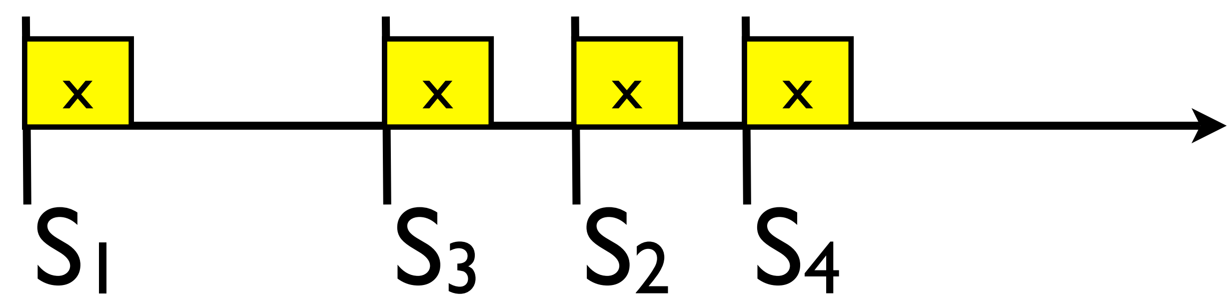
Triangle Scheduling

Christoph Dürr, Zdeněk Hanzálek, Christian Konrad, Yasmina Seddik, René Sitters, Óscar Carlos Vásquez, Gerhard Woeginger,

March 2nd 2016, seminar S

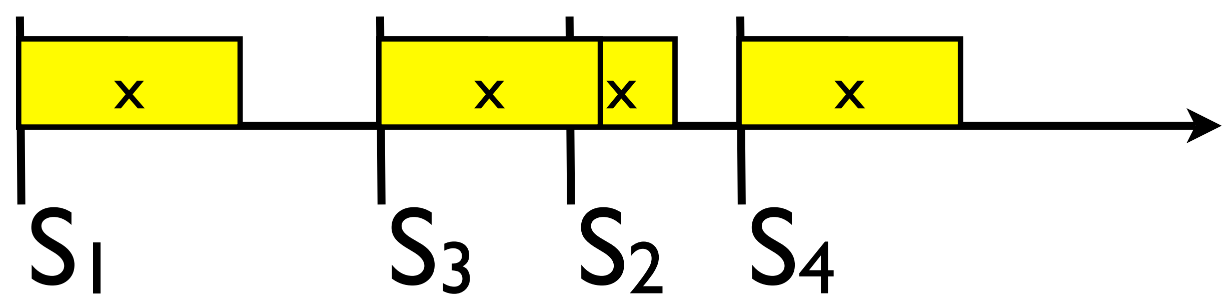
A scheduling problem

- Single machine
- n jobs, with priorities p_j
- equal processing time x
- decide starting times for jobs prior to knowledge of x
- job j is removed from schedule if $x>p_j$
- minimize makespan



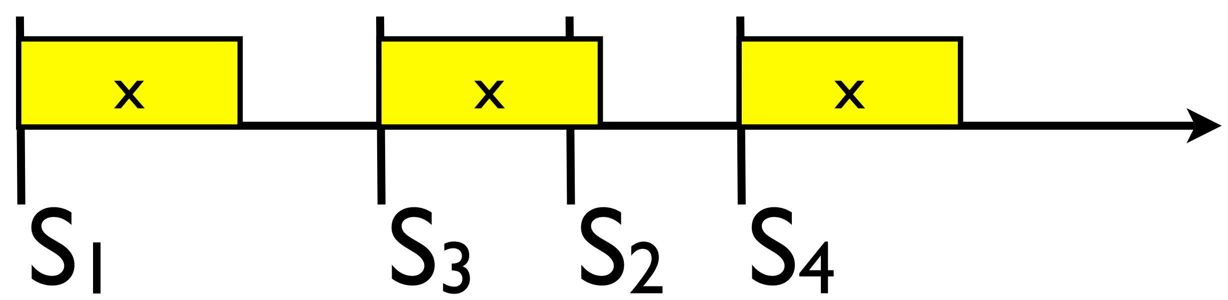
A scheduling problem

- Single machine
- n jobs, with priorities p_j
- equal processing time x
- decide starting times for jobs prior to knowledge of x
- job j is removed from schedule if $x>p_j$
- minimize makespan



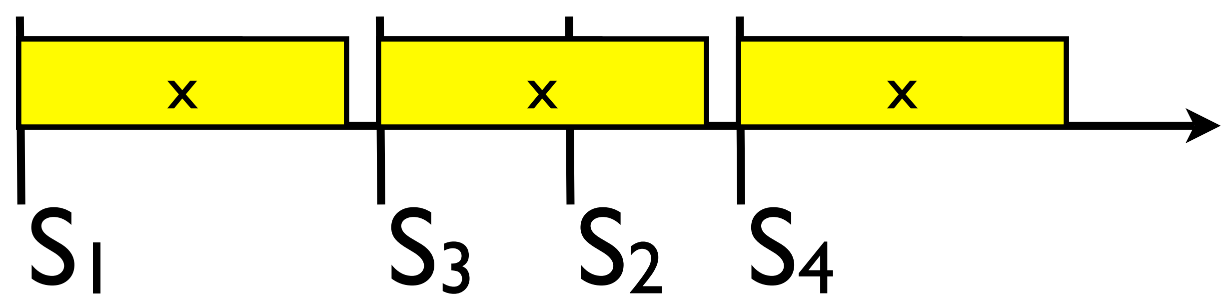
A scheduling problem

- Single machine
- n jobs, with priorities p_j
- equal processing time x
- decide starting times for jobs prior to knowledge of x
- job j is removed from schedule if $x>p_j$
- minimize makespan



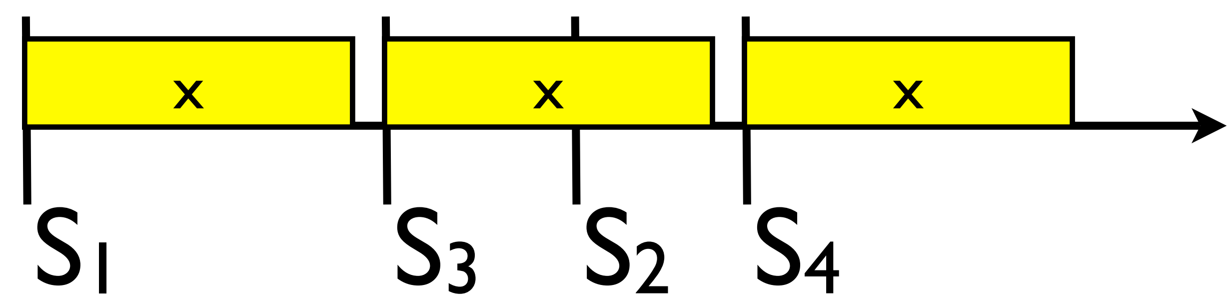
A scheduling problem

- Single machine
- n jobs, with priorities p_j
- equal processing time x
- decide starting times for jobs prior to knowledge of x
- job j is removed from schedule if $x>p_j$
- minimize makespan



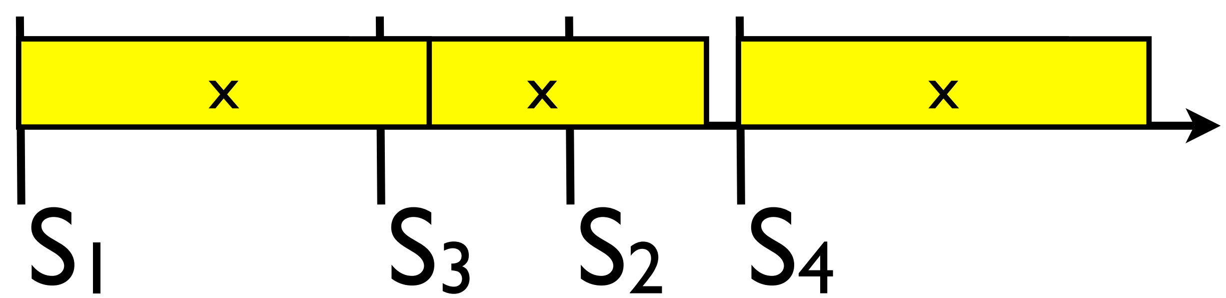
A scheduling problem

- Single machine
- n jobs, with priorities p_j
- equal processing time x
- decide starting times for jobs prior to knowledge of x
- job j is removed from schedule if $x>p_j$
- minimize makespan



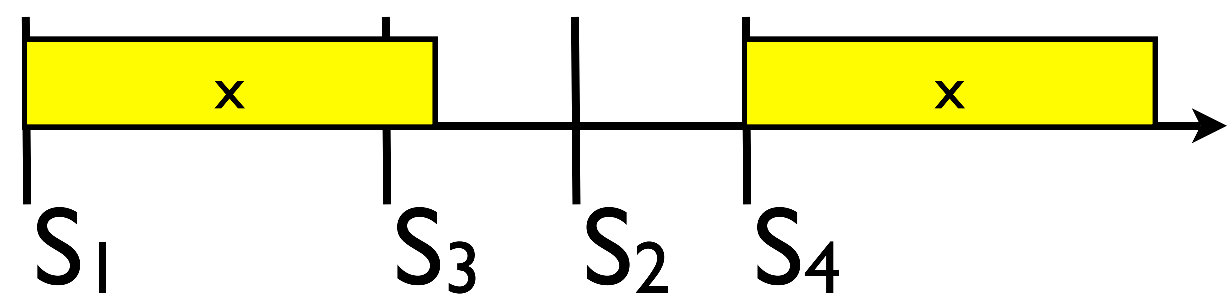
A scheduling problem

- Single machine
- n jobs, with priorities p_j
- equal processing time x
- decide starting times for jobs prior to knowledge of x
- job j is removed from schedule if $x>p_j$
- minimize makespan



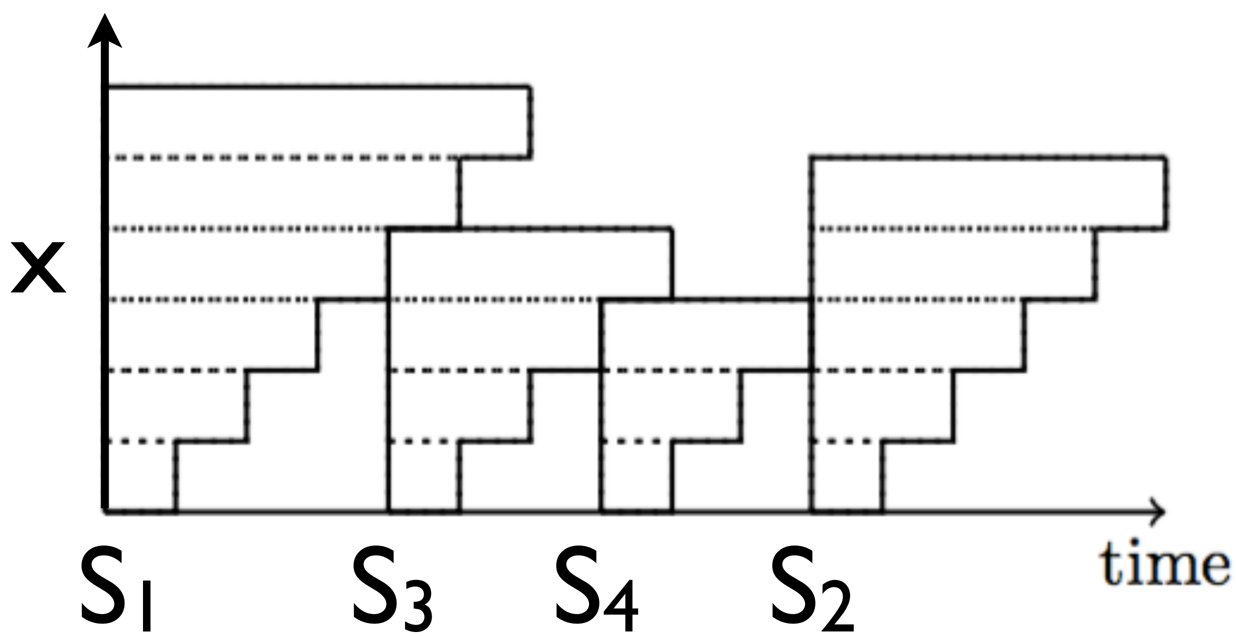
A scheduling problem

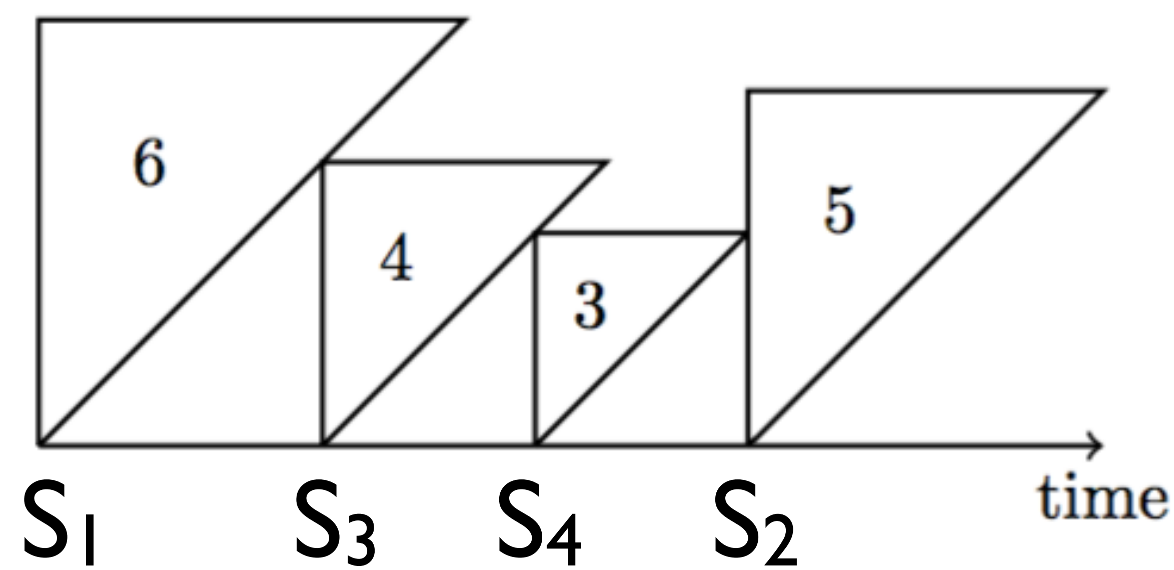
- Single machine
- n jobs, with priorities p_j
- equal processing time x
- decide starting times for jobs prior to knowledge of x
- job j is removed from schedule if $x > p_j$
- minimize makespan



in fact it is

A geometric problem

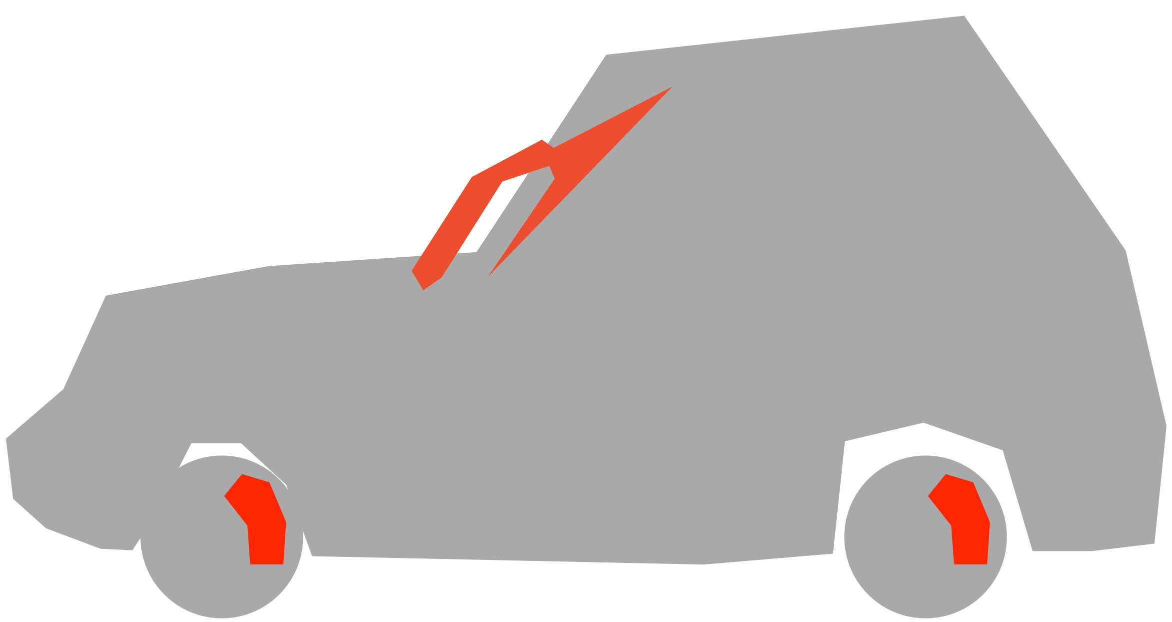




- Single machine
- n triangles, with sizes p_j
- decide starting times $S_j \geq 0$ for job
- such that $|S_i - S_j| \geq \min\{p_i, p_j\}$
- minimize makespan
  max $S_j + p_j$
- place triangles on the time line without overlapping

motivation

Mixed criticality scheduling



- Alan Burns and Robert I. Davis, **Mixed Criticality Systems - A Review,** 7th edition, 2016
- S. K. Baruah, V. Bonifaci, G. D'Angelo, H. Li, A. Marchetti-Spaccamela, N. Megow, L. Stougie, **Mixed-criticality scheduling,** IEEE Transactions on Computers, Vol. 61, pp. 1140-1152, 2012

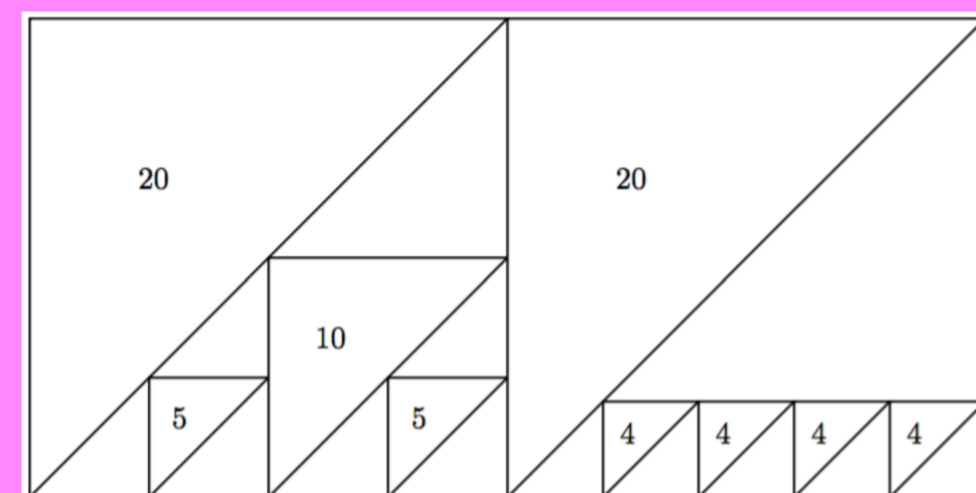
Results

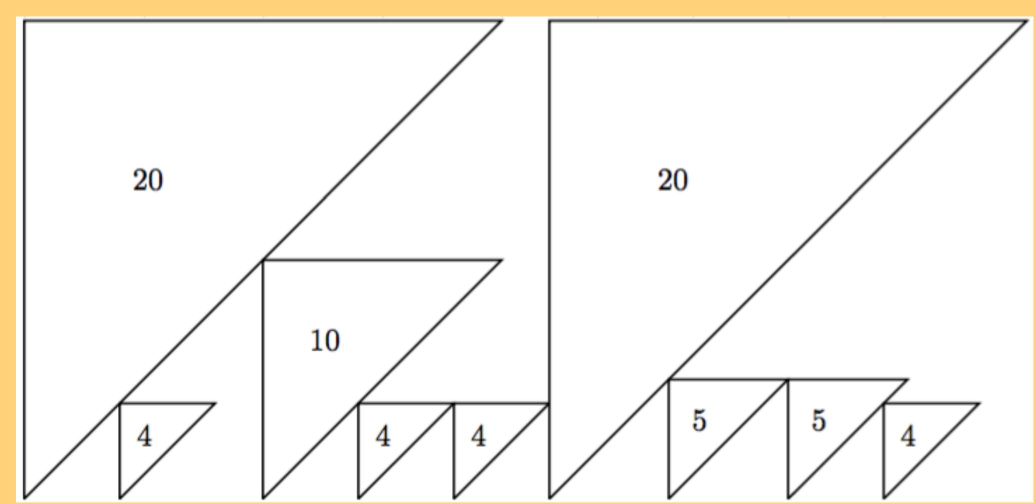
complexity ?	unary NP-hard
where is the barrier ?	binary tree ratio polynomial if ≤2 NP-hard if >2
approximation algorithm ?	Greedy is a 1.5 approximation
ratio tight ?	Greedy's ratio≥1.05
APX-hard ?	No, there is a QPTAS

Greedy

- Process jobs in order $p_1 \geq \ldots \geq p_n$
- Place job j in gap of maximum size s, right shift jobs following gap by $2p_j$-s if $2p_j$>s

opt

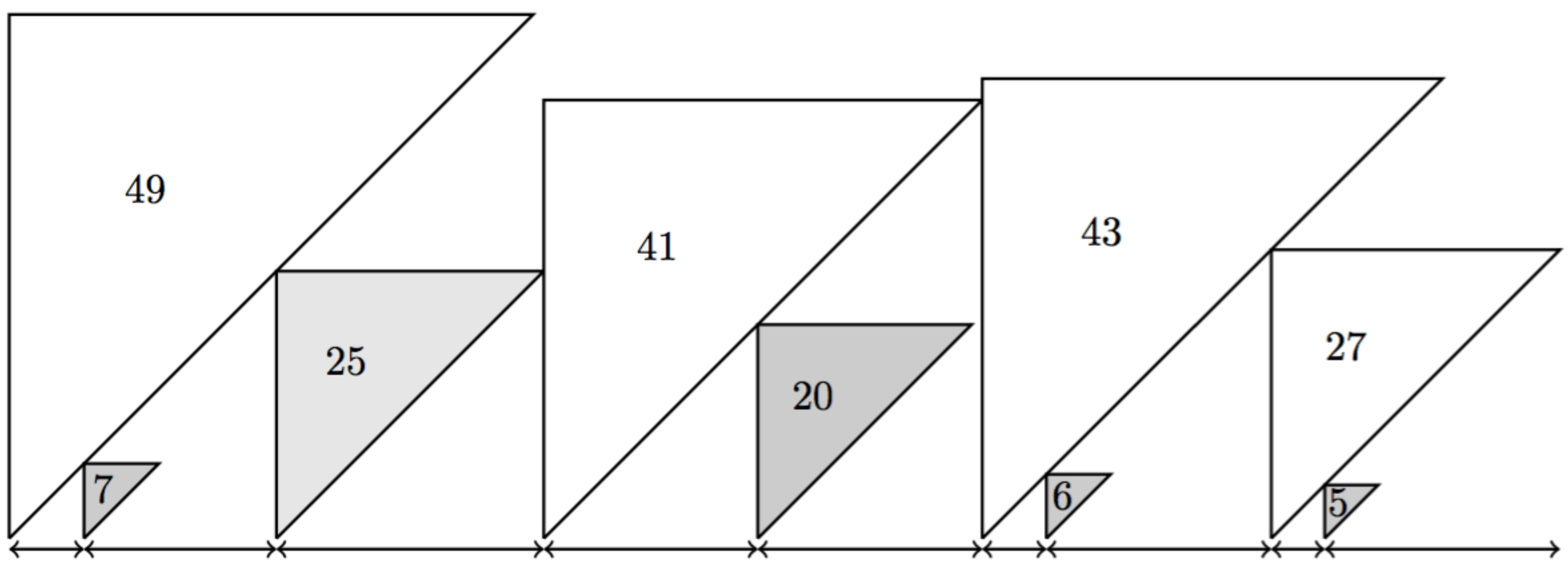




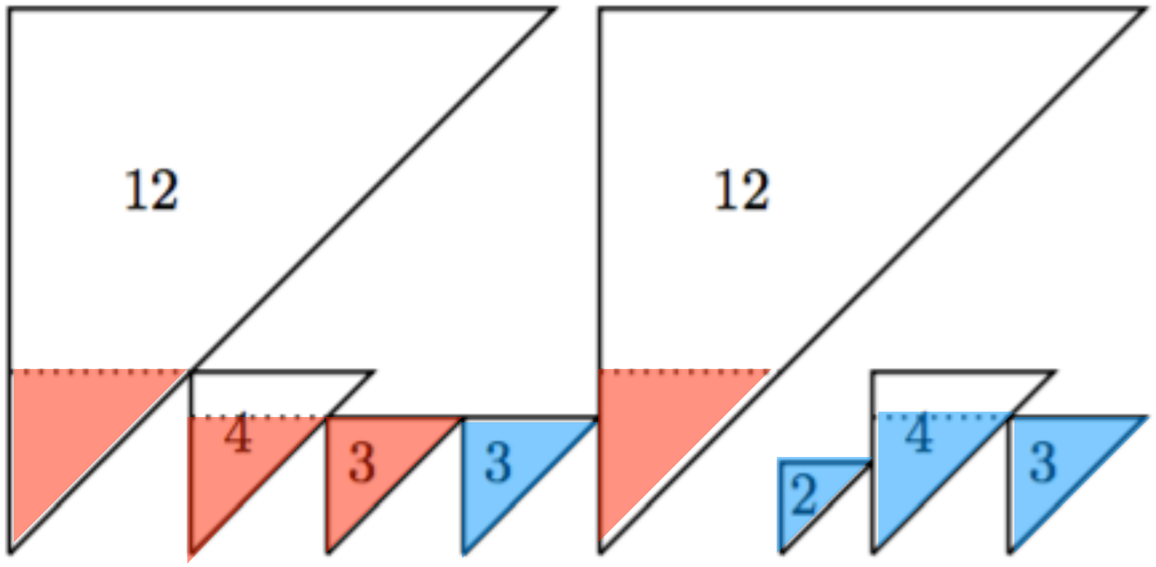
greedy

A lower bound for OPT

- assign every gap $[S_i,S_j]$ to smallest among jobs i,j
- For every job j let $a_j \in \{0,1,2\}$ be the number of assigned gaps
- Property: $\Sigma\, a_j = n$ (number of jobs)
- Property: gap size $\leq$ assigned job size
- Lower bound : $\Sigma\, a_j p_j \leq$ OPT for any $a \in \{0,1,2\}^n$ with $\Sigma\, a_j = n$
- Hence (n even) : 2 times the smallest half of $\{p_1,\ldots,p_n\} \leq$ OPT



Greedy is a 1.5 approximation



- Wlog suppose no job can be shifted to the right
- Truncate job sizes from p to p',
  p_j=size of gap starting at S_j
- Let A be total p'-sizes of larger half of jobs and B of smaller half
- Makespan produced by Greedy is A + B
- Wlog suppose insertion of job n increased makespan
- Hence all gaps have sizes less than $2p_n$, hence A < 2B
- But OPT ≥ 2B
- makespan = A + B ≤ 3B ≤ ⅔OPT(p')≤ ⅔OPT(p)

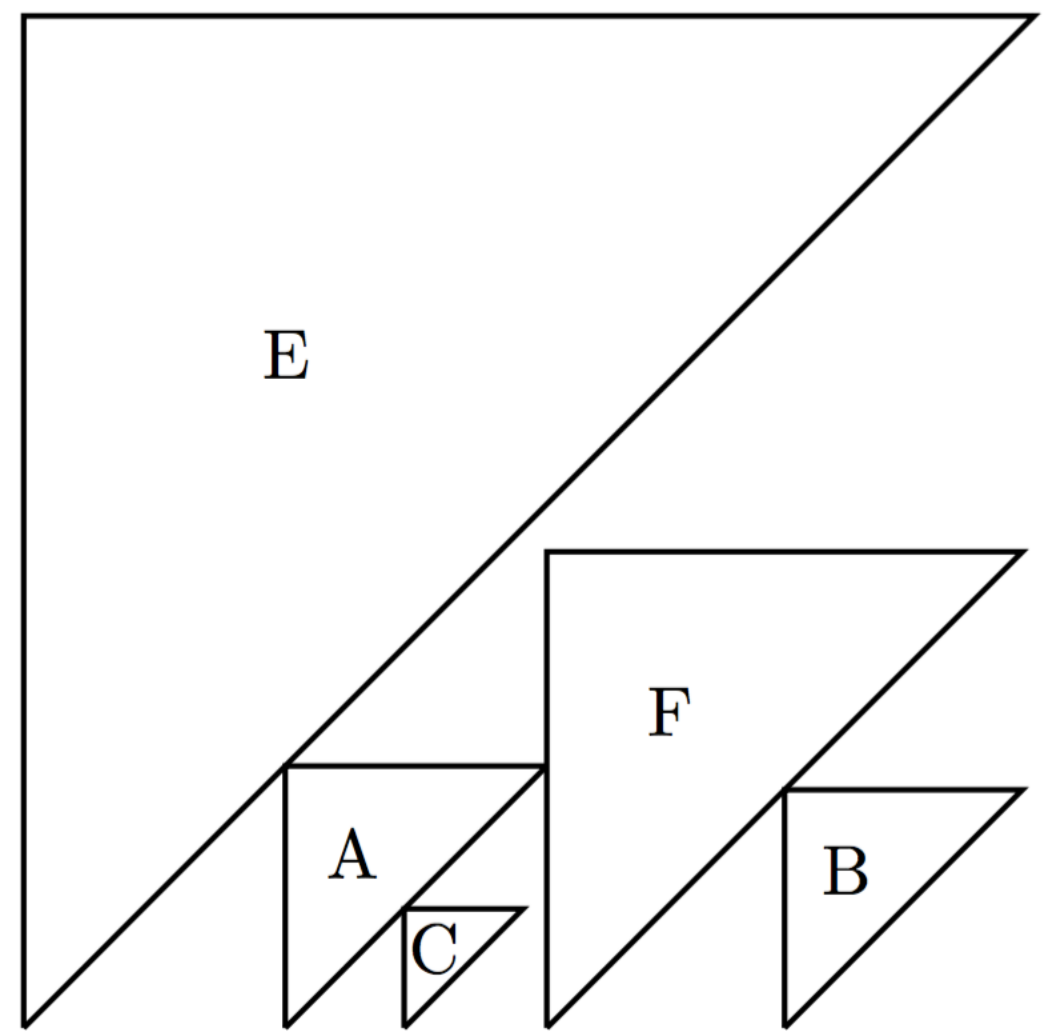
NP-hardness

reduction from 3-dimensional numerical matching

- given $a_1, \ldots, a_n, b_1, \ldots, b_n, c_1, \ldots, c_n, D$
  partition into triplets (a_i,b_j,c_k)
  with $a_i+b_j+c_k=D$

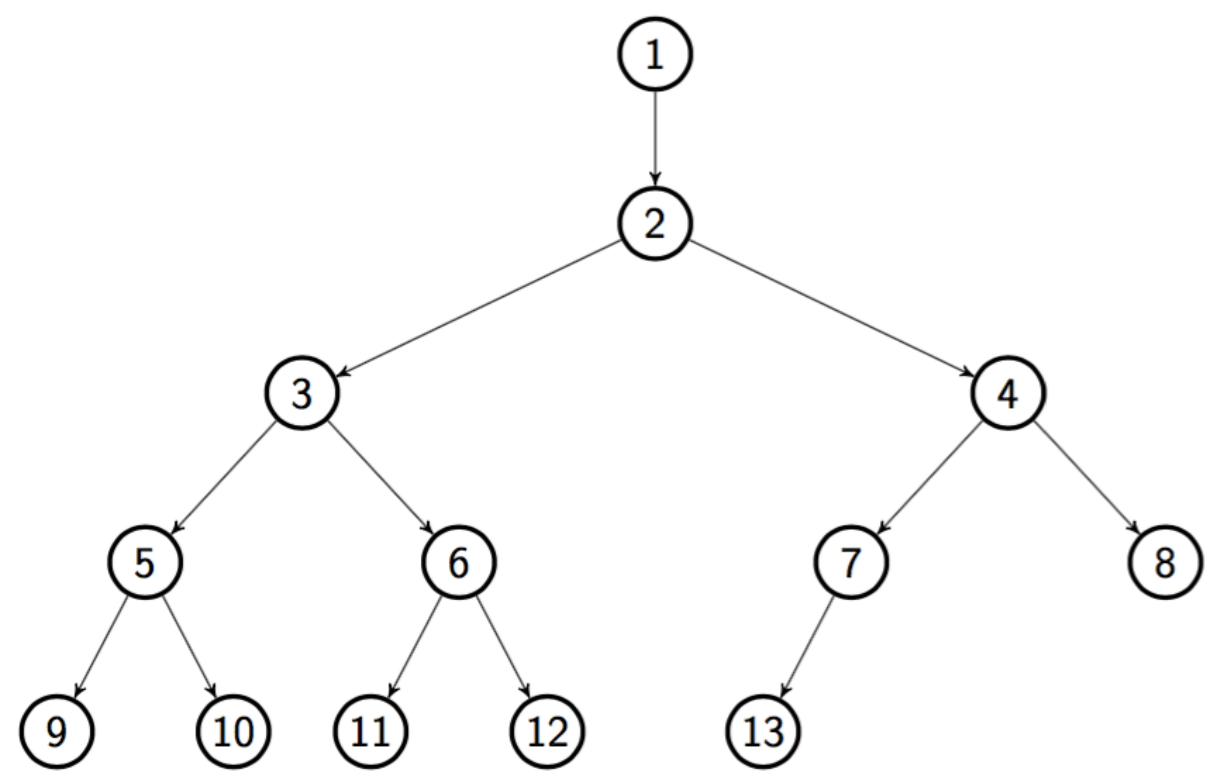
- generate 5n triangles
  M is some arbitrary constant

  - E (size $8M+5D$)
  - F (size $4M$)
  - A_i (size $2M+2a_i+D$)
  - B_j (size $2M+b_j$)
  - C_k (size $M+c_k+D$)



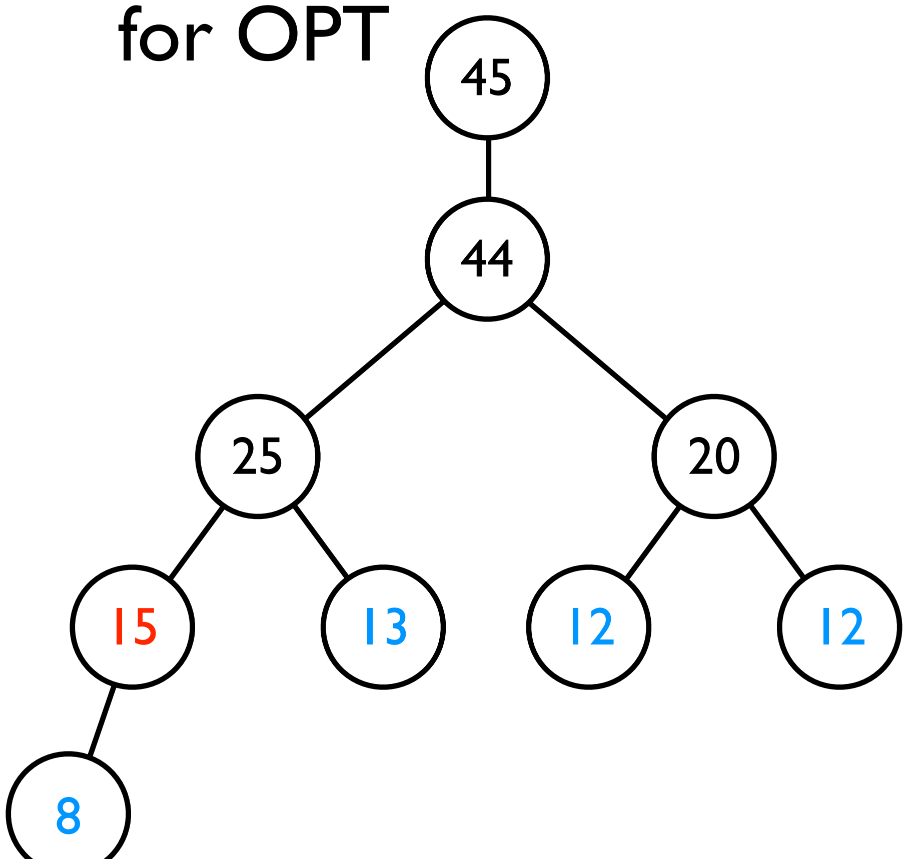
Binary tree ratio

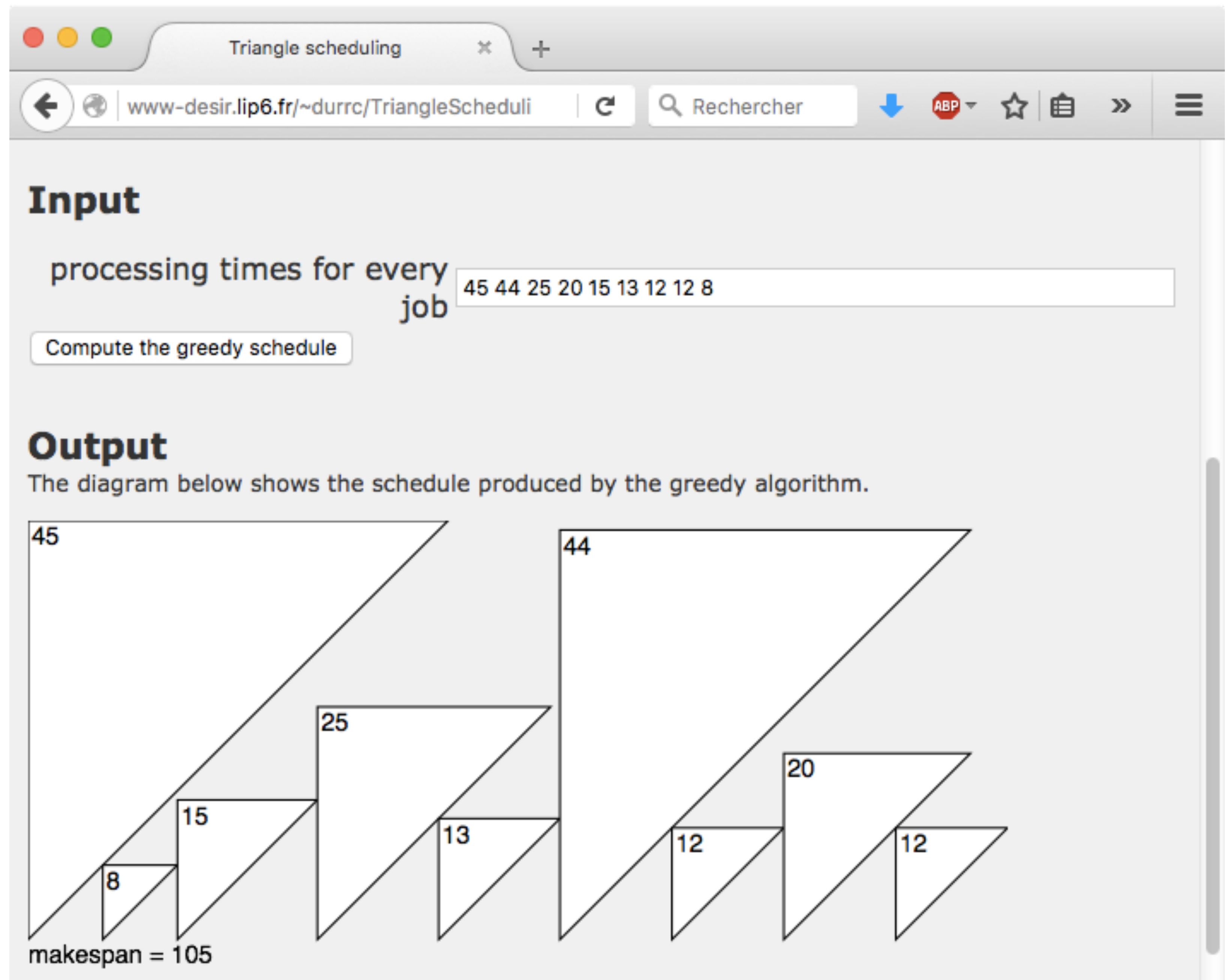
- Suppose order $p_1 \geq \ldots \geq p_n$
- Formally ratio is max $(p_{ceil(i/2)} / p_i)$
- NP-hardness proof generates instances with binary tree ratio > 2 (arbitrarily close)
- Greedy is optimal on instances with binary tree ratio $\leq$ 2.
- Informally it is the maximum ratio between vertex and successor if jobs are placed in row order on this tree



Greedy is optimal when binary tree ratio ≤ 2.

- We construct weights $a_j \in \{0, 1, 2\}$ such that $\Sigma a_j p_j$ is the makespan and the lower bound for OPT





Thank
you for
your
attention,
danke schön,
Děkuji, Merci,
Gracias, dank je