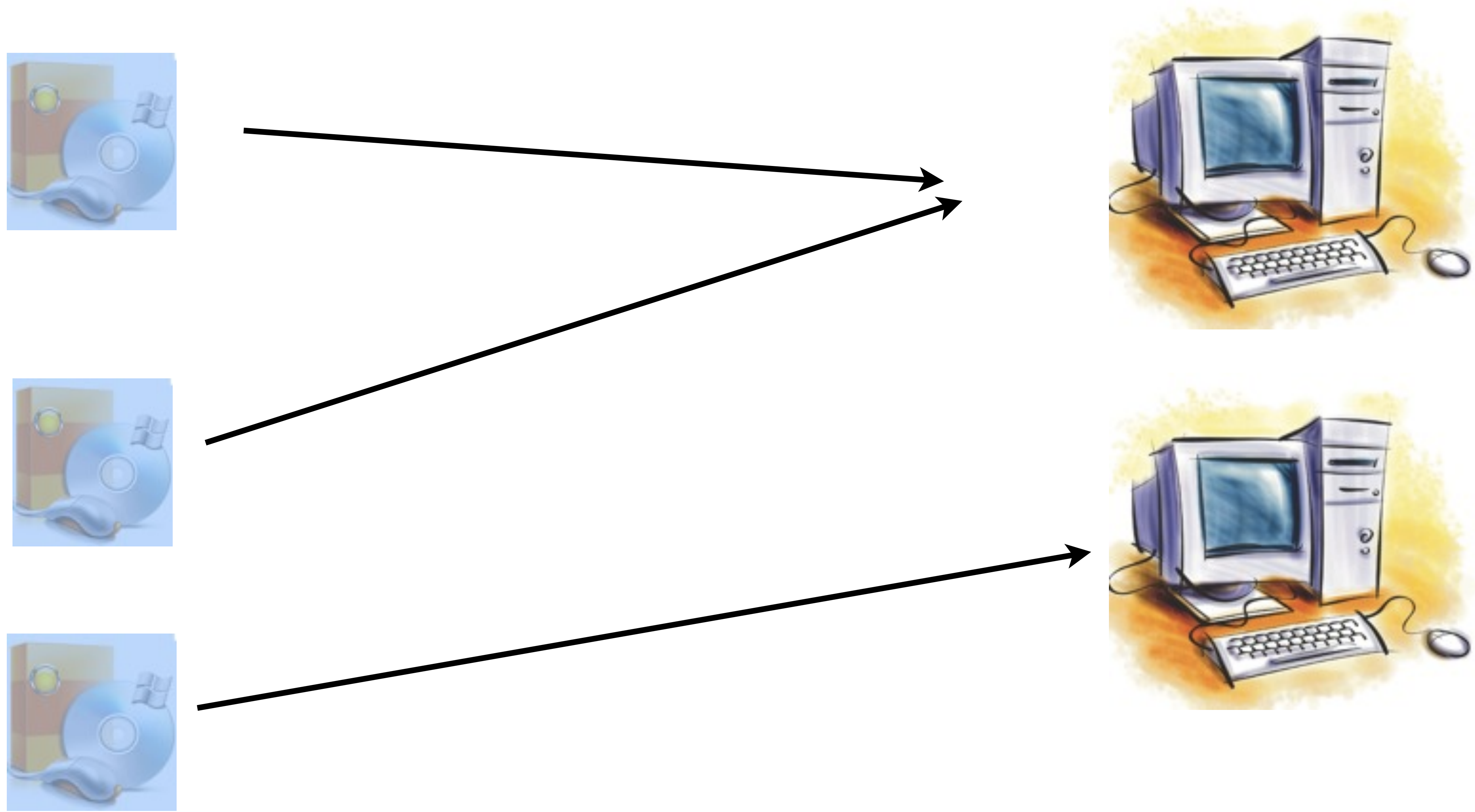
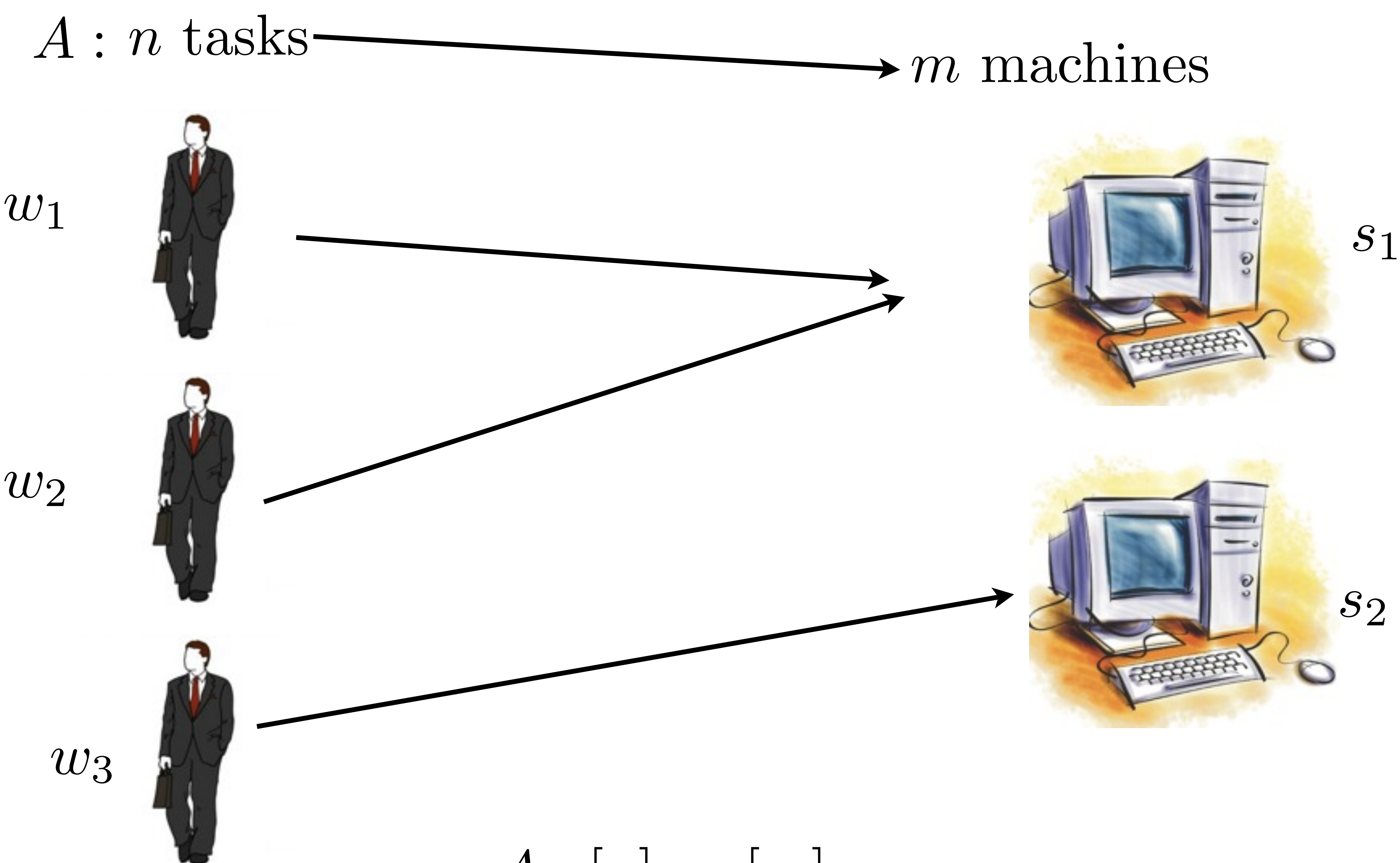
Selfish Load Balancing





$$A : [n] \to [m]$$

$A : n$ tasks → m machines

w_1

w_2

s_j

$$\text{load} = l_j = \sum_{i \in A^{-1}(j)} \frac{w_i}{s_j}$$

$$\text{cost}(1) = \text{cost}2 = l_j$$

$A : n$ tasks $\longrightarrow$ m machines

w_1

w_2

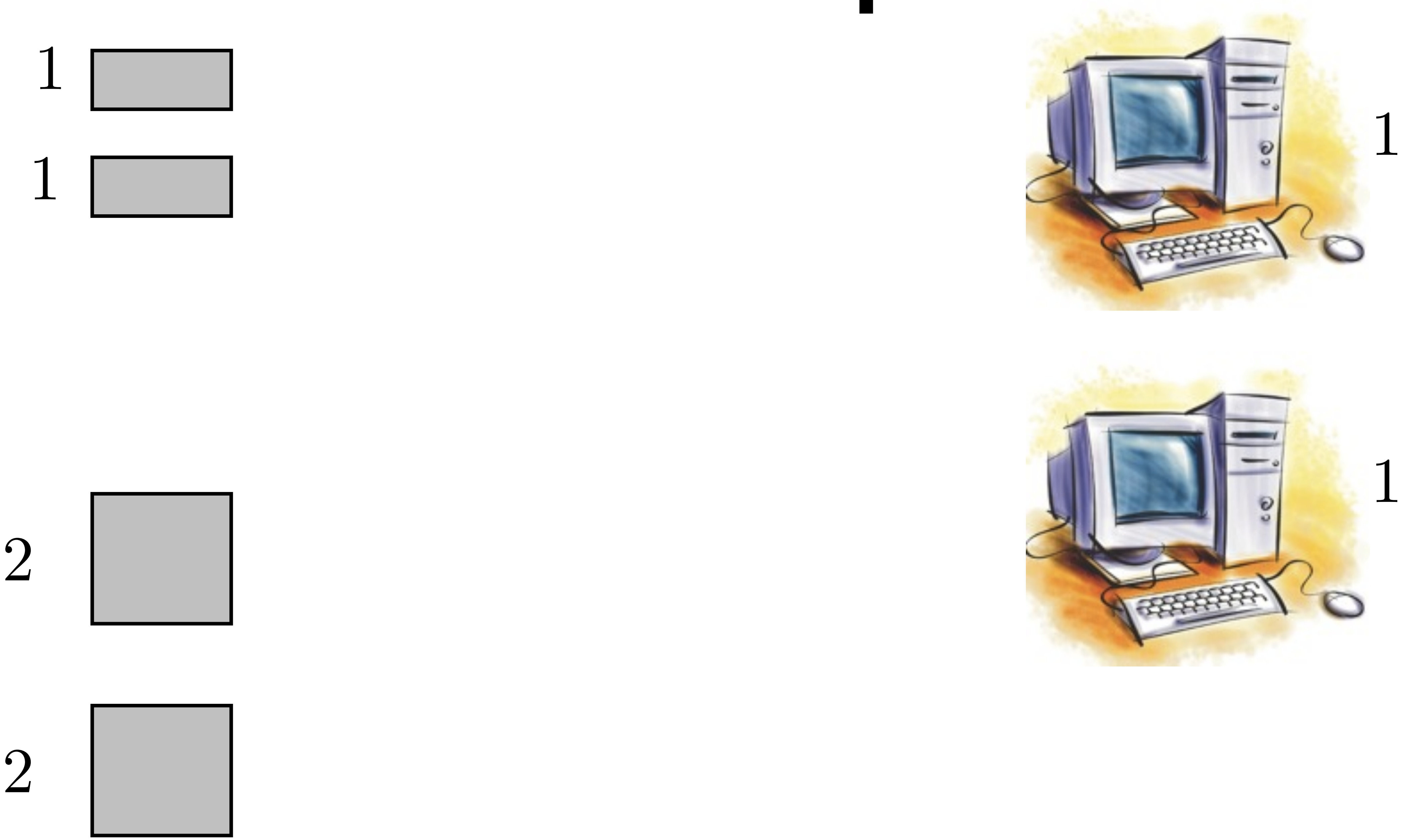
s_j

$$\text{load} = l_j = \sum_{i \in A^{-1}(j)} \frac{w_i}{s_j}$$

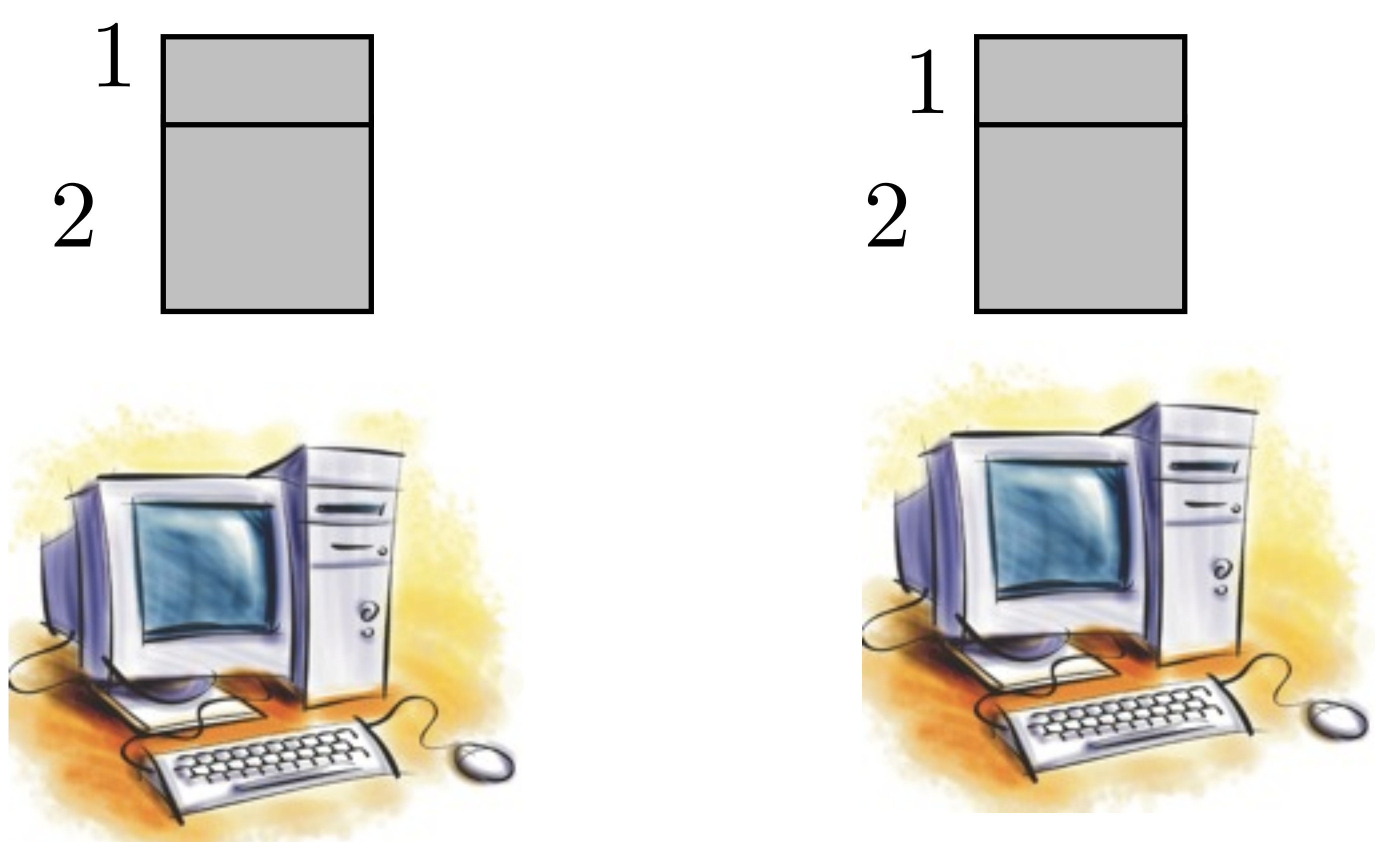
$$\text{cost}(i) = l_{A(i)}$$

social cost= maximum load

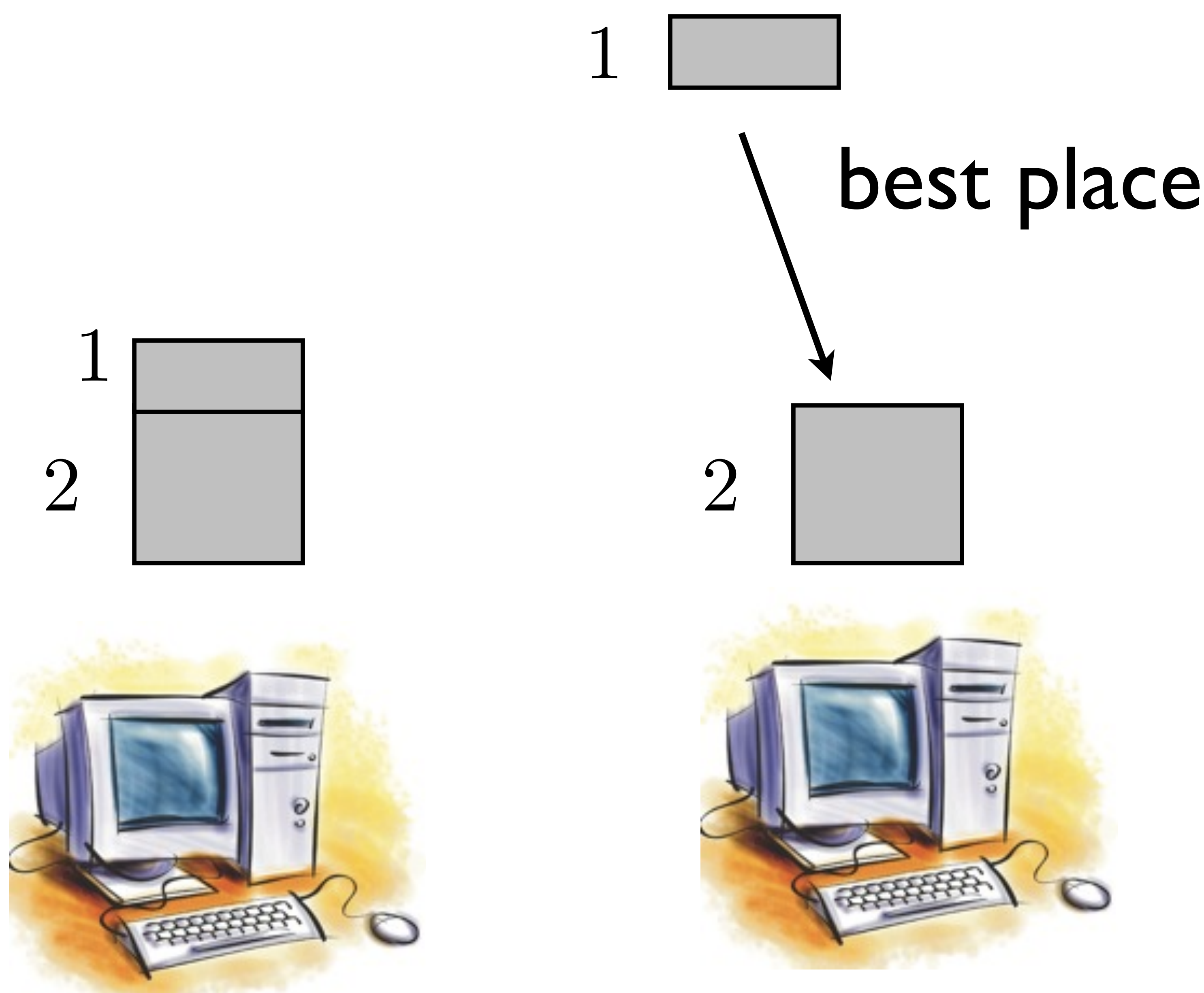
An example



An example



An example



Outline

- Pure Nash Equilibrium.
- Identical machines
    - Price of Anarchy.
    - Transforming to NE.
- Uniformly related machines
    - Price of Anarchy.
    - Finding NE.

Proposition

- Every instance of load balancing has at least one pure NE.

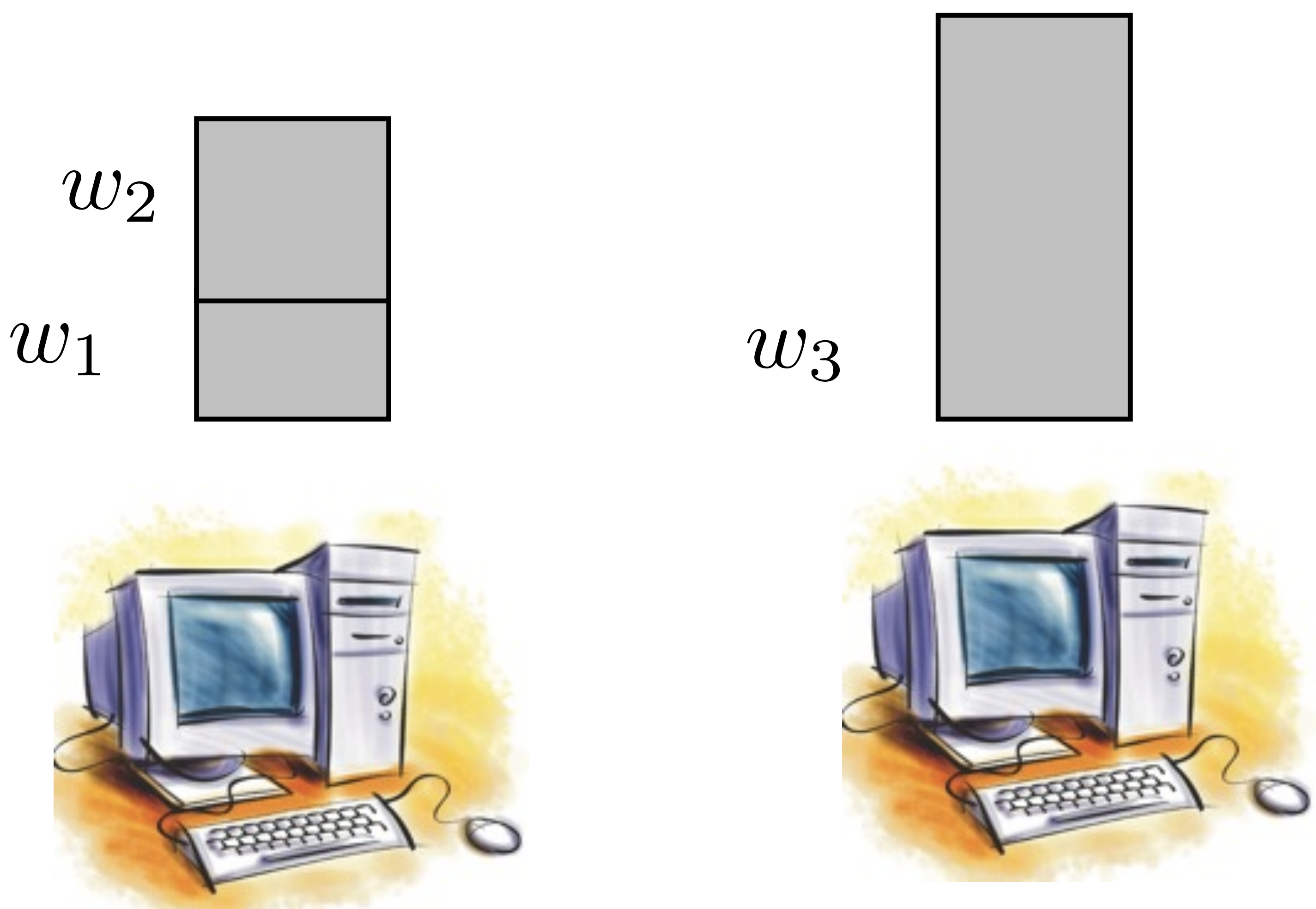
Proof

- An assignment A induces a sorted load vector $(\lambda_1, \lambda_2, ..., \lambda_m)$
- If its not a NE then an agent could change getting a smaller sorted vector.

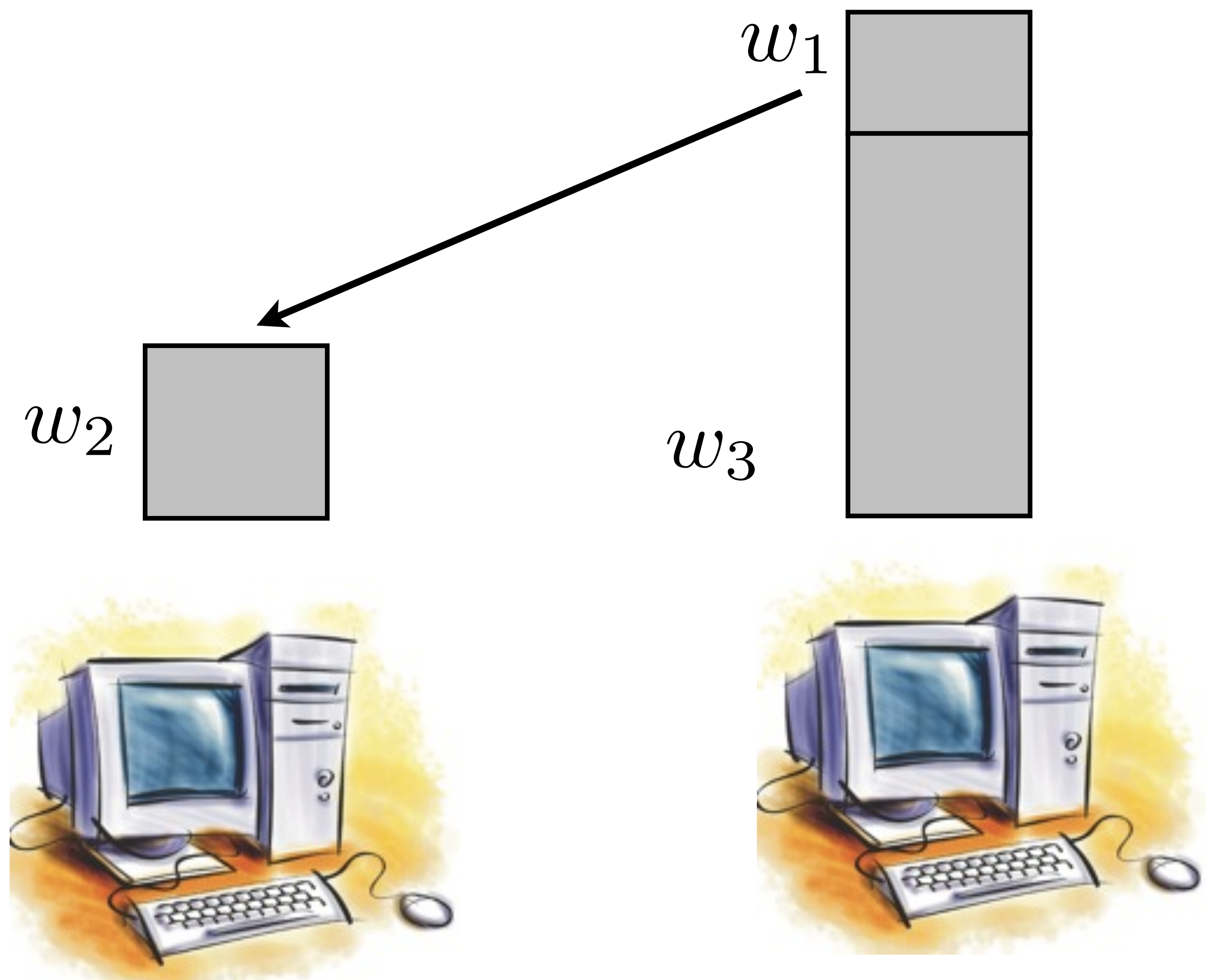
Identical machines
Price of Anarchy.

3 tasks

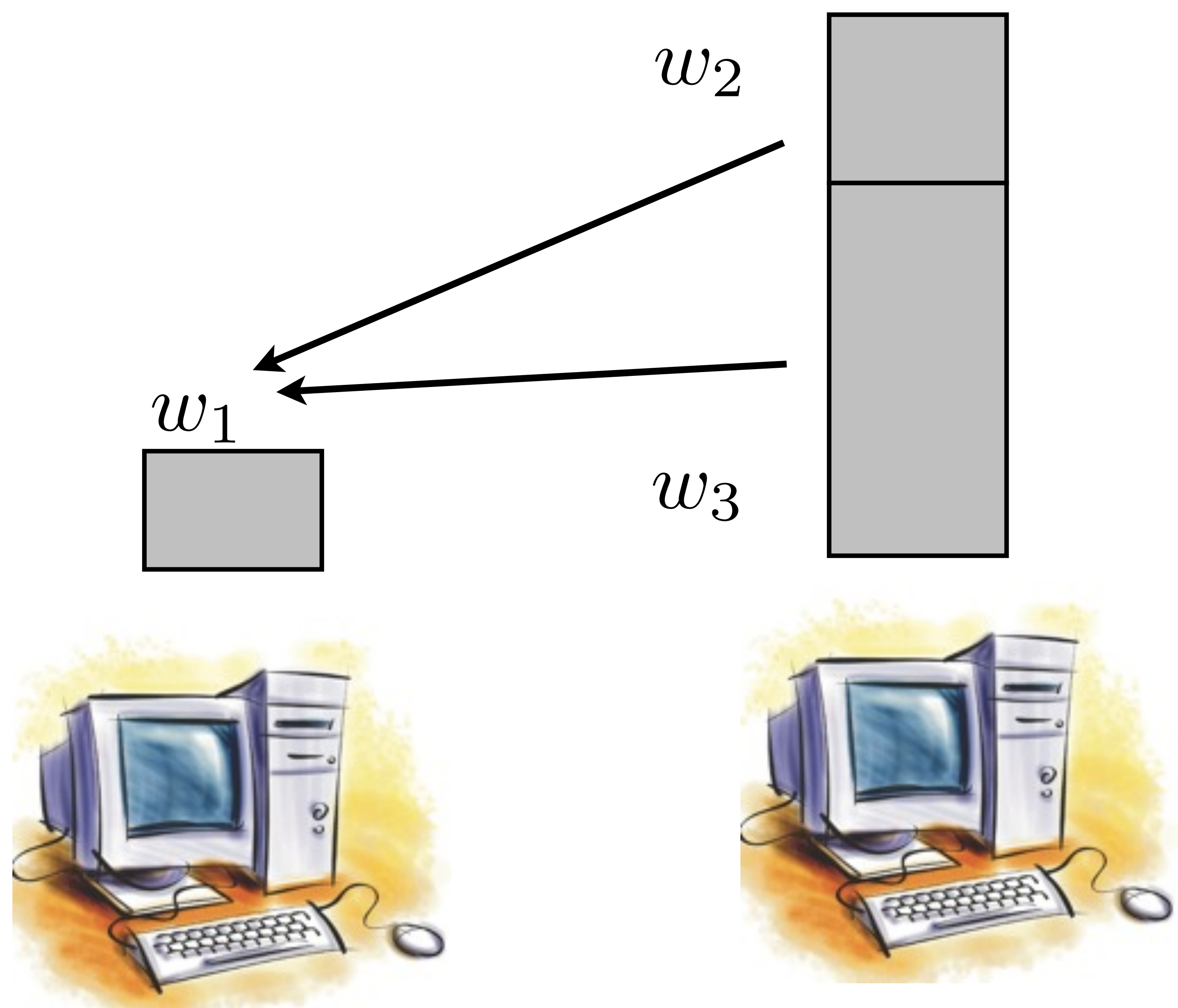
$$\mathrm{cost}(A) = \max(w_3, w_1 + w_2)$$



Optimum cost

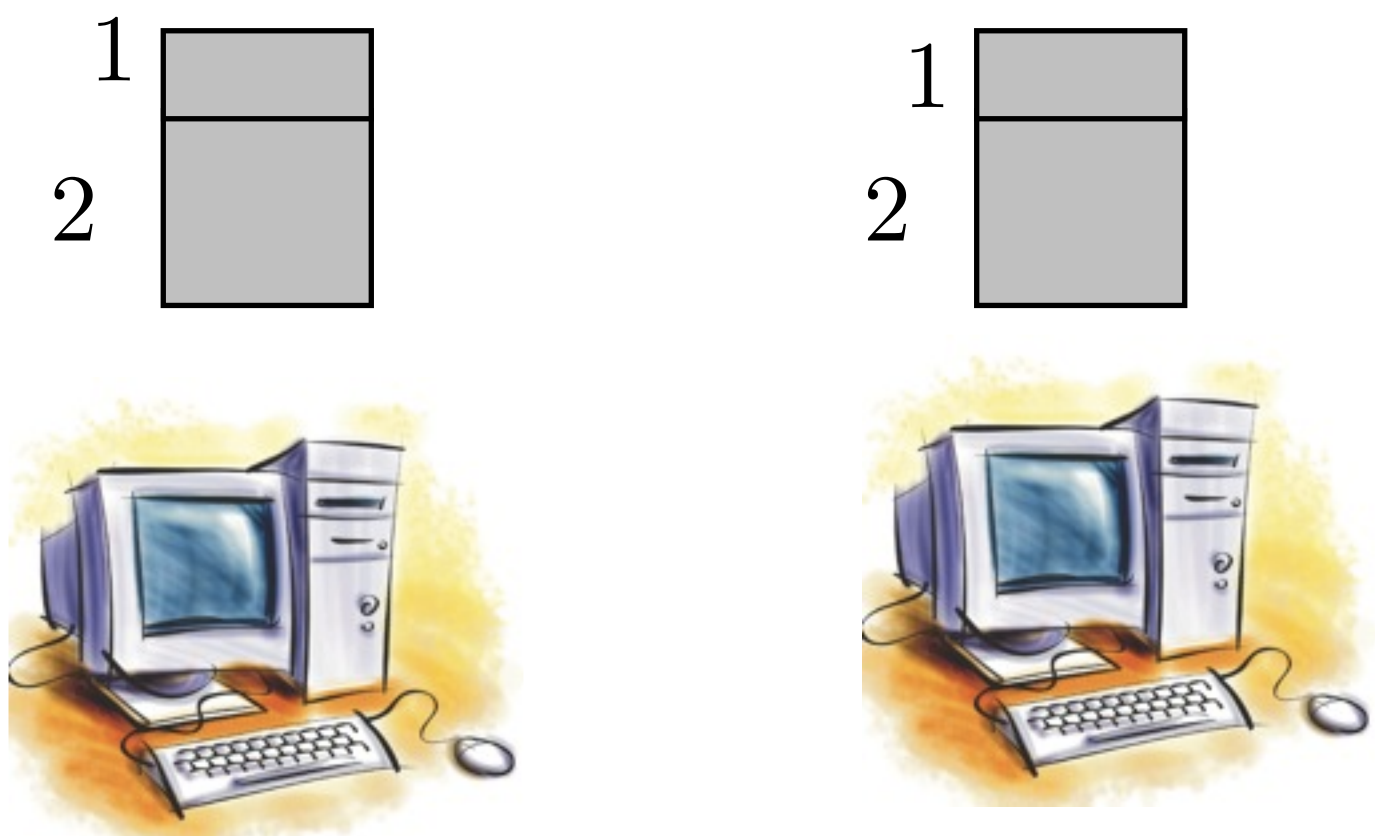


Optimum cost



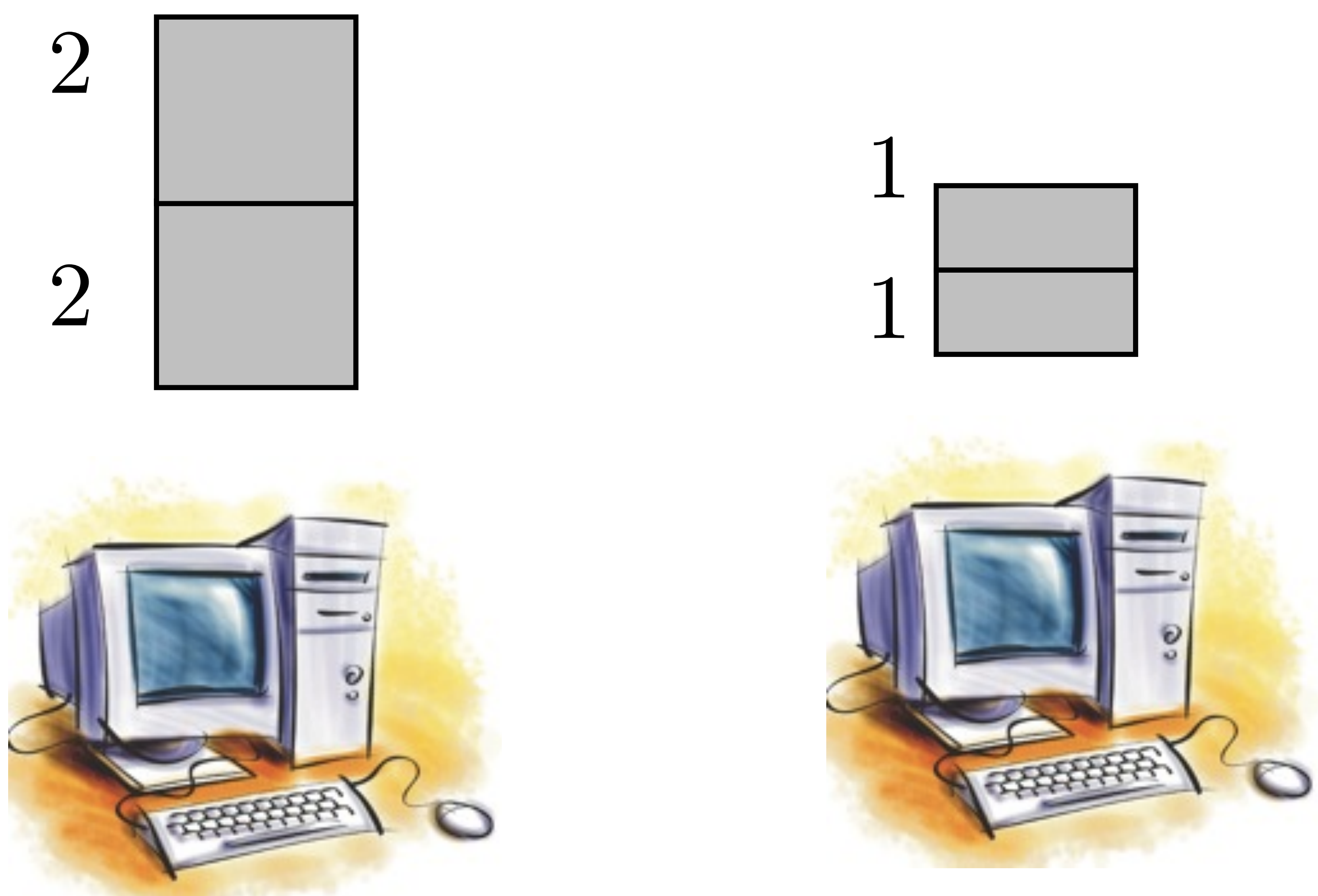
4 tasks

$$\mathrm{opt}(G) = 3$$



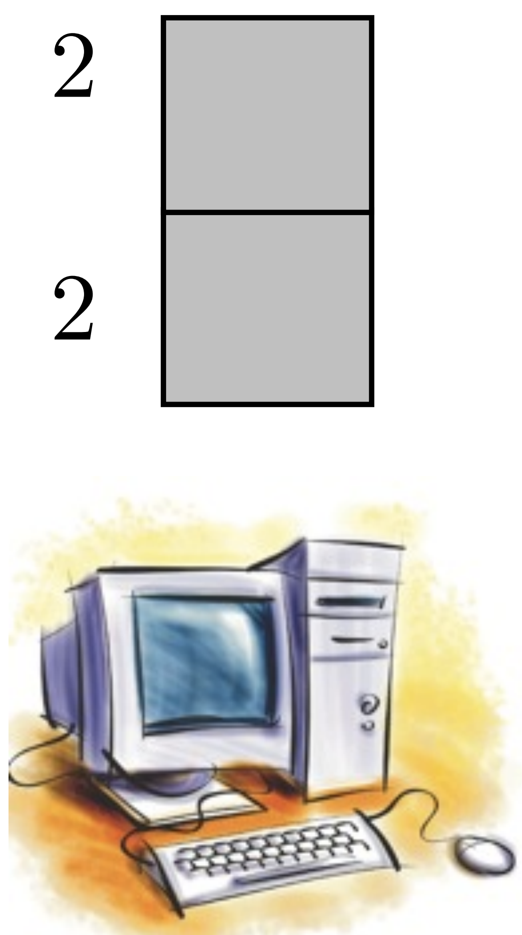
4 tasks

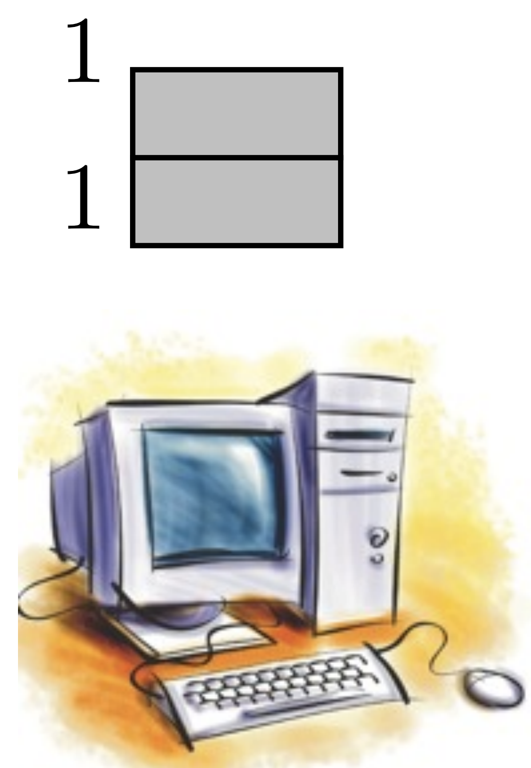
$$\text{cost}(A) = 4$$



4 tasks

$$\frac{\mathrm{cost}(A)}{\mathrm{opt}(G)} = \frac{4}{3} = 2 - \frac{2}{3} = 2 - \frac{2}{m+1}$$





Price of Anarchy

$$PoA(m) = \max_{G \in \mathcal{G}(m)} \max_{P \in \text{Nash}(G)} \frac{\text{cost}(\text{P})}{\text{opt}(G)}$$

Theorem

- Consider an instance G of the load balancing game and an assignment A that is a NE. Then

$$\text{cost}(A) \leq \left(2 - \frac{2}{m+1}\right) \text{opt}(G)$$

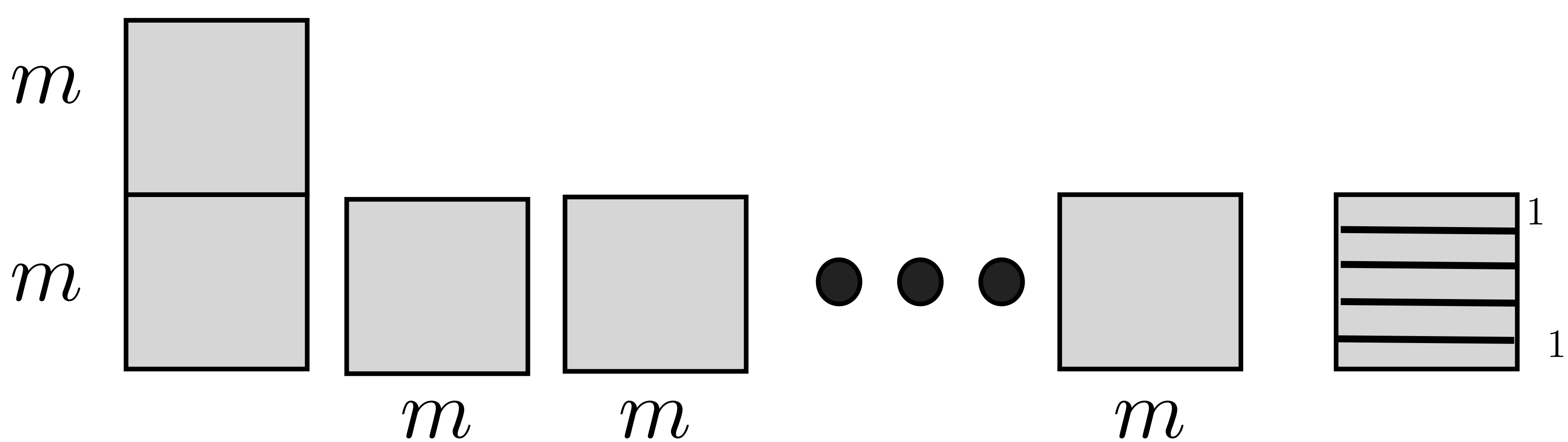
Hence

$$PoA(m) \leq 2 - \frac{2}{m+1}$$

Tightness of bound

m tasks of size m
m tasks of size 1

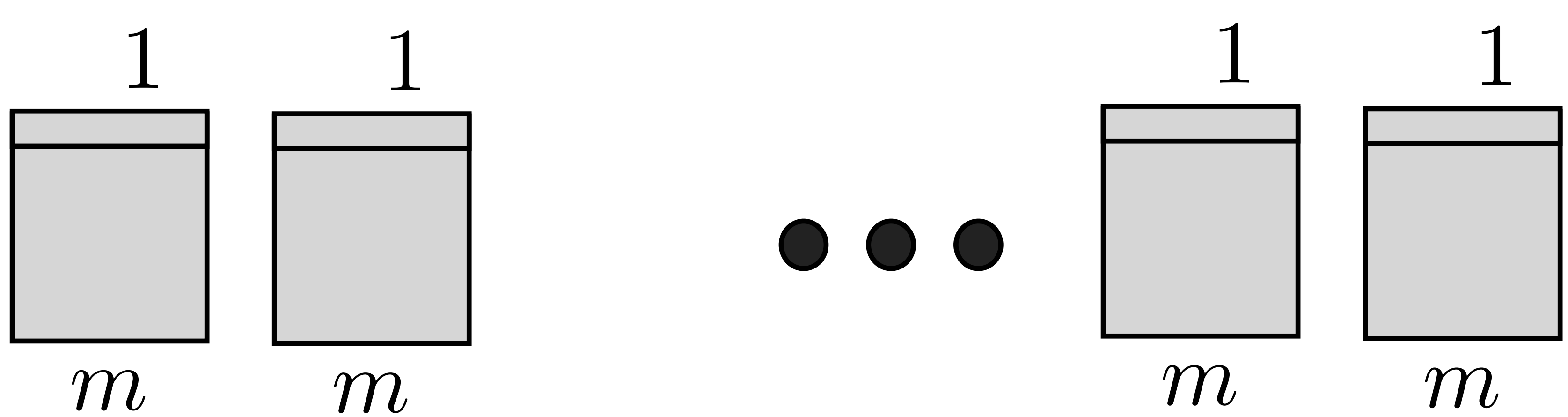
$$\text{cost}(A) = 2m$$



Tightness of bound

m tasks of size m
m tasks of size 1

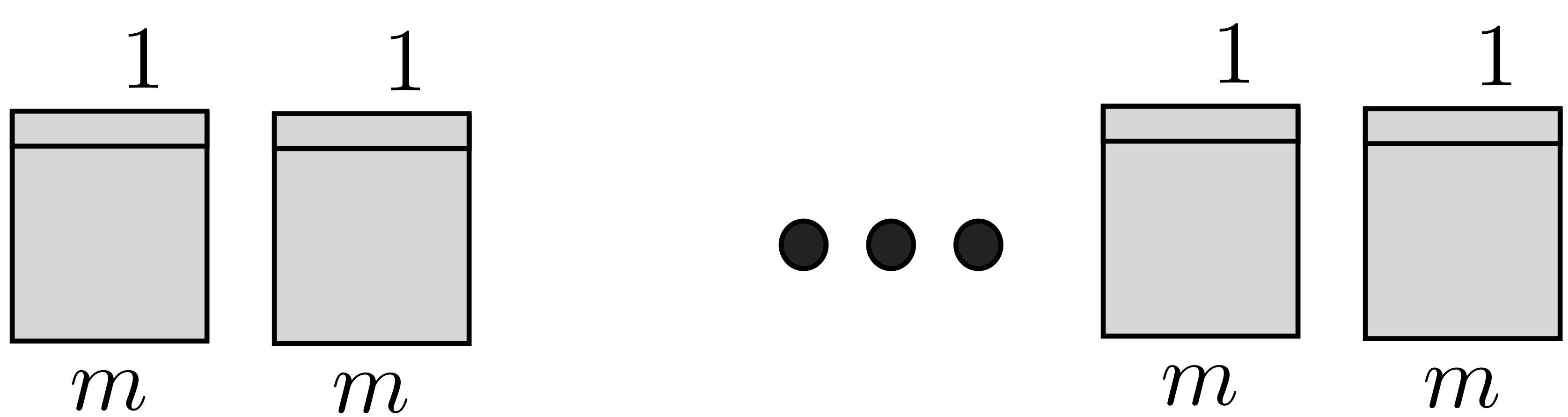
$$\text{opt}(G) = m + 1$$



Tightness of bound

m tasks of size m
m tasks of size 1

$$\frac{\text{cost}(A)}{\text{opt}(G)} = \frac{2m}{m+1} = \left(2 - \frac{2}{m+1}\right)$$



Proof

- Suppose the highest load is given to machine 1.
- This machine has at least 2 assignments.
- Consider the smallest assignment of machine one (assume its assignment 1).

$$l_j \geq l_1 - w_1 \geq \text{cost}(A) - \frac{1}{2}\text{cost}(A) = \frac{1}{2}\text{cost}(A)$$

Proof

$$\begin{aligned}
\mathrm{opt}(G) &\geq \frac{\sum_{i\in[n]} w_i}{m} \\
&= \frac{\sum_{j\in[m]} l_j}{m} \\
&\geq \frac{\mathrm{cost}(A) + \frac{1}{2}\mathrm{cost}(A)(m-1)}{m} \\
&= \frac{(m+1)\mathrm{cost}(A)}{2m}
\end{aligned}$$

$$\mathrm{cost}(A) \leq \frac{2m}{m+1}\mathrm{opt}(G) = \left(2 - \frac{2}{m+1}\right)\mathrm{opt}(G)$$

Identical machines
Moving to a NE.

Changing to a NE.

- Max-weight best response policy: activate an unsatisfied agent with the highest weight.
- An activated agent then chooses a best response.

Theorem

- Every agent gets activated at most once. Hence we get to a NE in linear time.

Proof

- What does being satisfied mean?
- A best response doesn’t decrease the minimum load.

Proof

- Consider an agent that was already activated and now is unsatisfied.

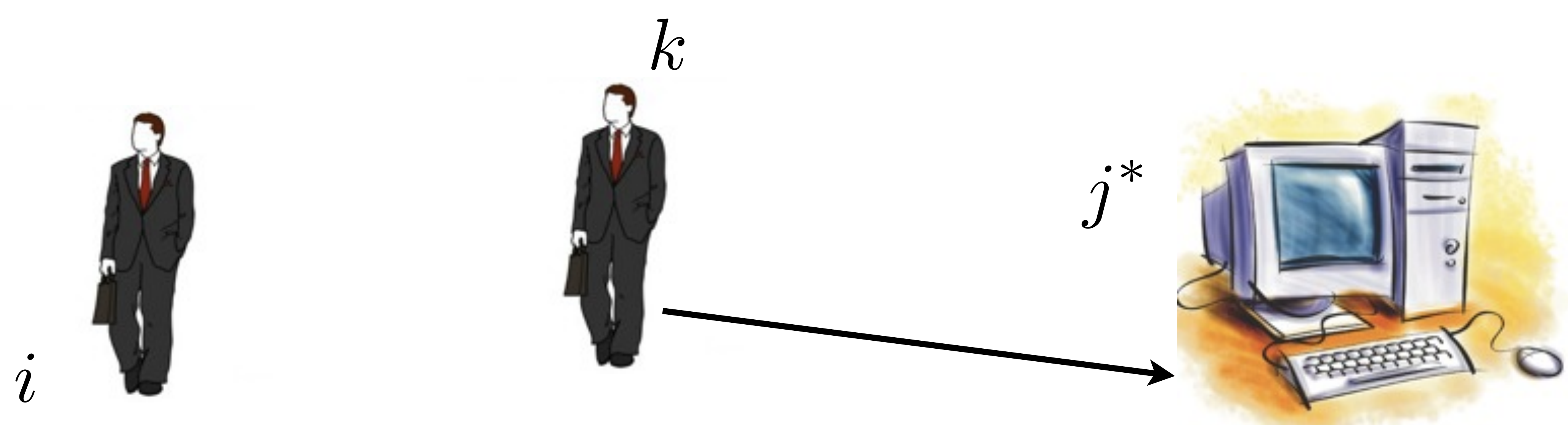
i 

j^* 

$$l_{j*} \leq l_j + w_k \leq l_j + w_i$$

Proof

- Consider an agent that was already activated and now is unsatisfied.



$$l_{j*} \leq l_j + w_k \leq l_j + w_i$$

Uniformly related machines.

Price of Anarchy.

this bound is also tight.

- For a NE the following inequality is satisfied:

$$\mathrm{cost}(A) = \mathcal{O}\left(\frac{\log m}{\log\log m}\right)\mathrm{opt}(G)$$

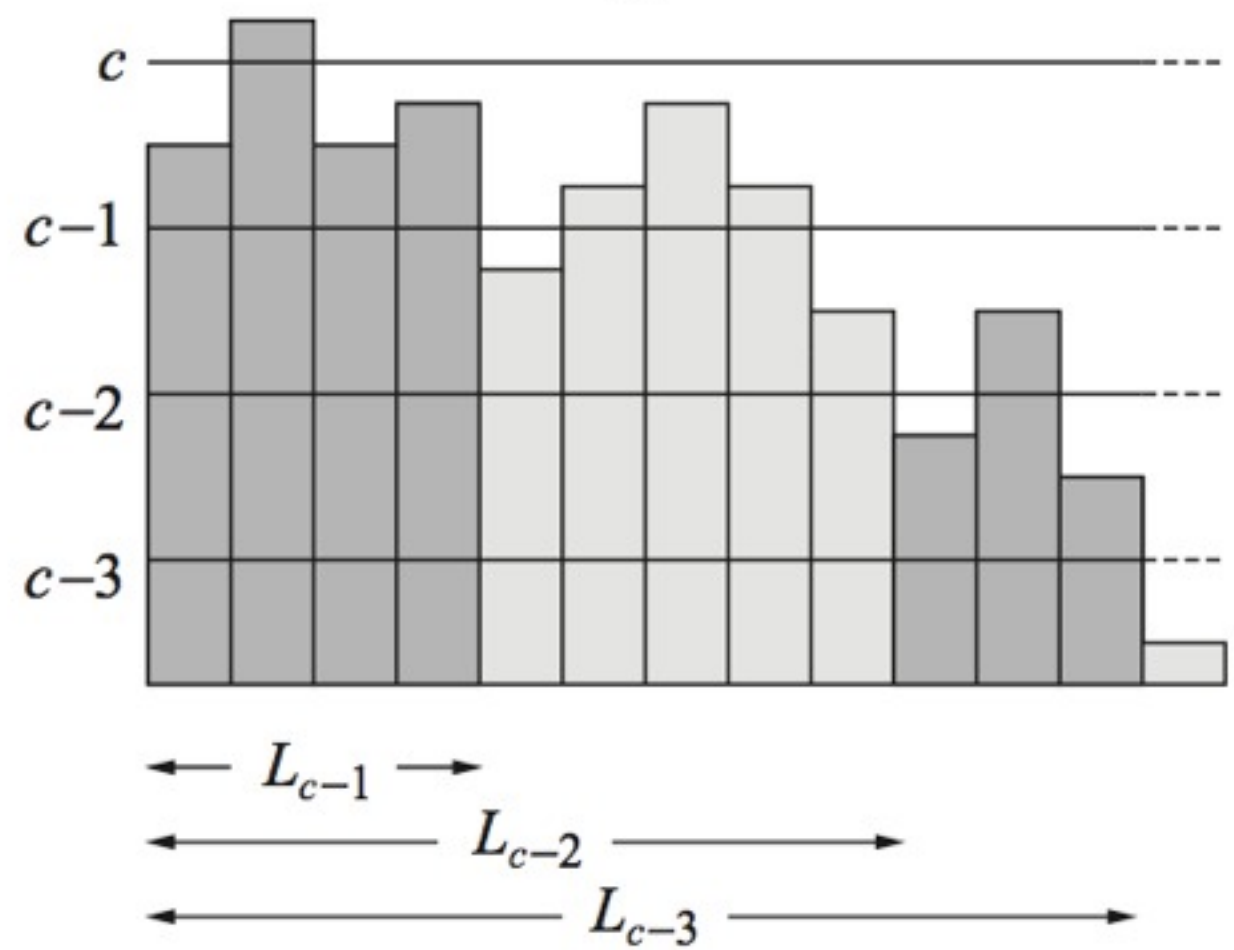
Proof

- We will show that $c = \left\lfloor \frac{\text{cost}(A)}{\text{opt}(G)} \right\rfloor \le \Gamma^{-1}(m)$
- This will prove it because

$$\Gamma^{-1}(m) \sim \frac{\log m}{\log \log m}$$

Proof

- Assume that the machines are labeled such that $s_1 \geq s_2 \geq s_3 ... \geq s_m$
- $L_k = \max\{k : l_i \geq k * \text{opt}(G) \forall i \leq k\}$



Proof

- $L_k = \max\{k : l_i \geq i * \text{opt}(G) \forall i \leq k\} = \{1, 2, ...L_k\}$
- $L_k \geq (k+1)L_{k+1}(0 \leq k \leq c-2)$
- $L_{c-1} \geq 1$
- $m = L = L_0 \geq (c-1)! = \Gamma(c)$ **and** $\Gamma^{-1}(m) \geq c$

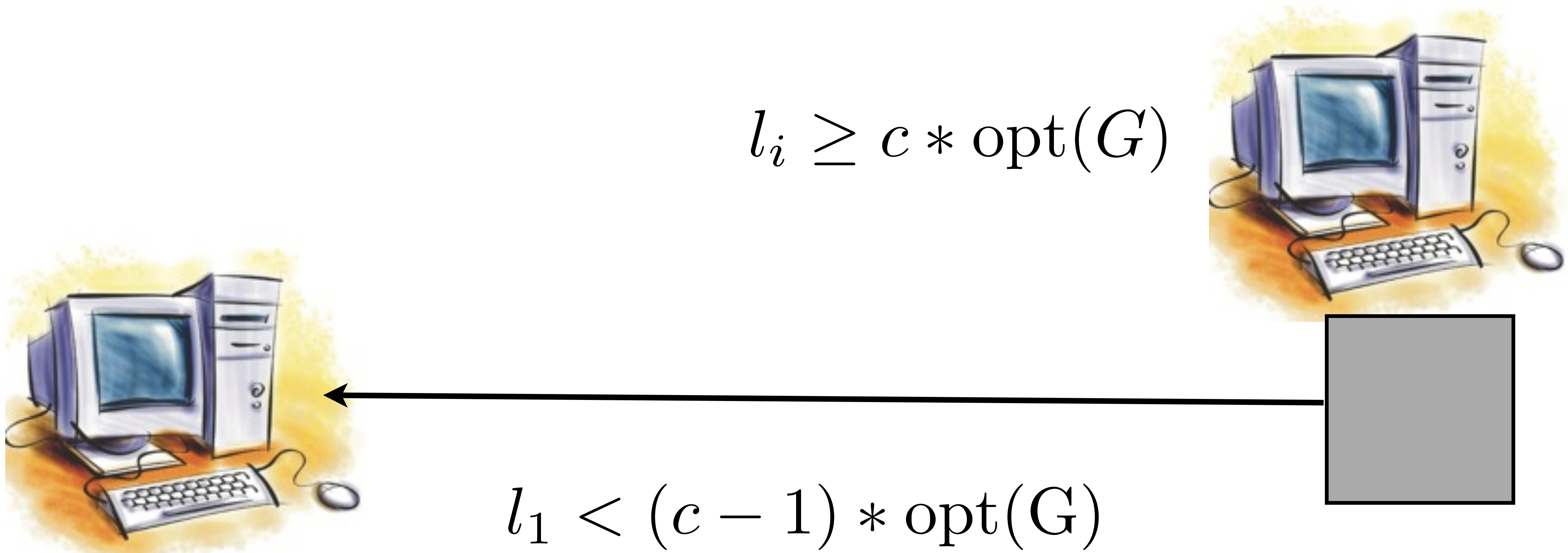
Proof

- First we will see that $L_{c-1} \geq 1$ by means of a contradiction.
-

$$l_1 < (c-1) * \text{opt(G)}$$

Proof

- First we will see that $L_{c-1} \geq 1$ by means of a contradiction.
- $l_1 < (c-1) \cdot \text{opt}(G) + \frac{w_i}{s_1} \leq (c-1) \cdot \text{opt}(G) + \text{opt}(G) \leq c \cdot \text{opt}(G)$,

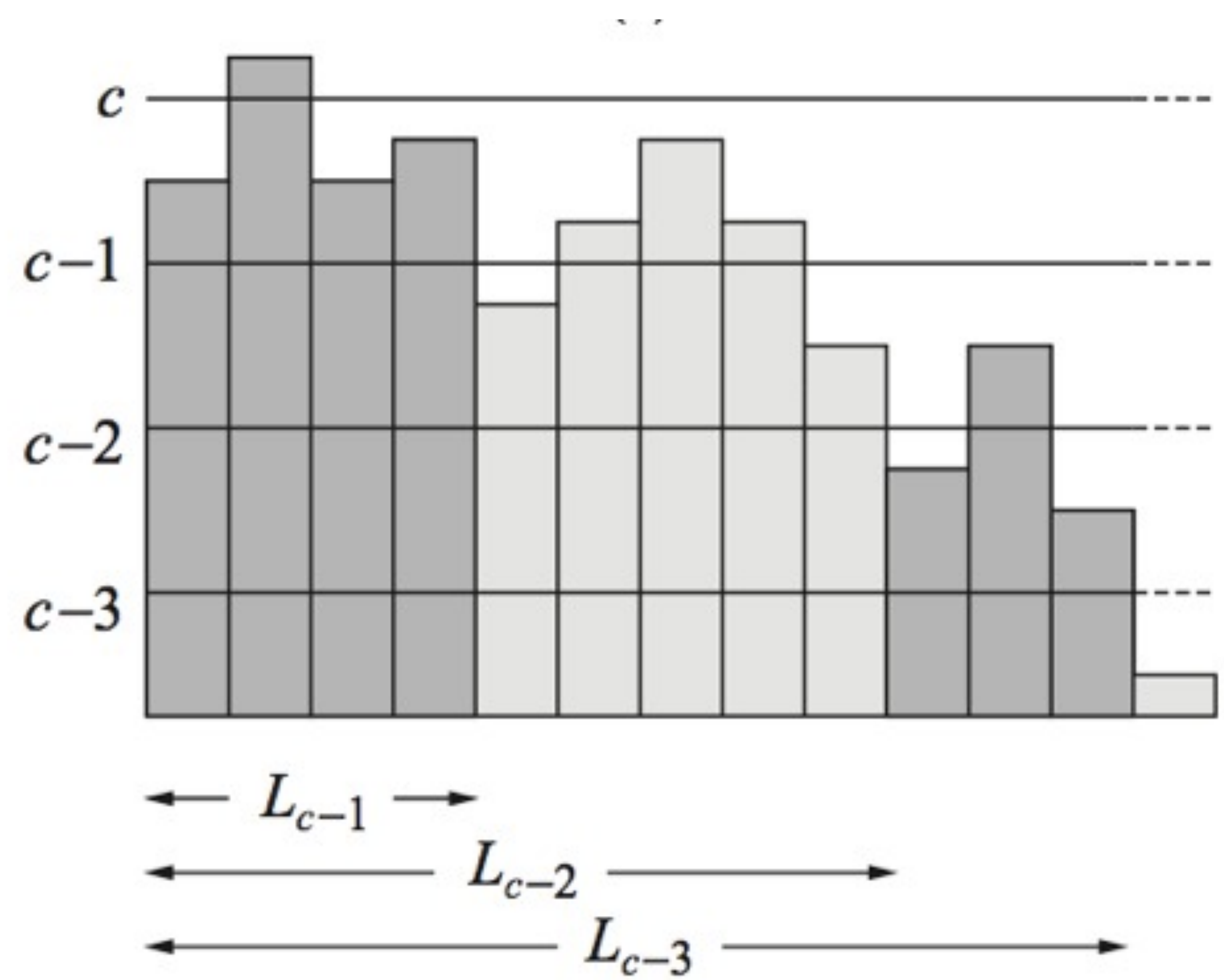


Lemma

- $L_k \geq (k+1)L_{k+1} \ (0 \leq k \leq c_2)$
- Let A^* be an optimum assignment. If $A(i) \in L_{k+1}$ then $A^*(i) \in L_k$

Proof of Lemma

- Let $q = \min \text{index} L - L_k$
-



Proof of Lemma

- Let $q = \min \text{index} L - L_k$
- $A(i) \in L_{k+1} \rightarrow l_{A(i)} \geq (k+1)\text{opt}(G)$
- Claim: $w_i > s_q * \text{opt}(G)$

Proof of Lemma

- Let $q = \min \text{index} L - L_k$
- $A(i) \in L_{k+1} \rightarrow l_{A(i)} \geq (k+1)\text{opt}(G)$
- Claim: $w_i > s_q * \text{opt}(G)$
- By contradiction assume otherwise.
- Move task i to machine q

$$\ell_q + \frac{w_i}{s_q} < k \cdot \text{opt}(G) + \text{opt}(G) \leq \ell_{A(i)},$$

Proof of Lemma

- By contradiction assume $A^*(i) = j$ and $j \in L \setminus L_k$
- Then load of machine j (in assignment A^*) is at least

$$\frac{w_i}{s_j} > \frac{s_q \cdot \mathrm{opt}(G)}{s_j} \geq \mathrm{opt}(G)$$

Lemma

- Let A^* be an optimum assignment. If $A(i) \in L_{k+1}$ then $A^*(i) \in L_k$

Proof

- Recall that $A(i) \in L_{k+1} \rightarrow l_{A(i)} \geq (k+1)\mathrm{opt}(G)$
- An optimum assignment must assign $\sum_{j \in L_{k+1}} (k+1) * \mathrm{opt}(G) * s_j$ to the machines L_k

Proof

- Hence $$\sum_{j \in L_{k+1}} (k+1) \cdot \mathrm{opt}(G) \cdot s_j \leq \sum_{j \in L_k} \mathrm{opt}(G) \cdot s_j .$$
- Dividing by $\mathrm{opt}(G)$ and substracting $\sum_{j \in L_{k+1}} s_j$

$$\sum_{j \in L_{k+1}} k \cdot s_j \leq \sum_{j \in L_k \setminus L_{k+1}} s_j .$$

Proof

- Let s^* be the slowest machine of L_{k+1}, then

for all $j \in L_{k+1}, s_j \geq s^*$ and for all $j \in L_k \setminus L_{k+1}, s_j \leq s^*$

Hence,

$$\sum_{j \in L_{k+1}} k \cdot s_j \leq \sum_{j \in L_k \setminus L_{k+1}} s_j. \longrightarrow \sum_{j \in L_{k+1}} k \cdot s^* \leq \sum_{j \in L_k \setminus L_{k+1}} s^*,$$

Proof

$$\sum_{j \in L_{k+1}} k \cdot s^* \leq \sum_{j \in L_k \setminus L_{k+1}} s^*, \longrightarrow \quad k * L_{k+1} \leq |L_k| - |L_{k+1}|$$

Hence, $(k+1) * L_{k+1} \leq |L_k|$

Proof

$$\sum_{j \in L_{k+1}} k \cdot s^* \leq \sum_{j \in L_k \setminus L_{k+1}} s^*, \longrightarrow \quad k * L_{k+1} \leq |L_k| - |L_{k+1}|$$

Hence, $(k+1) * L_{k+1} \leq |L_k|$

Q.E.D

Computing NE.

- Largest processing time algorithm: inserts task in a nonincreasing order and assigns them in a best response manner.

Theorem

- Largest processing time algorithm finds a NE.

Proof

- Induction on the number of tasks.
- Suppose for t-1 it is good. Add task t.
- What can go wrong?

$$\frac{\sum_{i \in A^{-1}(j^*)} w_i}{s_{j^*}} \leq \frac{\sum_{i \in A^{-1}(j)} w_i + w_t}{s_j} \leq \frac{\sum_{i \in A^{-1}(j)} w_i + w_k}{s_j}$$

Conclusion

Table 20.1. *The price of anarchy for pure and mixed equilibria in load balancing games on identical and uniformly related machines*

	Identical	Uniformly related
Pure	$2 - \frac{2}{m+1}$	$\Theta\left(\frac{\log m}{\log\log m}\right)$
Mixed	$\Theta\left(\frac{\log m}{\log\log m}\right)$	$\Theta\left(\frac{\log m}{\log\log\log m}\right)$

Conclusion

- Reaching a NE in the uniformly related instances.
- More realistic representations.