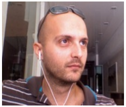
A proof of the Nisan-Ronen Conjecture

STOC 2023



Giorgos Christodoulou

Aristotle University of Thessaloniki, Greece

Elias Koutsoupias

University of Oxford, UK

Annamária Kovács

Goethe University, Frankfurt M., Germany

Unrelated Scheduling

Input:

$$
\begin{array}{cc}
 & m \text{ tasks} \\
n \text{ machines} & \begin{bmatrix} t_{11} & t_{12} & \cdots & t_{1m} \\ t_{21} & t_{22} & \cdots & t_{2m} \\ \vdots & \vdots & & \vdots \\ t_{n1} & t_{n2} & \cdots & t_{nm} \end{bmatrix}
\end{array}
$$

t_{ij} : running time of job j on machine i

Unrelated Scheduling

Input:

$$\begin{array}{cc} & m \text{ tasks} \\ n \text{ machines} & \begin{bmatrix} t_{11} & t_{12} & \cdots & t_{1m} \\ t_{21} & t_{22} & \cdots & t_{2m} \\ \vdots & \vdots & & \vdots \\ t_{n1} & t_{n2} & \cdots & t_{nm} \end{bmatrix} \end{array}$$

t_{ij} : running time of job j on machine i

Output: $x_{ij} \in \{0, 1\}$ an allocation of jobs to machines that minimizes the *makespan*

$$makespan = \max_i \; finish\,time_i$$

Truthful scheduling algorithms

- We are interested only in *weakly monotone (WMON)* scheduling algorithms.
- Exactly these can be complemented by payments to the machines...
- ...so that each machine i reports the running times t_{ij} *truthfully* even if these are private information [Saks, Yu EC05, Bikhchandani et Al. *Econometrica* 2006]
- weakly mon. algorithm + truthful payment = *truthful mechanism*

Definition: The scheduling algorithm is *weakly monotone*, if for every machine i, for every fixed bids of the other machines, for any two bid vectors $(t_{ij})_{j\in[m]}, (t'_{ij})_{j\in[m]}$ and the corresponding allocations $x \neq x'$ holds that $\sum_{j=1}^{m}(x'_{ij} - x_{ij}) \cdot (t'_{ij} - t_{ij}) \leq 0$.

The Vickrey-Clarke-Groves (VCG) mechanism

- the simplest truthful mechanism gives each task independently to the fastest machine for that task

$$\begin{bmatrix} \color{red}{\mathbf{1}^-} & \color{red}{\mathbf{1}^-} & \color{red}{\mathbf{1}^-} & \color{red}{\mathbf{1}^-} & \color{red}{\cdots} & \color{red}{\mathbf{1}^-} \\ 1 & 1 & 1 & 1 & & 1 \\ 1 & 1 & 1 & 1 & & 1 \\ 1 & 1 & 1 & 1 & & 1 \\ \vdots & & & & \ddots & \vdots \\ 1 & 1 & 1 & 1 & \cdots & 1 \end{bmatrix}$$

- VCG is *n-approximative* for makespan minimization

The Nisan-Ronen conjecture

No truthful mechanism for unrelated scheduling can have a better than n approximation of the optimal makespan (indep. of computational power).
[STOC'99, *Games and Economic behavior* 2001]

Lower bounds for truthful makespan approximation:

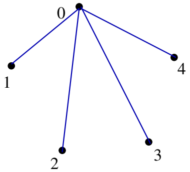
2	[Nisan, Ronen 1999]
$1+\sqrt{2}$	[Christodoulou, Koutsoupias, Vidali *Algorithmica* 2009]
$1+\varphi \approx 2.618$	[Koutsoupias, Vidali *Algorithmica* 2012]
n for *anonymous* mechanisms	[Ashlagi, Dobzinski, Lavi *Math.Op.Res.* 2012]
2.755	[Giannakopoulos, Hammerl, Poças SAGT20]
3	[Dobzinski, Shaulker 2020]
$\sqrt{n-1}+1$	[Christodoulou, Koutsoupias, K. FOCS21]

Our result: No truthful mechanism for unrelated scheduling with n machines has better than n approx. factor for the makespan objective.

Preliminaries I – *graph* and *multigraph* inputs

- we allow only 2 machines for each task:

$$\begin{array}{c} 0. \\ 1. \\ 2. \\ \vdots \\ n. \end{array} \begin{bmatrix} 0 & 0 & \dots & 0 \\ 1 & \infty & \dots & \infty \\ \infty & 1 & \dots & \infty \\ \vdots & & \ddots & \vdots \\ \infty & \infty & \dots & 1 \end{bmatrix} \begin{array}{c} = t \\ s_1 \\ s_2 \\ \vdots \\ s_n \end{array}$$

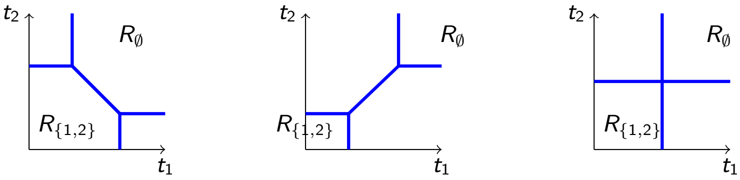


- the tasks can be modelled as edges, and machines as vertices of a graph

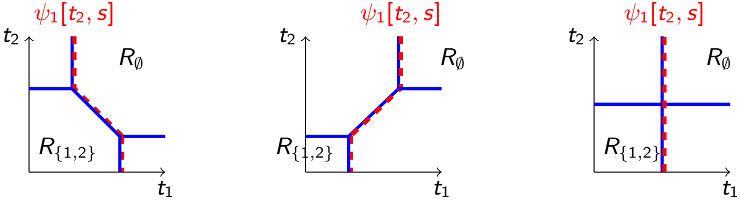
- most of our tasks will have a 0 value on one of their machines (*trivial tasks*)

Preliminaries II – weak monotonicity

- the geometry of WMON allocations
  (here for the t-player and 2 tasks, fixed mechanism, fixed input of other machines)



- the *boundary* ψ_j is the highest t_j value (supremum) that still receives task j



Proof sketch

Recall: ψ_j is the highest t_j value that player 0 still receives task j

$$\begin{array}{c} 0.\\ 1.\\ 2.\\ \vdots\\ \vdots\\ \vdots\\ n.\end{array}
\begin{bmatrix} 0 & 0 & \cdots & \psi_j & \cdots & 0\\ 1 & \infty & \cdots & \infty & \cdots & \infty \\ \infty & 1 & \cdots & \infty & \cdots & \infty\\ \vdots & & \ddots & & & \vdots\\ \vdots & & & 1 & & \vdots \\ \vdots & & & & \ddots & \vdots\\ \infty & \infty & \infty & \infty & \infty & 1\end{bmatrix}
\begin{array}{c} = t\\ s_1\\ s_2\\ \vdots\\ \vdots\\ \vdots\\ s_n\end{array}$$

Idea: Prove the existence of such a (partial) input so that...

A. $\sum_{j=1}^{n} \psi_j \geq n$

Proof sketch

Recall: ψ_j is the highest t_j value that still receives task j

$$
\begin{array}{c}
0. \\ 1. \\ 2. \\ \vdots \\ \vdots \\ \vdots \\ n.
\end{array}
\quad
\left[\begin{array}{cccccc}
\psi_1 & \psi_2 & \dots & \psi_j & \dots & \psi_n \\
1 & \infty & \dots & \infty & \dots & \infty \\
\infty & 1 & \dots & \infty & \dots & \infty \\
\vdots & & \ddots & & & \vdots \\
\vdots & & & 1 & & \vdots \\
\vdots & & & & \ddots & \vdots \\
\infty & \infty & \infty & \infty & \infty & 1
\end{array}\right]
\quad
\begin{array}{c}
= t \\ s_1 \\ s_2 \\ \vdots \\ \vdots \\ \vdots \\ s_n
\end{array}
$$

Idea: Prove the existence of such a (partial) input so that...

A. $\sum_{j=1}^{n} \psi_j \geq n$

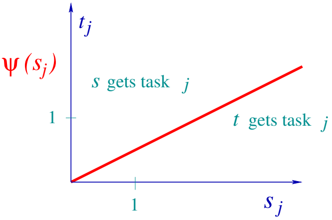
B. and setting ψ_j for *all* j *at once*, player 0 still gets *all* tasks

Then: $ALG = \sum_{j=1}^{n} \psi_j \geq n, \qquad OPT = 1$

Part A: prove existence of tasks with $\sum_j \psi_j \geq n$

$$
\begin{array}{c c}
\begin{bmatrix}
0 & 0 & 0 & \psi_j(s_j) & 0 & 0 & 0 \\
1 & & & & & & \\
 & 1 & & & & & \\
 & & 1 & & & & \\
 & & & s_j & & & \\
 & & & & 1 & & \\
 & & & & & 1 & \\
 & & & & & & 1
\end{bmatrix}
&
\begin{matrix}
= t \\
s_1 \\
\vdots \\
\\
\\
\vdots \\
\\
s_n
\end{matrix}
\end{array}
$$

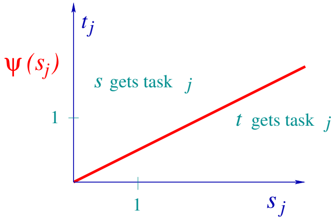
- consider boundary ψ_j as function of s_j
- assume first $\psi_j(s_j) = c \cdot s_j$



Part A: prove existence of tasks with $\sum_j \psi_j \geq n$

$$\begin{bmatrix} 0 & 0 & 0 & \psi_j(s_j) & 0 & 0 & 0 \\ 1 & & & & & & \\ & 1 & & & & & \\ & & 1 & & & & \\ & & & s_j & & & \\ & & & & 1 & & \\ & & & & & 1 & \\ & & & & & & 1 \end{bmatrix} \quad \begin{matrix} = t \\ s_1 \\ \vdots \\ \\ \vdots \\ \\ s_n \end{matrix} \quad \begin{bmatrix} 0 & 0 & 0 & t_j & 0 & 0 & 0 \\ 1 & & & & & & \\ & 1 & & & & & \\ & & 1 & & & & \\ & & & \psi^{-1}(t_j) & & & \\ & & & & 1 & & \\ & & & & & 1 & \\ & & & & & & 1 \end{bmatrix}$$

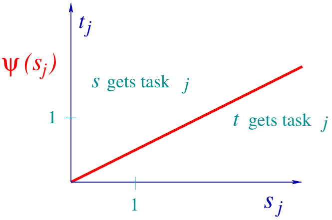
- consider boundary ψ_j as function of s_j
- assume first $\psi_j(s_j) = c \cdot s_j$
- then $\psi_j^{-1}(t_j) = t_j/c$, and ...



Part A: prove existence of tasks with $\sum_j \psi_j \geq n$

$$
\begin{bmatrix}
0 & 0 & 0 & \psi_j(s_j) & 0 & 0 & 0 \\
1 & & & & & & \\
& 1 & & & & & \\
& & 1 & & & & \\
& & & s_j & & & \\
& & & & 1 & & \\
& & & & & 1 & \\
& & & & & & 1
\end{bmatrix}
\quad
\begin{matrix}
= t \\ s_1 \\ \vdots \\ \\ \vdots \\ \\ s_n
\end{matrix}
\quad
\begin{bmatrix}
0 & 0 & 0 & t_j & 0 & 0 & 0 \\
1 & & & & & & \\
& 1 & & & & & \\
& & 1 & & & & \\
& & & \psi^{-1}(t_j) & & & \\
& & & & 1 & & \\
& & & & & 1 & \\
& & & & & & 1
\end{bmatrix}
$$

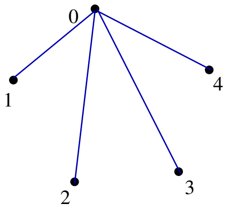
- consider boundary ψ_j as function of s_j
- assume first $\psi_j(s_j) = c \cdot s_j$
- then $\psi_j^{-1}(t_j) = t_j/c$, and ...
- $$\psi_j(1) + \psi_j^{-1}(1) = c + \frac{1}{c} \geq 2.$$

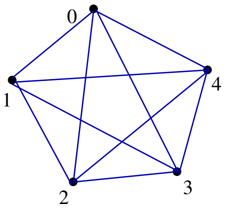


Part A: prove existence of tasks with $\sum_j \psi_j \geq n$

Rough idea:

- use a task for *each pair* of $n+1$ machines





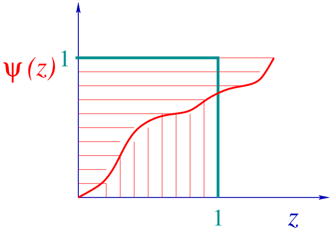
- modelling tasks as edges of a graph: previously *star*, now *clique*
- Sum up every $\psi_{ij}(1)$

$$\sum_i \sum_{j \neq i} \psi_{ij}(1) = \sum_{i,j | i \neq j} (\psi_{ij}(1) + \psi_{ji}(1)) \geq \binom{n+1}{2} \cdot 2 = n \cdot (n+1)$$

$\Rightarrow$ $\exists$ machine i with $\sum_{j \neq i} \psi_{ij}(1) \geq n$

Part A: prove existence of tasks with $\sum_j \psi_j \geq n$

Problem: ψ_{ij} is not linear



Idea: **integral**

$$\int_0^1 (\psi_{ij} + \psi_{ji})\, dz \geq 1 = \int_0^1 2z\, dz$$

$\Rightarrow \exists z \quad (\psi_{ij} + \psi_{ji})(z) \geq 2z$
(mean value theorem)

$\Rightarrow \exists\, z \in (0,1]$ and $\exists$ machine i such that

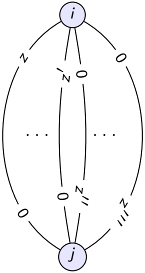
$$\sum_{j \mid j \neq i} \psi_{ij}(z) \geq n \cdot z$$

w.l.o.g. machine $i = 0$

$$\begin{bmatrix} 0 & 0 & \psi_j(z) & 0 & 0 \\ z & & & & \\ & z & & & \\ & & z & & \\ & & & z & \\ & & & & z \end{bmatrix}$$

Problem: As we change these tasks to $s_j = z$, the boundary functions ψ_{0j} change.

Idea: **multi-clique**



- use exp. many parallel tasks (edges) allover in the clique;
- *fix task values* for each edge to independent random $z \in (0, 1]$ and randomly to $0 \longleftrightarrow z$ or to $z \longleftrightarrow 0$; these tasks are trivial and $OPT = 0$
- round down each ψ_{ij}^e to one of finitely many step-functions;
- many parallel edges e between i and j have the same ψ_{ij}^e by pigeonhole; for each machine pair i, j consider only these edges and a single ψ_{ij};
- choose $z \in (0, 1]$ and machine i like above;
- many of the parallel edges will have value 0 for i, and the chosen z as fixed random value...
- **... given that ψ_{ij}^e and the values of *parallel* tasks are independent**

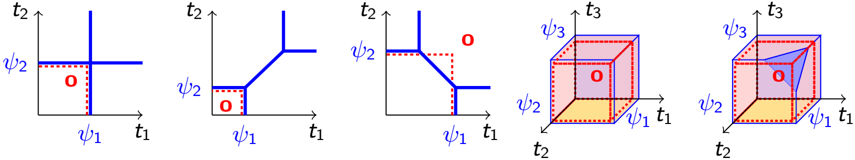
We have shown existence of a machine and tasks with $\sum_j \psi_j(z) \geq n \cdot z$

We call such a task set a *nice star*

$$\begin{bmatrix} 0 & 0 & \dots & 0 & \dots & 0 \\ z & & & & & \\ & z & & & & \\ & & \ddots & & & \\ & & & z & & \\ & & & & \ddots & \\ & & & & & z \end{bmatrix} \rightarrow \begin{bmatrix} \psi_1(z) & \psi_2(z) & \dots & \psi_j(z) & \dots & \psi_n(z) \\ z & & & & & \\ & z & & & & \\ & & \ddots & & & \\ & & & z & & \\ & & & & \ddots & \\ & & & & & z \end{bmatrix}$$

Part B: **But why can we set them to ψ_j at once?**

Good and bad examples:



box | box | non-box | box | non-box

Part B: change every 0 to ψ_j *at once!*

Theorem: If we have exp. many parallel tasks (edges) for each machine j in a *multistar*, then it contains a star which is a box (unless $approx = \infty$).

$$\begin{bmatrix} 0 & 0 & 0 & \psi_1 & 0 & 0 & \psi_2 & 0 & 0 & 0 & \cdots & 0 & \psi_n & 0 & 0 & 0 \\ z & z & z & z & z & & & & & & & & & & & \\ & & & & & z & z & z & z & z & & & & & & \\ & & & & & & & & & & \ddots & & & & & \\ & & & & & & & & & & & z & z & z & z & z \end{bmatrix}$$

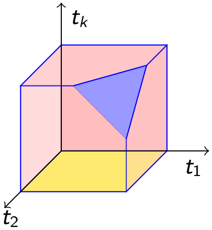
- for each machine j we need *many* parallel tasks with the same ψ_j and allover the same z
- by the above Theorem there exists a star which is a box, and we obtain:

$$ALG \geq \sum_j \psi_j(z) \geq n \cdot z, \qquad OPT = z, \qquad approx \geq n$$

Theorem: If we have exp. many parallel tasks (edges) for each machine j in a *multistar*, then it contains a star which is a box (or $approx = \infty$).

Proof (intuition):

- induction on the number of satellites $k = 2, \ldots, n$;
- we use that *all* truthful mechanisms for 2 machines, 2 parallel tasks are known;
- induction step $(k-1) \to k$: assume $\{1, 2, \ldots, k\}$ is not a box (only its subsets)

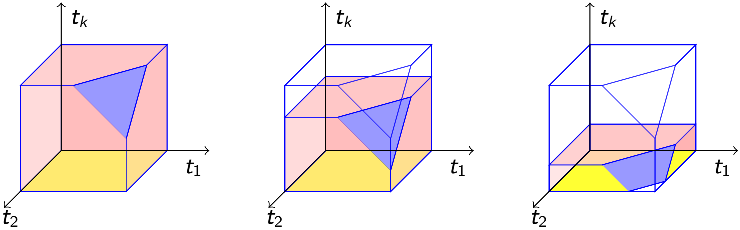


▸

Theorem: If we have exp. many parallel tasks (edges) for each machine j in a *multistar*, then it contains a star which is a box (or *approx* $= \infty$).

Proof (intuition):

- induction on the number of satellites $k = 2, \ldots, n$;
- we use that *all* truthful mechanisms for 2 machines, 2 parallel tasks are known;
- induction step $(k-1) \to k$: assume $\{1, 2, \ldots, k\}$ is not a box (only its subsets)

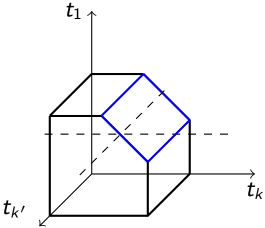


  - in the 'blue' points, if $\psi_k(s_k)$ were linear function, then it would have a non-box subset for some s_k

Theorem: If we have exp. many parallel tasks (edges) for each machine j in a *multistar*, then it contains a star which is a box (or $approx = \infty$).

Proof (intuition):

- induction on the number of satellites $k = 2, \ldots, n$;
- we use that *all* truthful mechanisms for 2 machines, 2 parallel tasks are known;
- induction step $(k-1) \to k$: assume $\{1, 2, \ldots, k\}$ is not a box (only its subsets)



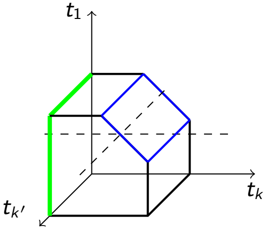
- in the 'blue' points, if $\psi_k(s_k)$ were linear function, then it would have a non-box subset for some s_k

$\Rightarrow$ since $\psi_k(s_k)$ nonlinear, the allocation of task k is independent of $t_{k'}$ of *every parallel* task k'

Theorem: If we have exp. many parallel tasks (edges) for each machine j in a *multistar*, then it contains a star which is a box (or $approx = \infty$).

Proof (intuition):

- induction on the number of satellites $k = 2, \ldots, n$;
- we use that *all* truthful mechanisms for 2 machines, 2 parallel tasks are known;
- induction step $(k-1) \to k$: assume $\{1, 2, \ldots, k\}$ is not a box (only its subsets)



  - in the 'blue' points, if $\psi_k(s_k)$ were linear function, then it would have a non-box subset for some s_k

$\Rightarrow$ since $\psi_k(s_k)$ nonlinear, the allocation of task k is independent of $t_{k'}$ of *every parallel* task k'

$\Rightarrow$ $\{1, 2, \ldots, k'\}$ is a box

$\Rightarrow$ the multistar contains plenty of stars that are boxes

Thank you!