# Selfish, Local and Online Scheduling via Vector Fitting

Danish Kashaev

SODA 2026, CWI Amsterdam

January 13, 2026

# Introduction

A technique, applicable to scheduling/congestion problems with a quadratic objective function $C$, allowing to upper bound

$$\frac{C(x)}{\mathrm{OPT}}$$

where $x$ can be a

- Nash equilibrium ($\rightarrow$ *price of anarchy*)
- Local optimum ($\rightarrow$ *approximation ratio of local search algorithms*)
- Output of an online algorithm ($\rightarrow$ *competitive ratio*)

# Example: Load balancing

**Input:** Set of machines $M$ and jobs $N$ with weights $w_{ij} \in \mathbb{R}_+ \cup \{\infty\}$.

**Goal:** Find an assignment $x \in \{0,1\}^{M \times N}$ of jobs to machines to minimize

$$C(x) = \sum_{i \in M} \ell_i(x)^2$$

where $\ell_i(x) = \sum_{j \in N} w_{ij} x_{ij}$ is the *load* of a machine.

# Example: Load balancing

**Input:** Set of machines $M$ and jobs $N$ with weights $w_{ij} \in \mathbb{R}_+ \cup \{\infty\}$.

**Goal:** Find an assignment $x \in \{0,1\}^{M \times N}$ of jobs to machines to minimize

$$C(x) = \sum_{i \in M} \ell_i(x)^2$$

where $\ell_i(x) = \sum_{j \in N} w_{ij} x_{ij}$ is the *load* of a machine.

**Game-theoretic setting:** each job selfishly picks one of its feasible machines $\mathcal{S}_j \subseteq M$ and wants to minimize its own share

$$C_j(x) = \sum_{i \in M} \ell_i(x) w_{ij} x_{ij}$$

**Online setting:** each job arrives one by one and needs to irrevocably be assigned to a machine by an algorithm

# Example: $R||\sum w_j C_j$

**Input:** $M$, $N$, processing times $p_{ij} \in \mathbb{R}_+ \cup \{\infty\}$, weights $w_j \in \mathbb{R}_+$

**Goal:** Find assignment $x$ (and ordering on each machine) to minimize

$$C(x) = \sum_{j \in N} w_j C_j(x)$$

where $C_j(x)$ is the completion time of $j \in N$.

**Game-theoretic setting:** each job needs to pick one of its feasible machines $\mathcal{S}_j \subseteq M$ in order to minimize its own completion time $C_j(x)$.

**Online setting:** each job arrives one by one and needs to irrevocably be assigned to a machine by an algorithm

# Price of Anarchy

## Definition

An assignment $x$ is a *pure Nash equilibrium* if

$$C_j(x) \leq C_j(x_{-j}, i) \qquad \forall j \in N, \forall i \in \mathcal{S}_j.$$

## Price of Anarchy (PoA)

The *price of anarchy* of a game is the worst-case, over all instances, of

$$\frac{C(x)}{\text{OPT}} \in [1, \infty]$$

where $x$ is any Nash equilibrium.

# Obtainable results

For $R||\sum w_j C_j$, changing the ordering policy on each machine can improve the price of anarchy.

**Paper:** *Inner product spaces for MinSum coordination mechanisms* [CCGMO, STOC 2011]

## Price of anarchy for $R||\sum w_j C_j$

- *Smith's Rule* leads to a PoA of 4
- A preemptive policy called *Proportional Sharing* leads to a PoA of $(3+\sqrt{5})/2 \approx 2.618$
- A randomized policy called *Rand* leads to a PoA of 2.133

## Other PoA results

- Price of anarchy of weighted affine congestion games
- Pure price of anarchy of $P||\sum w_j C_j$

## Local search for $R||\sum w_j C_j$

- Move a job from one machine to another if that improves the objective function gives an approximation ratio of $\approx 2.618$ [CM22]
- Best known combinatorial approximation algorithm based on local search achieves a ratio of $\approx 1.809$ [CGV17]

## Online algorithms for $R||\sum w_j C_j$

- One deterministic (greedy) algorithm [GMUX20] and another randomized algorithm achieving the optimal competitive ratio of 4

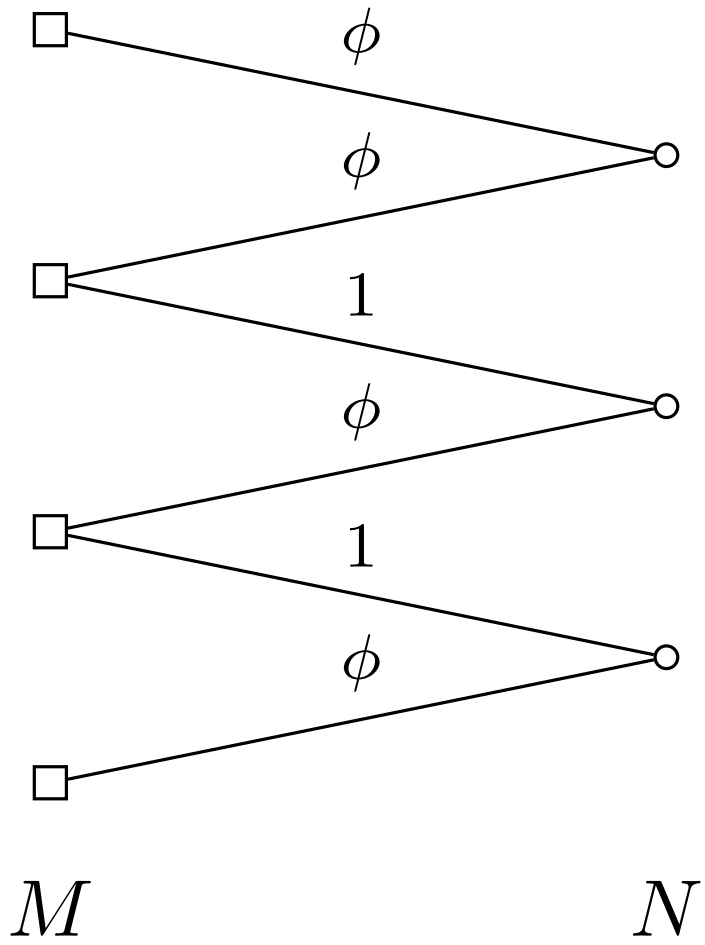
## Follow-up work

**Paper:** *Improved online load balancing in the two-norm* [Borst, K. 25]

- Greedy algorithm is $\approx 5.828$-competitive [AAG+95], randomized algorithm based on independent rounding is 5-competitive [Car08].
- Primal-dual randomized algorithm achieving $\approx 4.98$ using negative correlation rounding

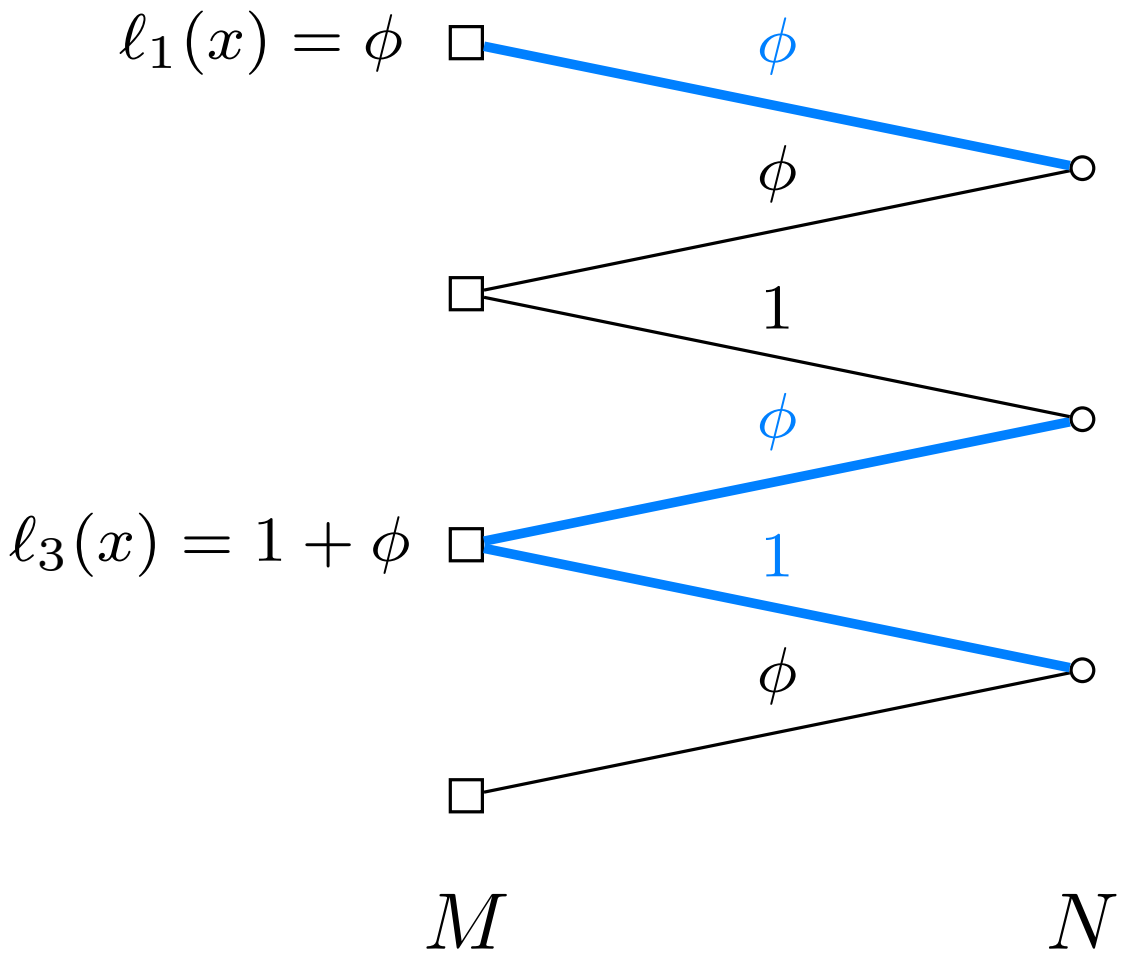
# Example: load balancing

Let $\phi \approx 1.618$ such that $\phi^2 = 1 + \phi$. Instance with $|M| = |N| + 1 = n + 1$



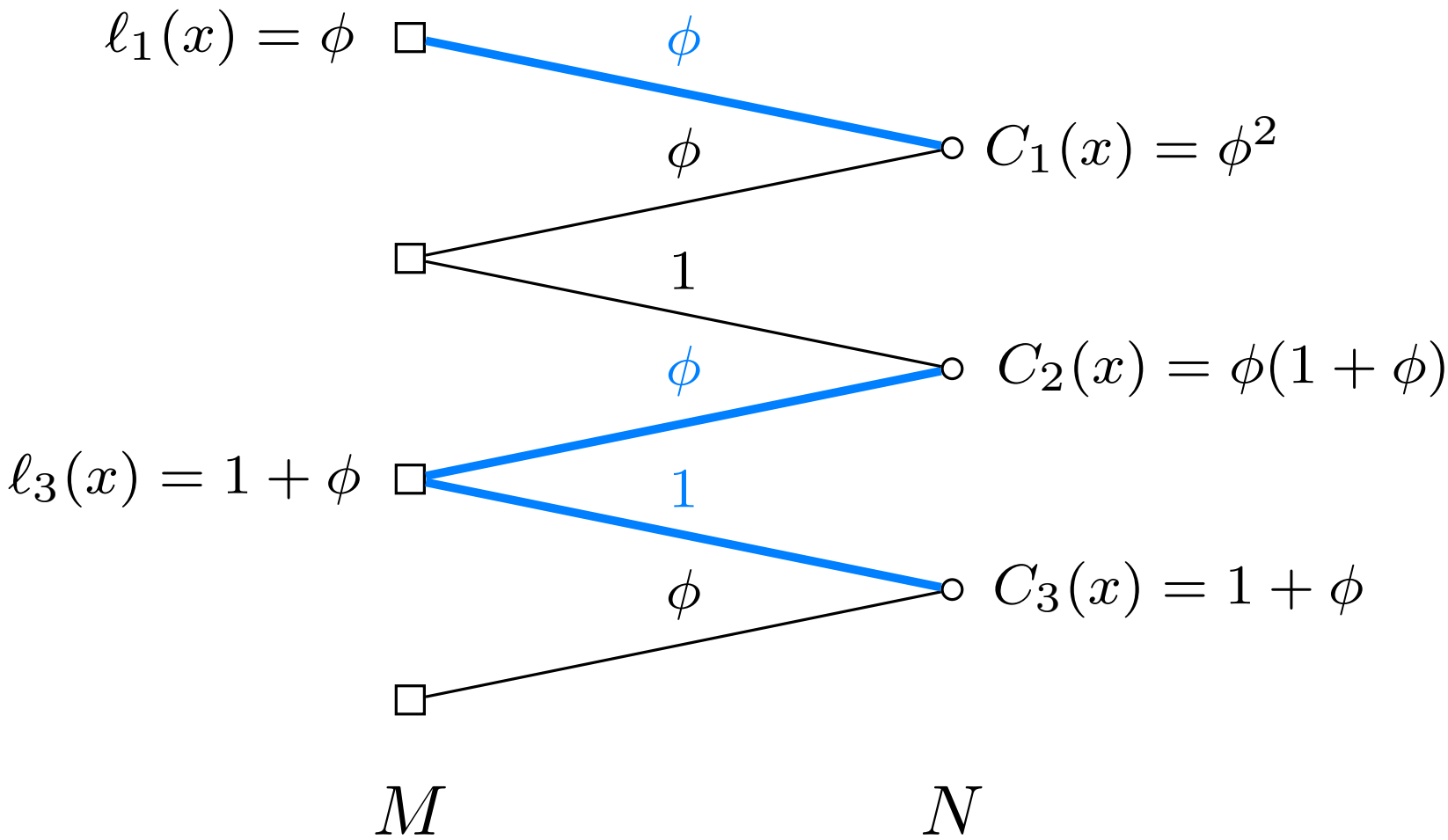
# Example: load balancing

Let $\phi \approx 1.618$ such that $\phi^2 = 1 + \phi$. Instance with $|M| = |N| + 1 = n + 1$



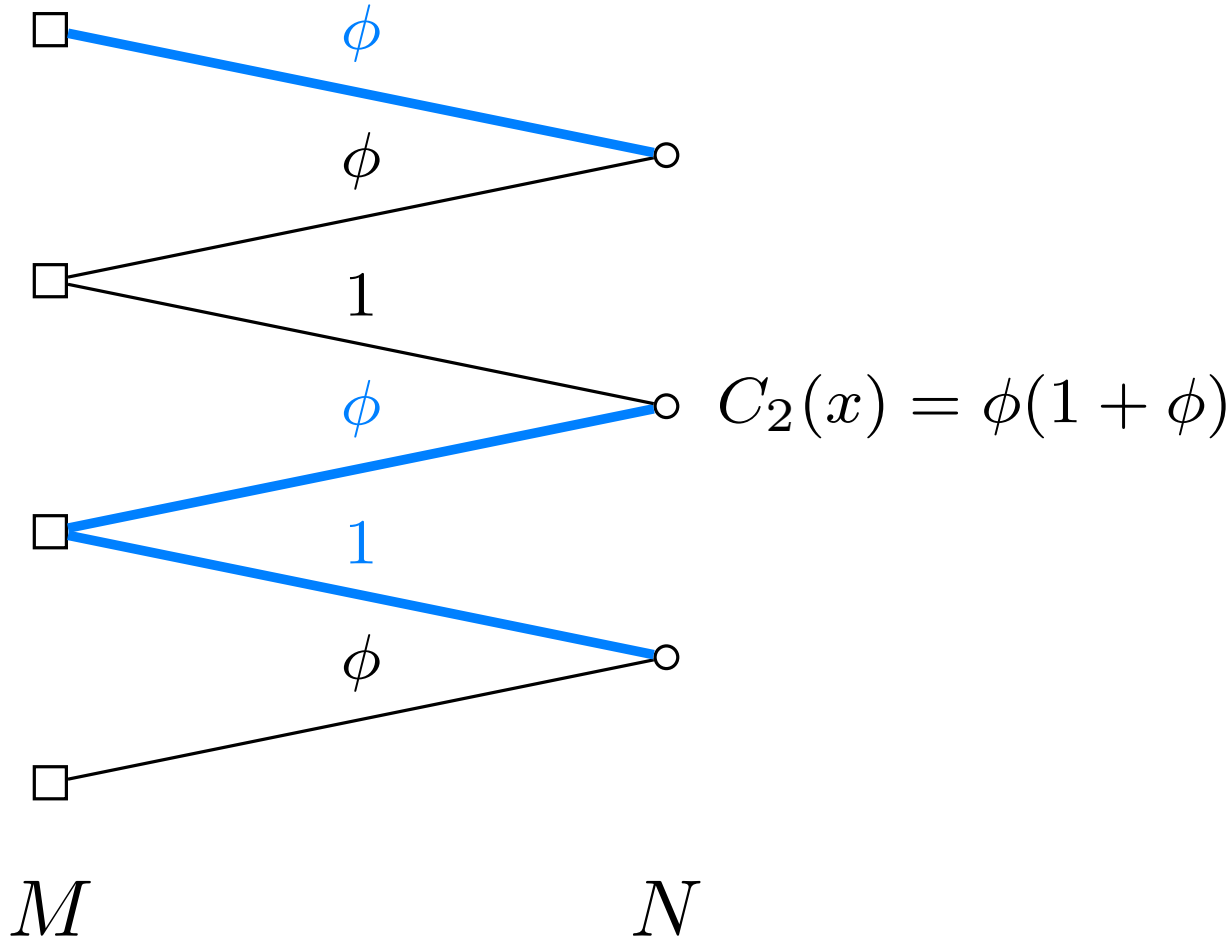
# Example: load balancing

Let $\phi \approx 1.618$ such that $\phi^2 = 1 + \phi$. Instance with $|M| = |N| + 1 = n + 1$



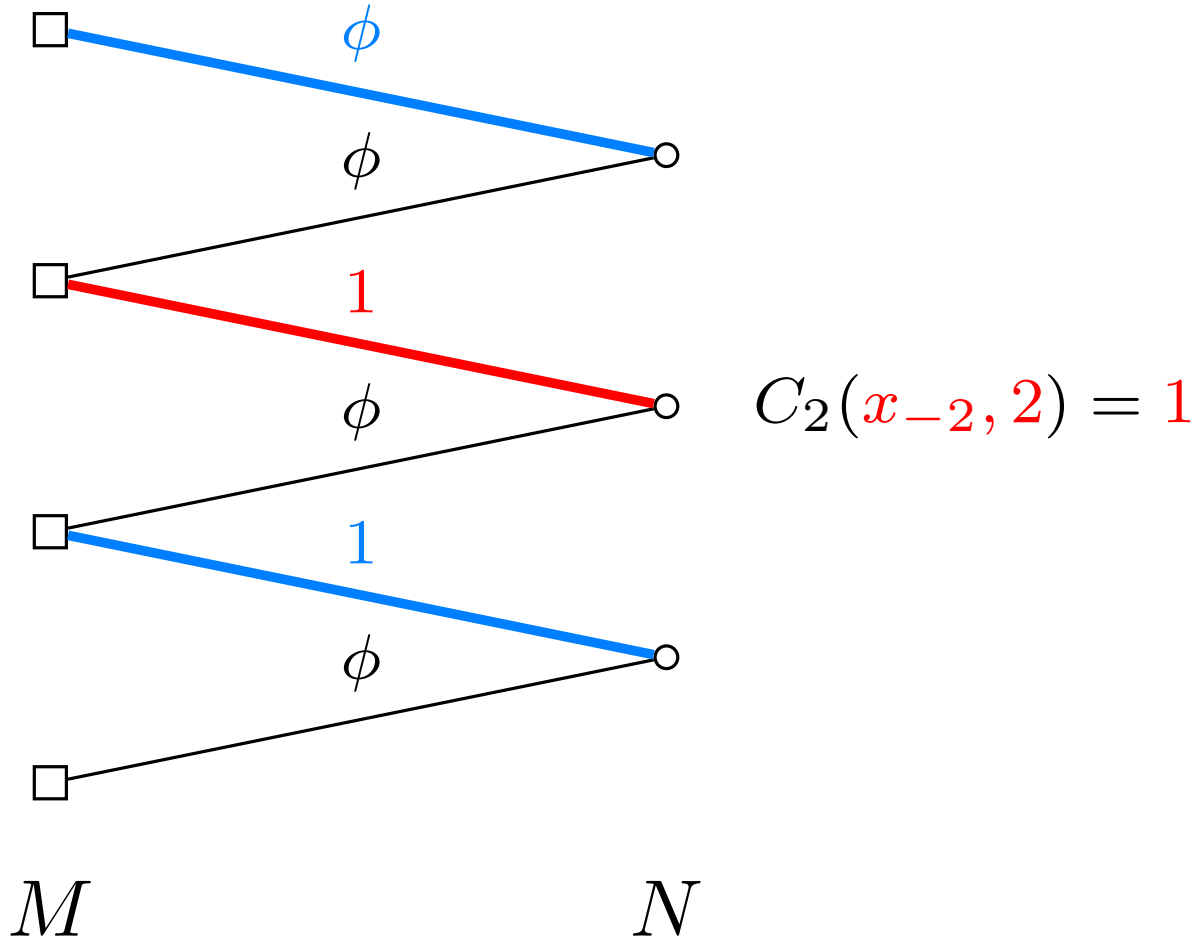
# Example: load balancing

Let $\phi \approx 1.618$ such that $\phi^2 = 1 + \phi$. Instance with $|M| = |N| + 1 = n + 1$



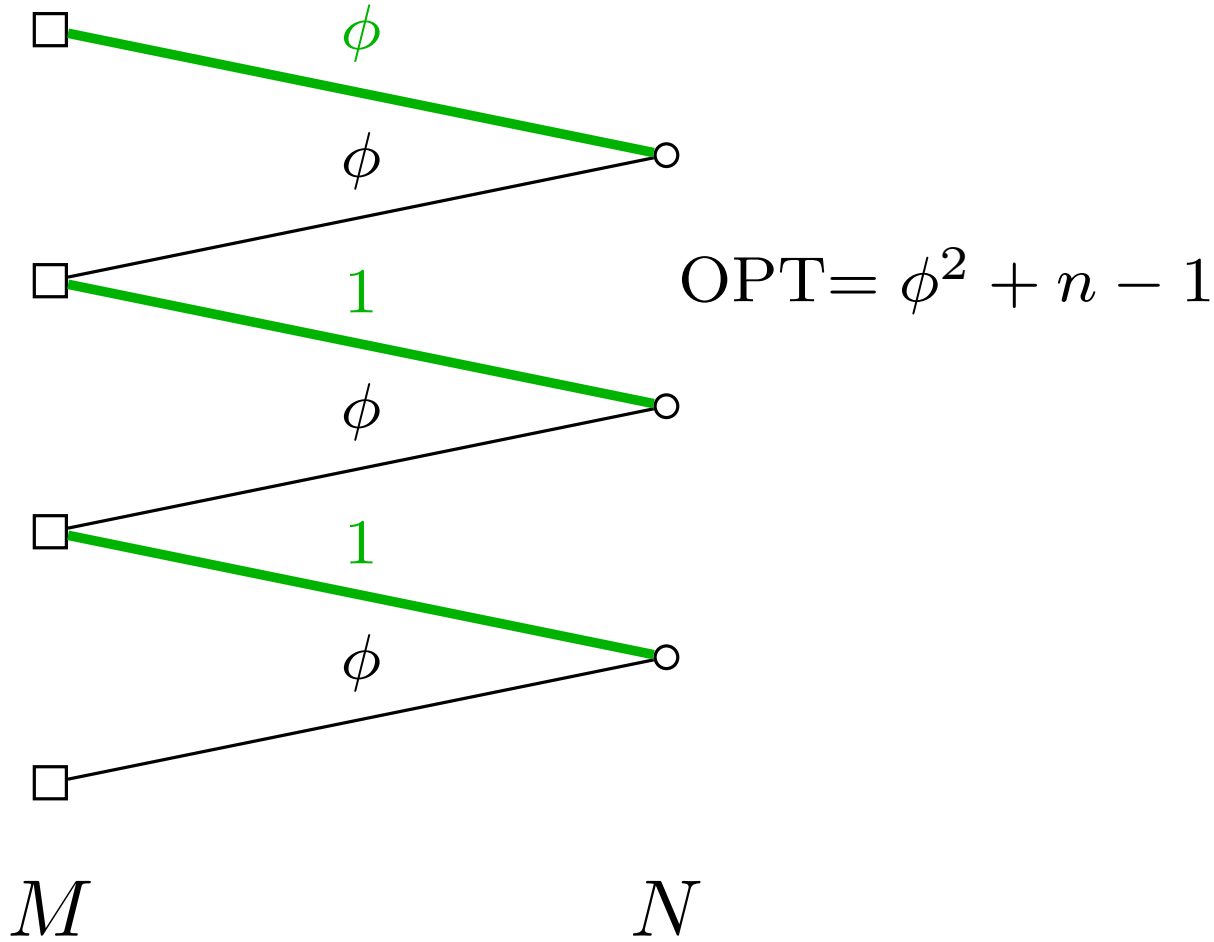
# Example: load balancing

Let $\phi \approx 1.618$ such that $\phi^2 = 1 + \phi$. Instance with $|M| = |N| + 1 = n + 1$



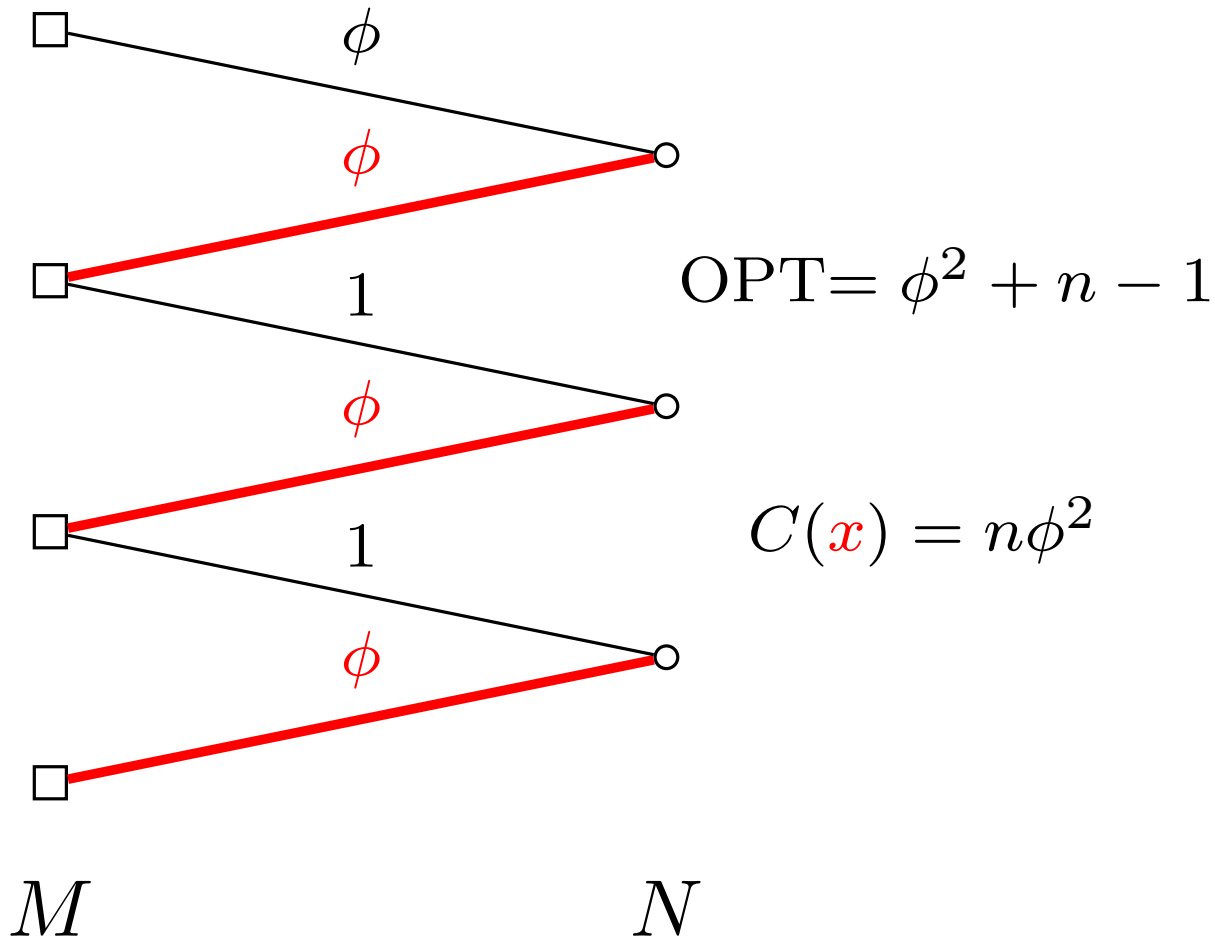
# Example: load balancing

Let $\phi \approx 1.618$ such that $\phi^2 = 1 + \phi$. Instance with $|M| = |N| + 1 = n + 1$



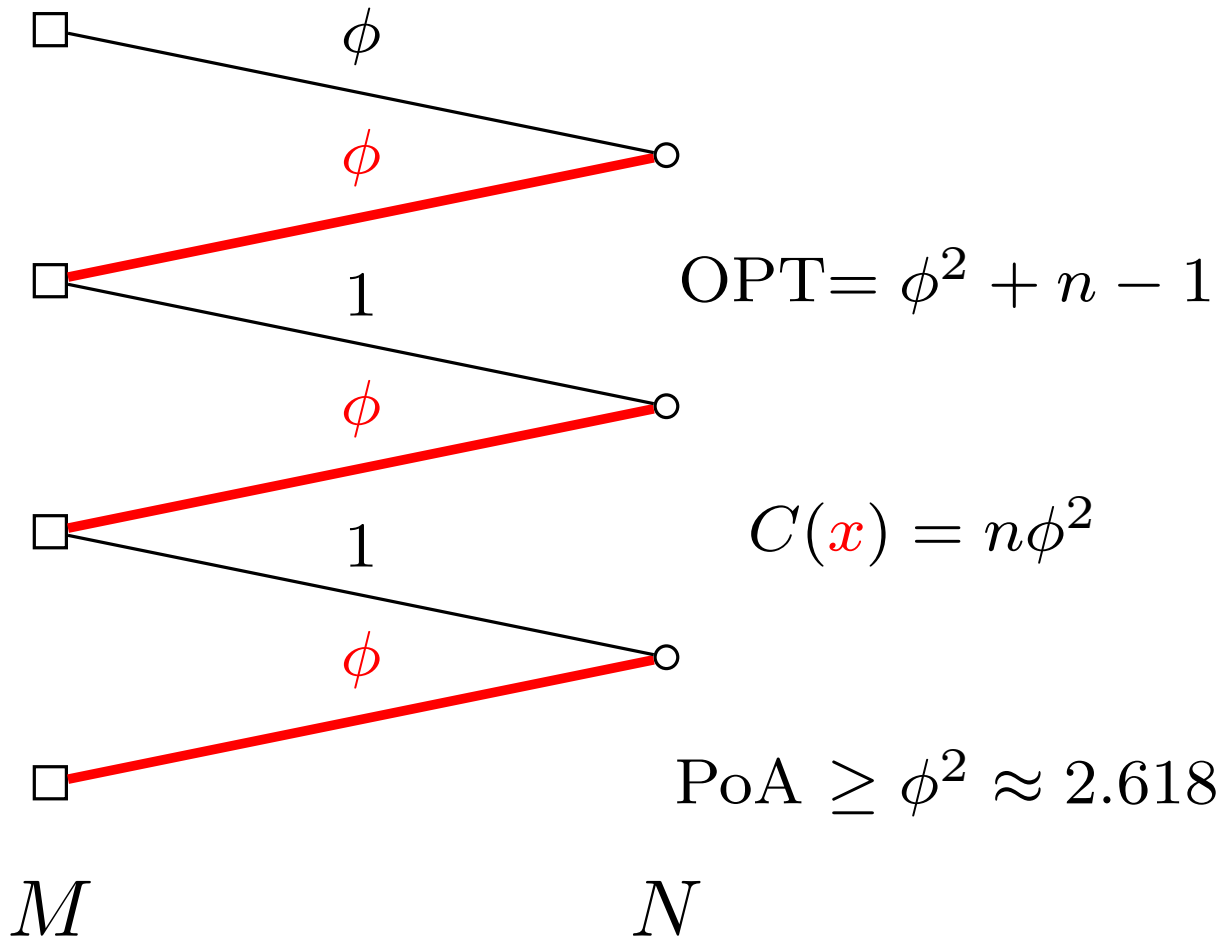
# Example: load balancing

Let $\phi \approx 1.618$ such that $\phi^2 = 1 + \phi$. Instance with $|M| = |N| + 1 = n + 1$



# Example: load balancing

Let $\phi \approx 1.618$ such that $\phi^2 = 1 + \phi$. Instance with $|M| = |N| + 1 = n + 1$



# Dual fitting: high-level view

Exact binary quadratic program to compute OPT:

$$\begin{aligned} \min\ & C(x) \\ & \sum_{i \in \mathcal{S}_j} x_{ij} = 1 \qquad \forall j \in N \\ & x_{ij} \in \{0,1\} \qquad \forall j \in N, \forall i \in \mathcal{S}_j. \end{aligned}$$

**Goal:** [Kulkarni-Mirrokni, 2014]

For any Nash equilibrium $x$, find a feasible dual solution to some convex relaxation with objective value $V$ such that

$$V \geq \rho\, C(x) \quad \text{for some} \quad \rho \in [0,1]$$

$$\implies \text{PoA} = \frac{C(x)}{C(x^*)} \leq \frac{C(x)}{V} \leq \frac{1}{\rho}$$

# Convex SDP relaxation

Since $C(x)$ is quadratic, then

$$C(x) = \langle C, X \rangle := \mathrm{Tr}(CX)$$

for some symmetric matrix $C$, where $X = (1, x)(1, x)^T$ encodes all the linear and quadratic terms of $x$.

A semidefinite programming relaxation:

$$\begin{aligned}
\min \langle C, X \rangle & \\
\sum_{i \in \mathcal{S}_j} X_{\{ij,\, ij\}} &= 1 && \forall j \in N \\
X_{\{0,0\}} &= 1 && \\
X_{\{0,\, ij\}} &= X_{\{ij,\, ij\}} && \forall j \in N, i \in \mathcal{S}_j \\
X &\succeq 0 &&
\end{aligned}$$

# The dual SDP

**Variables:**

- Scalars $y_j \in \mathbb{R}$ for every $j \in N$
- Vector $v_0 \in \mathbb{R}^M$
- Vectors $v_{ij} \in \mathbb{R}^M$ for every $j \in N, i \in \mathcal{S}_j$

$$\max \sum_{j \in N} y_j - \frac{1}{2}\|v_0\|^2$$

$$y_j \leq C_{\{ij,\, ij\}} - \frac{1}{2}\|v_{ij}\|^2 + \langle v_0, v_{ij} \rangle \qquad \forall j \in N, i \in \mathcal{S}_j$$

$$\langle v_{ij}, v_{i'k} \rangle \leq 2\, C_{\{ij,\, i'k\}} \qquad \forall (i,j) \neq (i',k) \text{ with } j,k \in N$$

# Dual fitting: high-level view

$$\max \sum_{j \in N} y_j - \frac{1}{2}\|v_0\|^2$$

$$y_j \leq C_{\{ij,\, ij\}} - \frac{1}{2}\|v_{ij}\|^2 + \langle v_0, v_{ij} \rangle \qquad \forall j \in N, i \in \mathcal{S}_j \tag{1}$$

$$\langle v_{ij}, v_{i'k} \rangle \leq 2\, C_{\{ij,\, i'k\}} \qquad \forall (i,j) \neq (i',k) \text{ with } j,k \in N$$

- *Price of anarchy*: make (1) correspond to Nash equilibria inequalities

$$C_j(x) \leq C_j(x_{-j}, i) \qquad \forall j \in N, i \in \mathcal{S}_j.$$

- *Local search:* make (1) correspond to local optima inequalities
- *Online algorithms:* make (1) correspond to inequalities satisfied by an online algorithm at every time step

# Back to our example: load balancing

**Specialize the SDP:**

$$\max \sum_{j \in N} y_j - \frac{1}{2}\|v_0\|^2$$

$$y_j \leq w_{ij}^2 - \frac{1}{2}\|v_{ij}\|^2 + \langle v_0, v_{ij} \rangle \qquad \forall j \in N, i \in \mathcal{S}_j$$

$$\langle v_{ij}, v_{i'k} \rangle \leq 2\, w_{ij}\, w_{ik}\, \mathbb{1}_{\{i=i'\}} \qquad \forall (i,j) \neq (i',k) \text{ with } j,k \in N$$

**Nash equilibria inequalities:**

$$C_j(x) \leq w_{ij}^2 + w_{ij}\, \ell_i(x) \qquad \forall j \in N, \forall i \in \mathcal{S}_j.$$

# Back to our example: load balancing

**Specialize the SDP:**

$$\max \sum_{j \in N} y_j - \frac{1}{2}\|v_0\|^2$$

$$y_j \leq w_{ij}^2 - \frac{1}{2}\|v_{ij}\|^2 + \langle v_0, v_{ij} \rangle \qquad \forall j \in N, i \in \mathcal{S}_j$$

$$\langle v_{ij}, v_{i'k} \rangle \leq 2\, w_{ij}\, w_{ik}\, \mathbb{1}_{\{i=i'\}} \qquad \forall (i,j) \neq (i',k) \text{ with } j,k \in N$$

**Nash equilibria inequalities:**

$$C_j(x) \leq w_{ij}^2 + w_{ij}\, \ell_i(x) \qquad \forall j \in N, \forall i \in \mathcal{S}_j.$$

**Fitting:**

- $v_{ij}(i') = \alpha\, w_{ij}\, \mathbb{1}_{\{i=i'\}}$ for some $\alpha \in [0, \sqrt{2}]$
- $v_0(i) = \beta\, \ell_i(x)$ for some $\beta \geq 0$
- $y_j = \gamma\, C_j(x)$ for some $\gamma \geq 0$

**SDP constraints:**

$$y_j \leq w_{ij}^2 - \frac{1}{2}\|v_{ij}\|^2 + \langle v_0, v_{ij}\rangle$$

$$\iff \gamma\, C_j(x) \leq \left(1 - \frac{\alpha^2}{2}\right) w_{ij}^2 + \alpha\beta\, w_{ij}\ell_i(x) \tag{2}$$

**Constraints on the constants:** $\gamma = 1 - \frac{\alpha^2}{2} = \alpha\beta$

**SDP objective:** $\sum_{j\in N} y_j - \frac{1}{2}\|v_0\|^2 = \left(\alpha\beta - \frac{\beta^2}{2}\right) C(x)$

**SDP objective:** $\sum_{j \in N} y_j - \frac{1}{2}\|v_0\|^2 = \left(\alpha\beta - \frac{\beta^2}{2}\right) C(x)$

**Goal:**

For any Nash equilibrium $x$, find a feasible dual solution such that

$$\sum_{j \in N} y_j - \frac{1}{2}\|v_0\|^2 \geq \rho\, C(x) \quad \text{for some} \quad \rho \in [0, 1]$$

**SDP objective:** $\sum_{j \in N} y_j - \frac{1}{2}\|v_0\|^2 = \left(\alpha\beta - \frac{\beta^2}{2}\right) C(x)$

**Goal:**

For any Nash equilibrium $x$, find a feasible dual solution such that

$$\sum_{j \in N} y_j - \frac{1}{2}\|v_0\|^2 \geq \rho\, C(x) \quad \text{for some} \quad \rho \in [0,1]$$

**Solve:**

$$\max\left\{\alpha\beta - \frac{\beta^2}{2} : \alpha\beta = 1 - \frac{\alpha^2}{2}, \alpha \in [0,\sqrt{2}], \beta \geq 0\right\} = \frac{2}{3+\sqrt{5}} = \frac{1}{\phi^2}$$

**Extension to congestion games**

Also works for affine *congestion games*, where $\mathcal{S}_j \subseteq 2^M$.

# Conclusion

## Conclusion

- Unified proof technique for scheduling and congestion problems whose optimal solution can be cast as a binary quadratic program
- SDP relaxation can be obtained by the first round of Lasserre/SoS hierarchy
- Recovers and unifies numerous results
- Works in game-theoretic, local search and online settings

## Future work ideas

- Apply this technique to new problems with a quadratic objective
- Design a scheduling policy for $R||\sum w_j C_j$ improving PoA of $\approx 2.133$
- Extend this technique to higher degree polynomial objectives

# Conclusion

## Conclusion

- Unified proof technique for scheduling and congestion problems whose optimal solution can be cast as a binary quadratic program
- SDP relaxation can be obtained by the first round of Lasserre/SoS hierarchy
- Recovers and unifies numerous results
- Works in game-theoretic, local search and online settings

## Future work ideas

- Apply this technique to new problems with a quadratic objective
- Design a scheduling policy for $R||\sum w_j C_j$ improving PoA of $\approx 2.133$
- Extend this technique to higher degree polynomial objectives

*Thanks!*