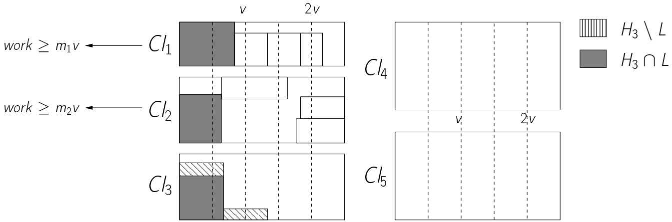
Finishing when C_1 is false



Let us suppose that A_{select} fails for Cl_3 (i.e. returns X with $W(X) < m_3 v$). Let I' be the remaining jobs after filling Cl_3.

- $I' \subset H_3$
- $P(I' \setminus L) = \sum_{I'} p_x \leq 2v$
- if $X \cap (H_3 \cap L) = \emptyset$ then $I' \subset H_3 \setminus L$.
- if $X \cap (H_3 \cap L) \neq \emptyset$: $P(I' \setminus L) \leq \frac{3v}{2}$
    - how to schedule $I' \cap L$?
    - how to schedule $I' \setminus L$?

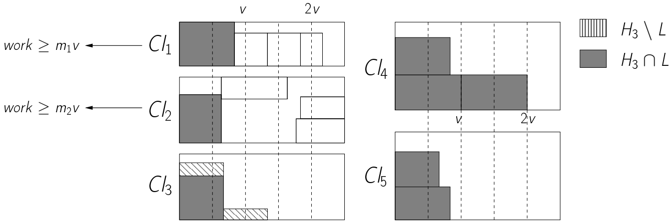
Finishing when C_1 is false



Let us suppose that A_{select} fails for Cl_3 (i.e. returns X with $W(X) < m_3 v$). Let I' be the remaining jobs after filling Cl_3.

- $I' \subset H_3$
- $P(I' \setminus L) = \sum_{I'} p_x \leq 2v$
- if $X \cap (H_3 \cap L) = \emptyset$ then $I' \subset H_3 \setminus L$.
- if $X \cap (H_3 \cap L) \neq \emptyset$: $P(I' \setminus L) \leq \frac{3v}{2}$
  - how to schedule $I' \cap L$?
  - how to schedule $I' \setminus L$?

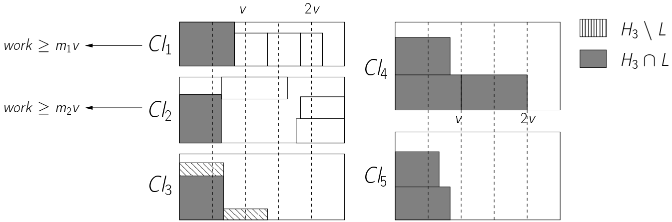
Finishing when C_1 is false



Let us suppose that A_{select} fails for Cl_3 (i.e. returns X with $W(X) < m_3 v$). Let I' be the remaining jobs after filling Cl_3.

- $I' \subset H_3$
- $P(I' \setminus L) = \sum_{I'} p_x \leq 2v$
- if $X \cap (H_3 \cap L) = \emptyset$ then $I' \subset H_3 \setminus L$.
- if $X \cap (H_3 \cap L) \neq \emptyset$: $P(I' \setminus L) \leq \frac{3v}{2}$
  - how to schedule $I' \cap L$?
  - how to schedule $I' \setminus L$?

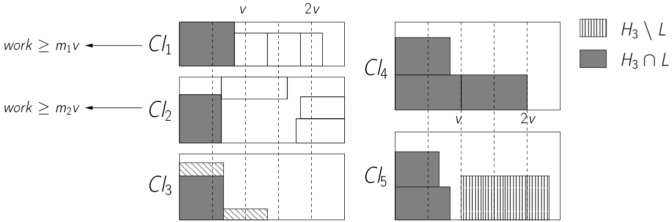
Finishing when C_1 is false



Let us suppose that A_{select} fails for Cl_3 (i.e. returns X with $W(X) < m_3 v$). Let I' be the remaining jobs after filling Cl_3.

- $I' \subset H_3$
- $P(I' \setminus L) = \sum_{I'} p_x \leq 2v$
- if $X \cap (H_3 \cap L) = \emptyset$ then $I' \subset H_3 \setminus L$.
- if $X \cap (H_3 \cap L) \neq \emptyset$: $P(I' \setminus L) \leq \frac{3v}{2}$
  - how to schedule $I' \cap L$?
  - how to schedule $I' \setminus L$?

Finishing when C_1 is false



Let us suppose that A_{select} fails for Cl_3 (i.e. returns X with $W(X) < m_3 v$). Let I' be the remaining jobs after filling Cl_3.

- $I' \subset H_3$
- $P(I' \setminus L) = \sum_{I'} p_x \leq 2v$
- if $X \cap (H_3 \cap L) = \emptyset$ then $I' \subset H_3 \setminus L$.
- if $X \cap (H_3 \cap L) \neq \emptyset$: $P(I' \setminus L) \leq \frac{3v}{2}$
  - how to schedule $I' \cap L$?
  - how to schedule $I' \setminus L$?

Conclusion

This algorithm

- is a $\frac{5}{2}$-approximation (improving the previous 3 bound, the lower bound being 2)
- runs in $\mathcal{O}(log(np_{mapx})kC_{Steinb})$ with $C_{Steinb} = nlog^2(n)/log(log(n))$
- also applies for continuous scheduling (i.e. rectangle packing)
- :(requires that every job fits everywhere

Remarks / future work

- why not $\frac{7}{3}$?
  - $A(X) \leq \frac{7m_iv}{6} \Rightarrow$Steinberg
  - $A(X) > \frac{7m_iv}{6} \Rightarrow$ at most 6 rectangles..
- remove the "fit everywhere" assumption..

Bibliography

[HS87] D.S. Hochbaum and D.B. Shmoys.
Using dual approximation algorithms for scheduling problems theoretical and practical results.
Journal of the ACM (JACM), 34(1):144–162, 1987.

[Ste97] A. Steinberg.
A strip-packing algorithm with absolute performance bound 2.
SIAM Journal on Computing, 26:401, 1997.

[STY08] U. Schwiegelshohn, A. Tchernykh, and R. Yahyapour.
Online scheduling in grids.
In *IEEE International Symposium on Parallel and Distributed Processing (IPDPS)*, pages 1–10, 2008.

[YHZ09] D. Ye, X. Han, and G. Zhang.
On-Line Multiple-Strip Packing.
In *Proceedings of the 3rd International Conference on Combinatorial Optimization and Applications (COCOA)*, page 165. Springer, 2009.

[Zhu06] SN Zhuk.
Approximate algorithms to pack rectangles into several strips.
Discrete Mathematics and Applications, 16(1):73–85, 2006.