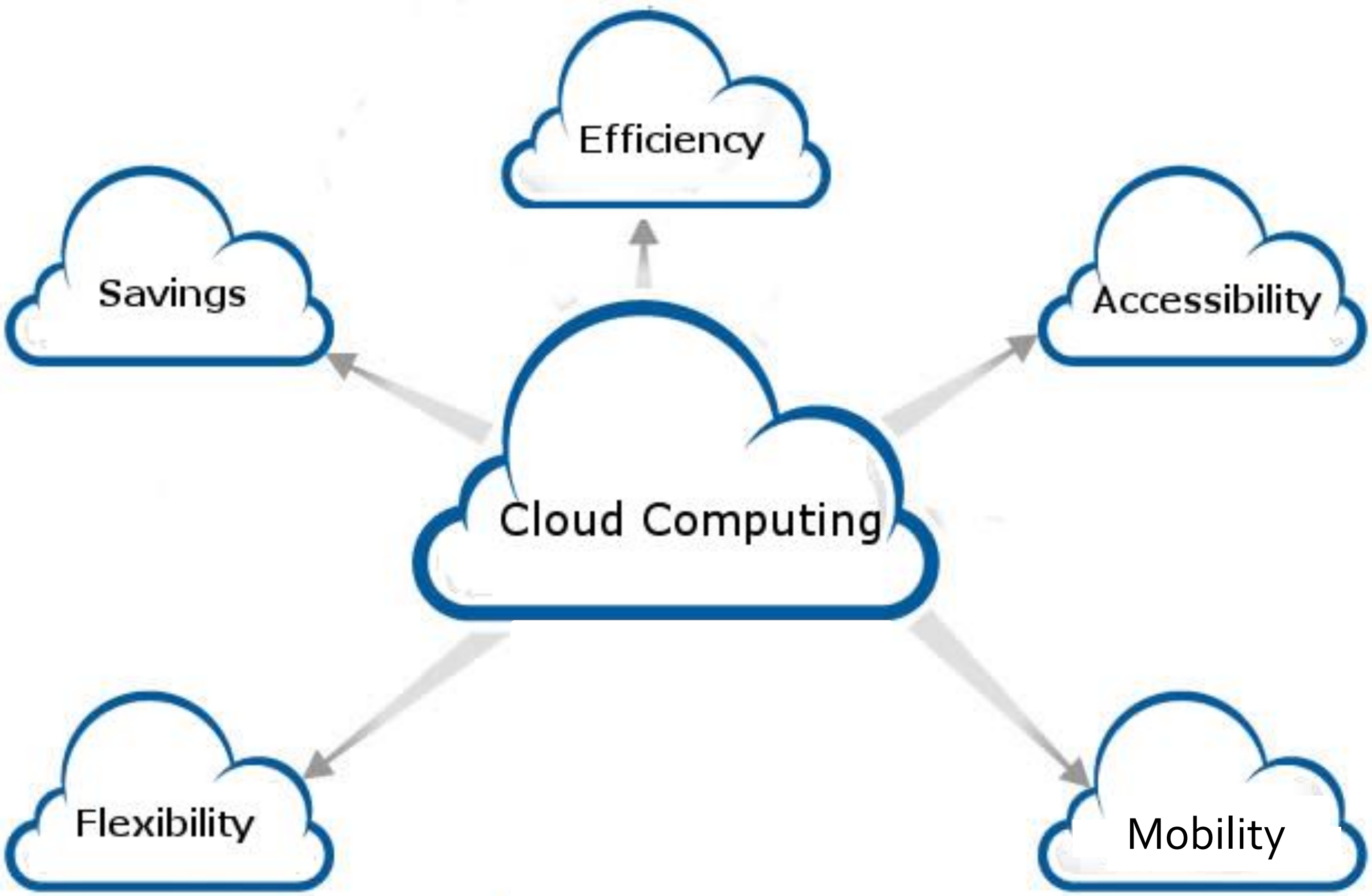
Combinatorial Markets with Covering Constraints: Algorithms and Applications

Ruta Mehta

ILLINOIS
UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN

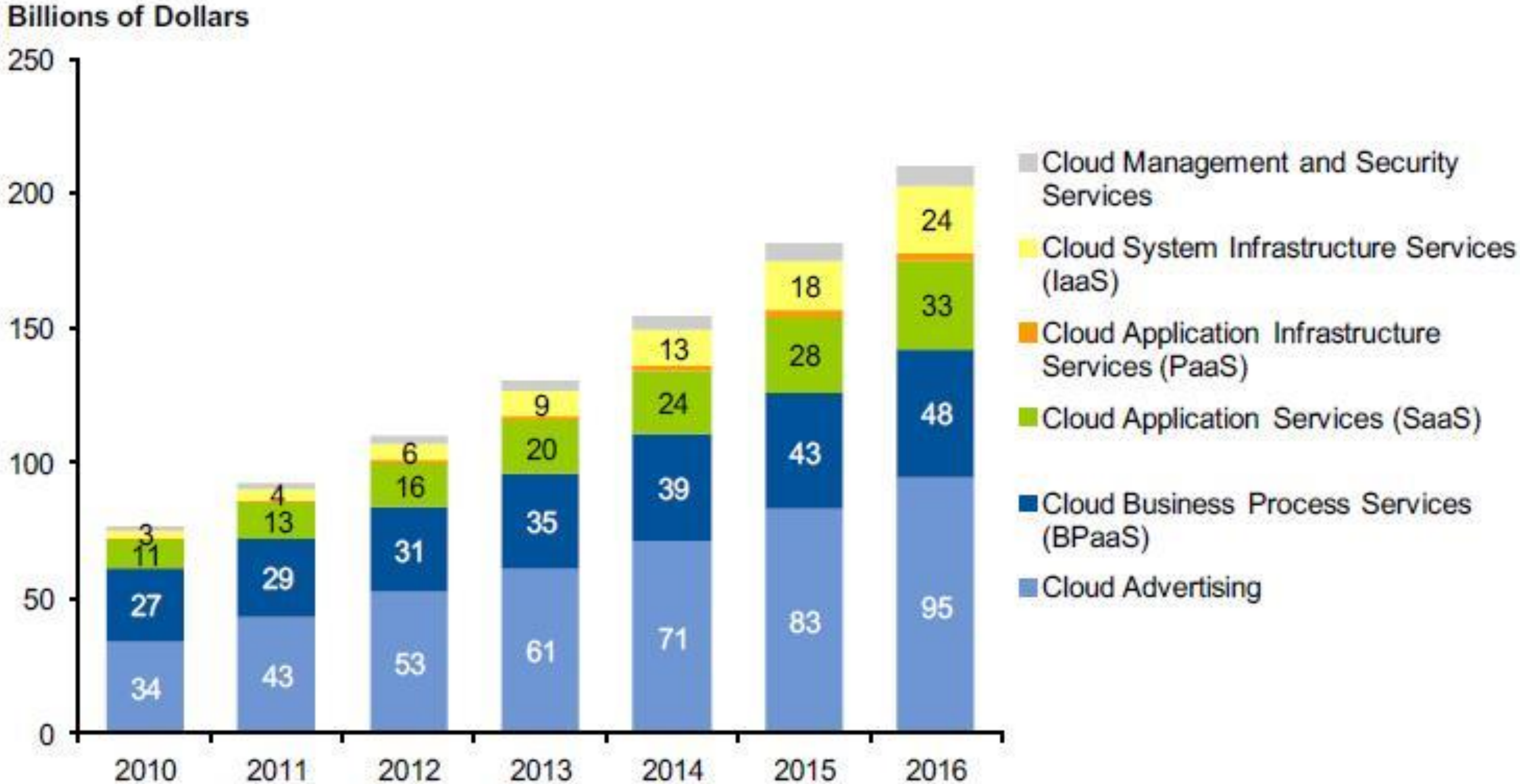
With Nikhil Devanur, Jugal Garg,
Vijay V. Vazirani, and Sadra Yazdanbod

Using servers hosted on the Internet to store, manage, and process data.




Azure Services Platform
BUNGEEconnect
salesforce.com
Microsoft
3tera
Sun microsystems
force.com
YAHOO!, DEVELOPER NETWORK
Google App Engine
ORACLE ON DEMAND
vmware
Google Docs
ACROBAT.COM
Work. Together. Anywhere.
IBM
EMC²
where information lives
Sun Grid Compute Utility
MOSSO
amazon webservices
intel
NETSUITE
Joyent Accelerators
OpenID
hp invent
DELL

Public Cloud Services Market by Segment, 2010-2016



Source: Gartner (February 2013)

Basic Problem

Resources: VM (virtual machine) configurations

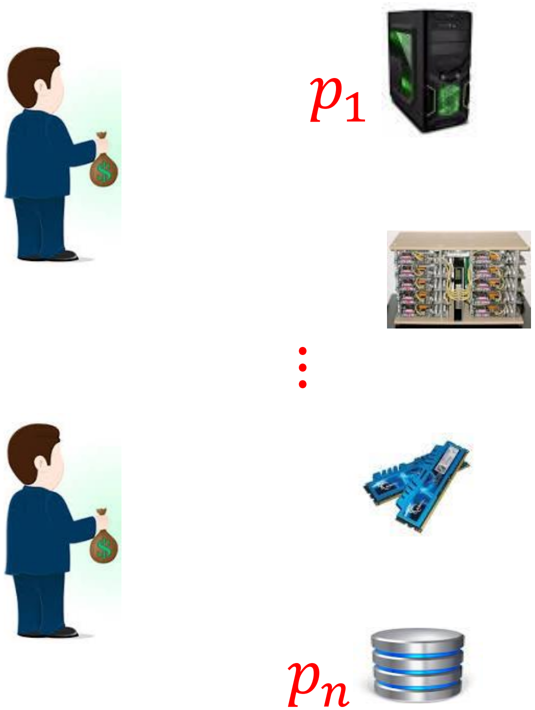
Covering constraints

Each customer wants to run a set of jobs to finish within a budget. Wants to finish ASAP.

Objective

Goal: Set **prices** of resources over time and **allocate**

Generalization



Agent i:

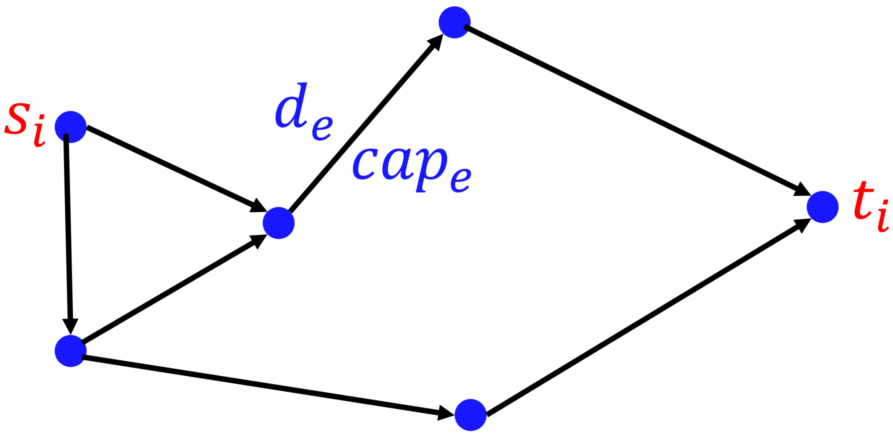
- Has money m_i.
- Covering constraints over goods.

Agent demands bundle s.t.

1. Objective: Min cost
   - Good j costs d_{ij} per unit
2. Within budget & satisfies constraints

Equilibrium: Market clears

More Applications: Network Flow



Equilibrium Existence

Not Always!

Strong Feasibility: Every minimally feasible allocation for a subset of agents extends to all the agents.

Theorem. Equilibrium exists if the market satisfies strong feasibility.

Equilibrium Computation

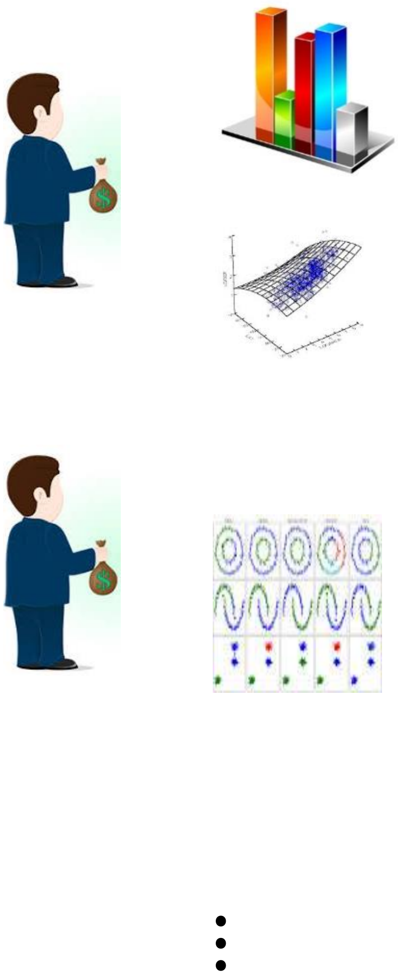
PPAD-hard! [Rubinstein'17]

Extensibility: Every mincost allocation for a subset of agents can be extended to a mincost allocation to all.

Theorem. Equilibrium can be computed in polynomial-time if the market satisfies extensibility.

Includes cloud markets with arrivals and varying capacity of resources, network flow with series-parallel network, etc.

Algorithm: Cloud (scheduling) market

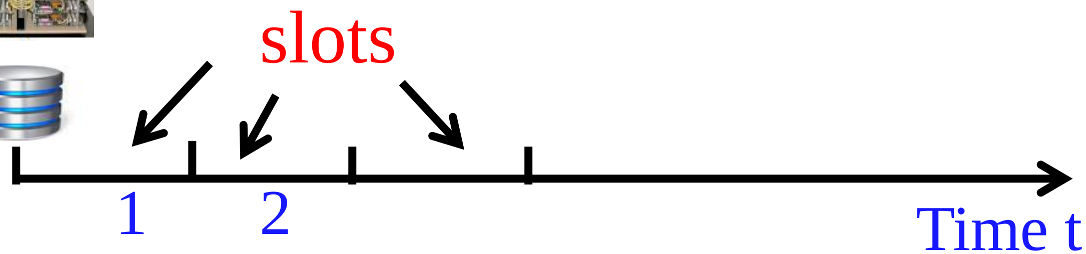


Agent i:

Has money m_i

Wants r_{ij} units of machine j

Price: p_{jt}



Agent demands bundle s.t.

1. Min delay-cost (flow-time)
   - Delay-cost of slot t is t
2. Within budget & finishes jobs

m_1 r_{1j}s

m_2 r_{2j}s

⋮

Price: p_{jt} of good j in time slot t

Agent i demands f_{ijt} s.t.

1. Covering const. : $\sum_t f_{ijt} \geq r_{ij}, \forall j$
2. Within budget $\sum_{j,t} p_{jt} f_{ijt} \leq m_i$
3. Min delay-cost : min: $\sum_{j,t} t f_{ijt}$

Optimal bundle LP

Equilibrium: Market clears

$\forall (j,t),$ Aggregate demand $\leq 1.$

If less than 1, then $p_{jt} = 0$

Equilibrium Characterization

p_{jt}: Price of good j in slot t
f_{ijt}: Agent $i's$ demand
min: $\sum_{j,t} t f_{ijt}$

For each good j

1. Price is decreasing over time
   - If $p_{jt} < p_{j(t+1)}$ then no one buys good j in slot (t+1)

2. Difference in price is decreasing

Equilibrium Characterization

p_{jt}: Price of good j in slot t

f_{ijt}: Agent $i's$ demand

min: $\sum_{j,t} t f_{ijt}$

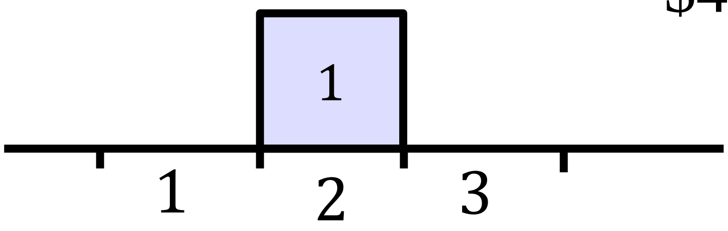
For each good j

1. Price is decreasing over time
   - If $p_{jt} < p_{j(t+1)}$ then no one buys good j in slot (t+1)

2. Difference in price is decreasing

Cost: 2

$4



p_1	p_2	p_3
6	4	0

2 < 4

Equilibrium Characterization

p_{jt}: Price of good j in slot t

f_{ijt}: Agent $i's$ demand

min: $\sum_{j,t} t f_{ijt}$

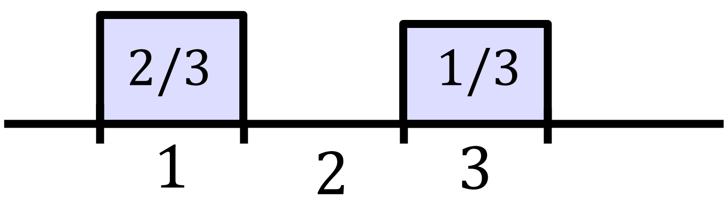
For each good j

1. Price is decreasing over time
   - If $p_{jt} < p_{j(t+1)}$ then no one buys good j in (t+1)

2. Difference in price is decreasing

Cost: $\frac{2}{3}+1 < 2$

$4



p_1	p_2	p_3
6	4	0

2 < 4

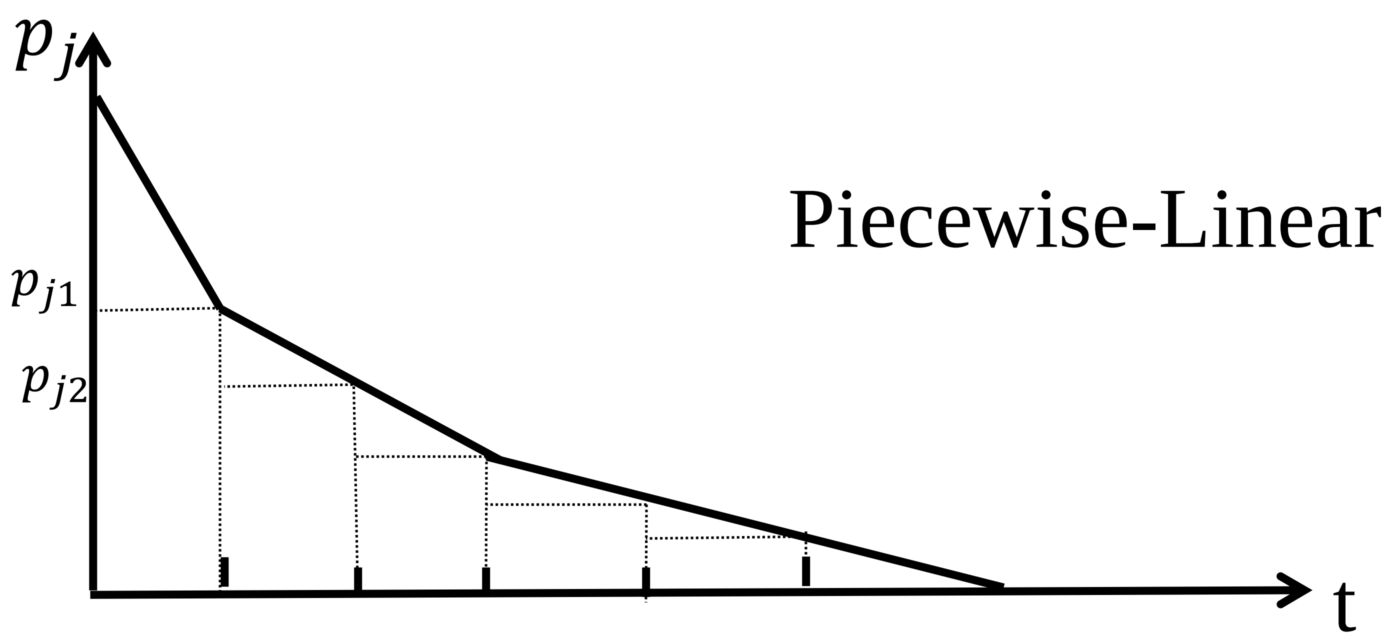
Equilibrium Characterization

p_{jt}: Price of good j in slot t
f_{ijt}: Agent $i's$ demand
min: $\sum_{j,t} t f_{ijt}$

For each good j

1. Price is decreasing over time

2. Difference in price is decreasing



Piecewise-Linear Convex

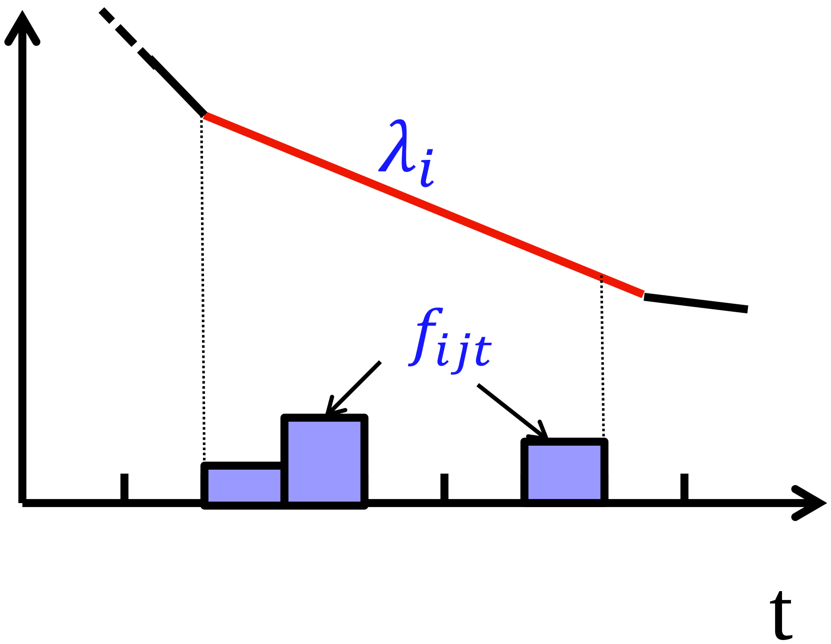
Equilibrium Characterization

For each agent i: (1) delay cost, (2) monetary cost.

Optimal bundle LP $\xrightarrow{\text{KKT}}$ $t\lambda_i + p_{jt}$ **perceived cost**

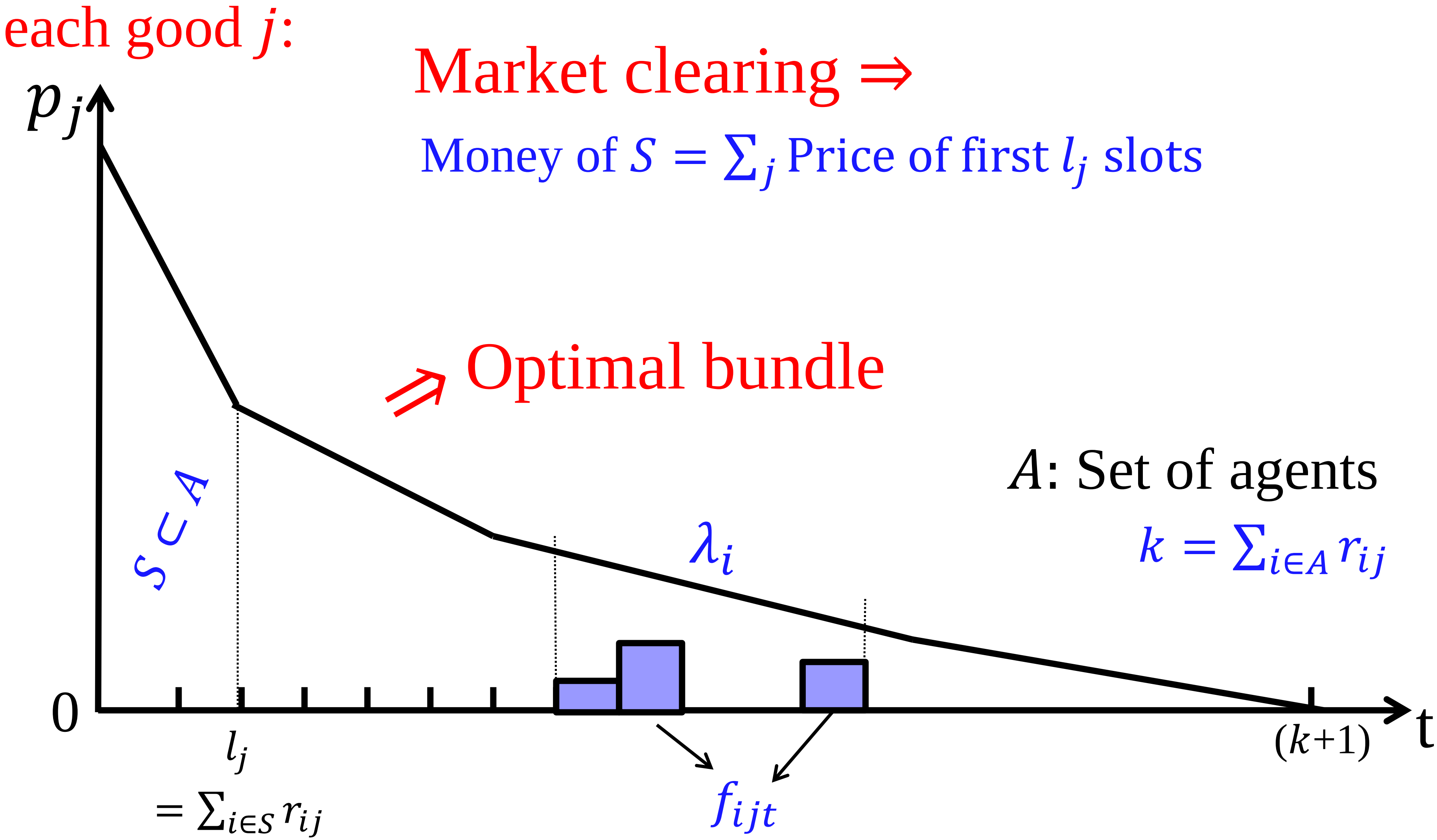
For each good: Buy only where perceived cost is minimum.

Lemma:

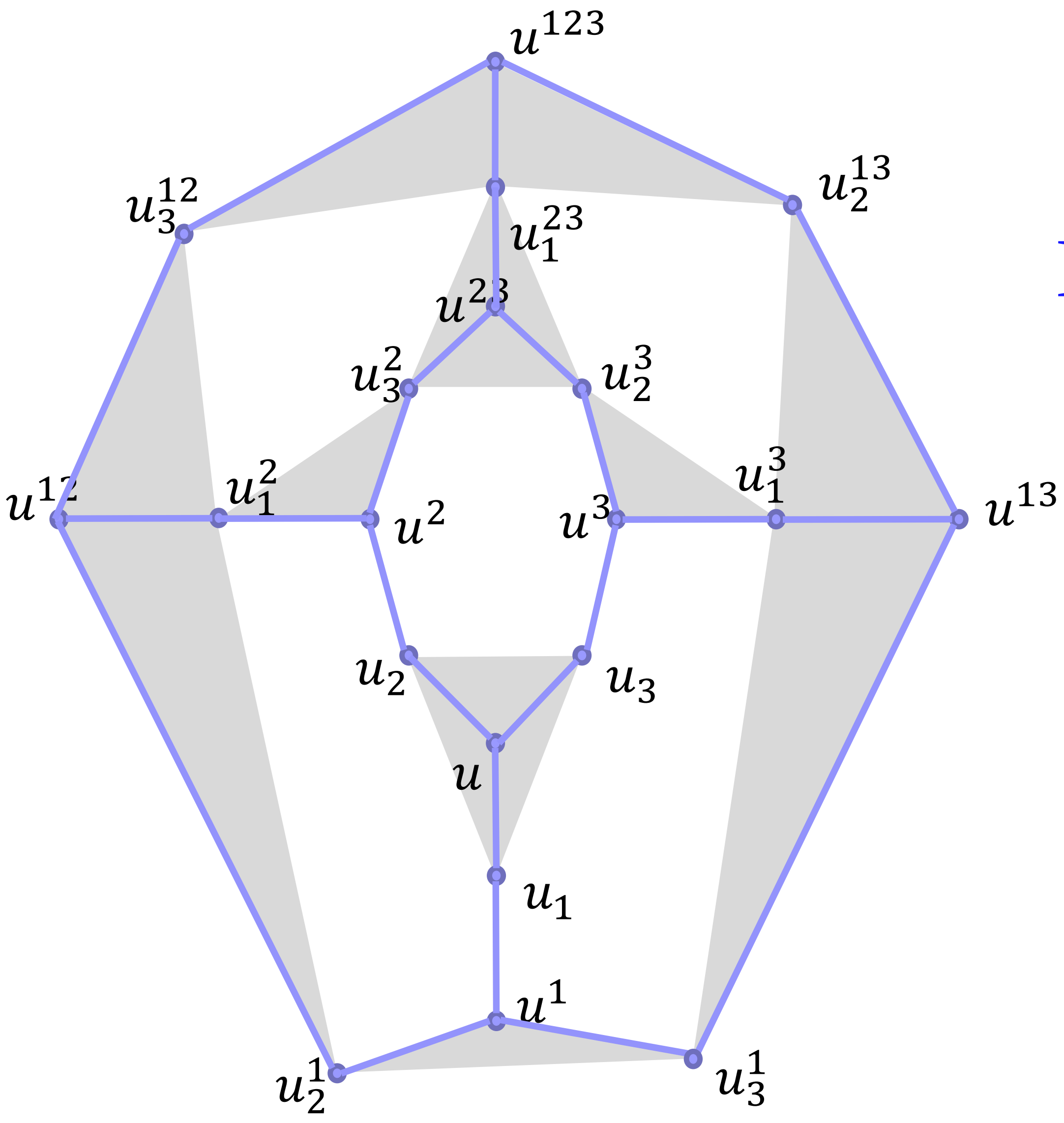


Note: λ_i common across goods.

Equilibrium Characterization



Non-Convex Equilibria

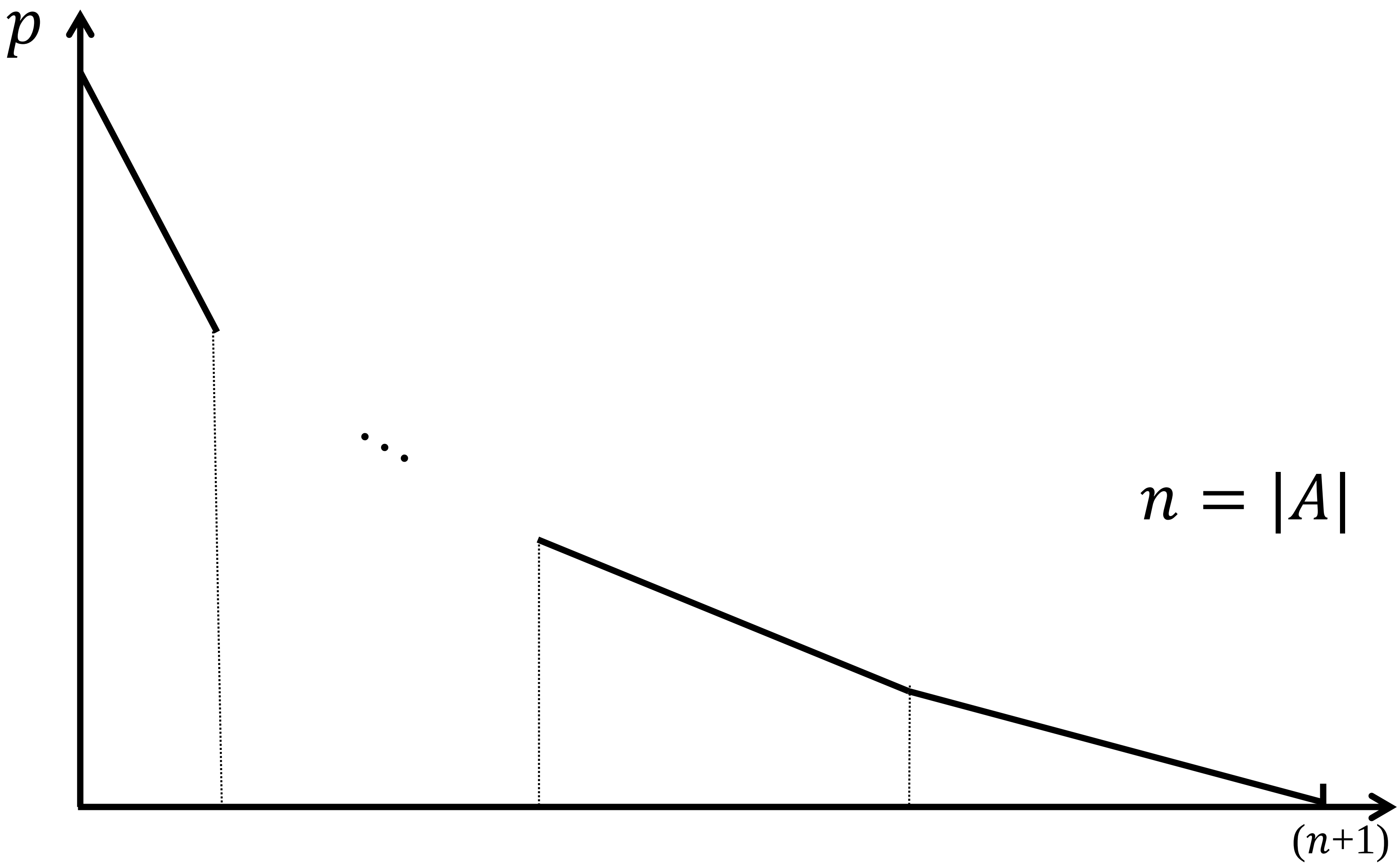


Market with 6 agents and 1 good

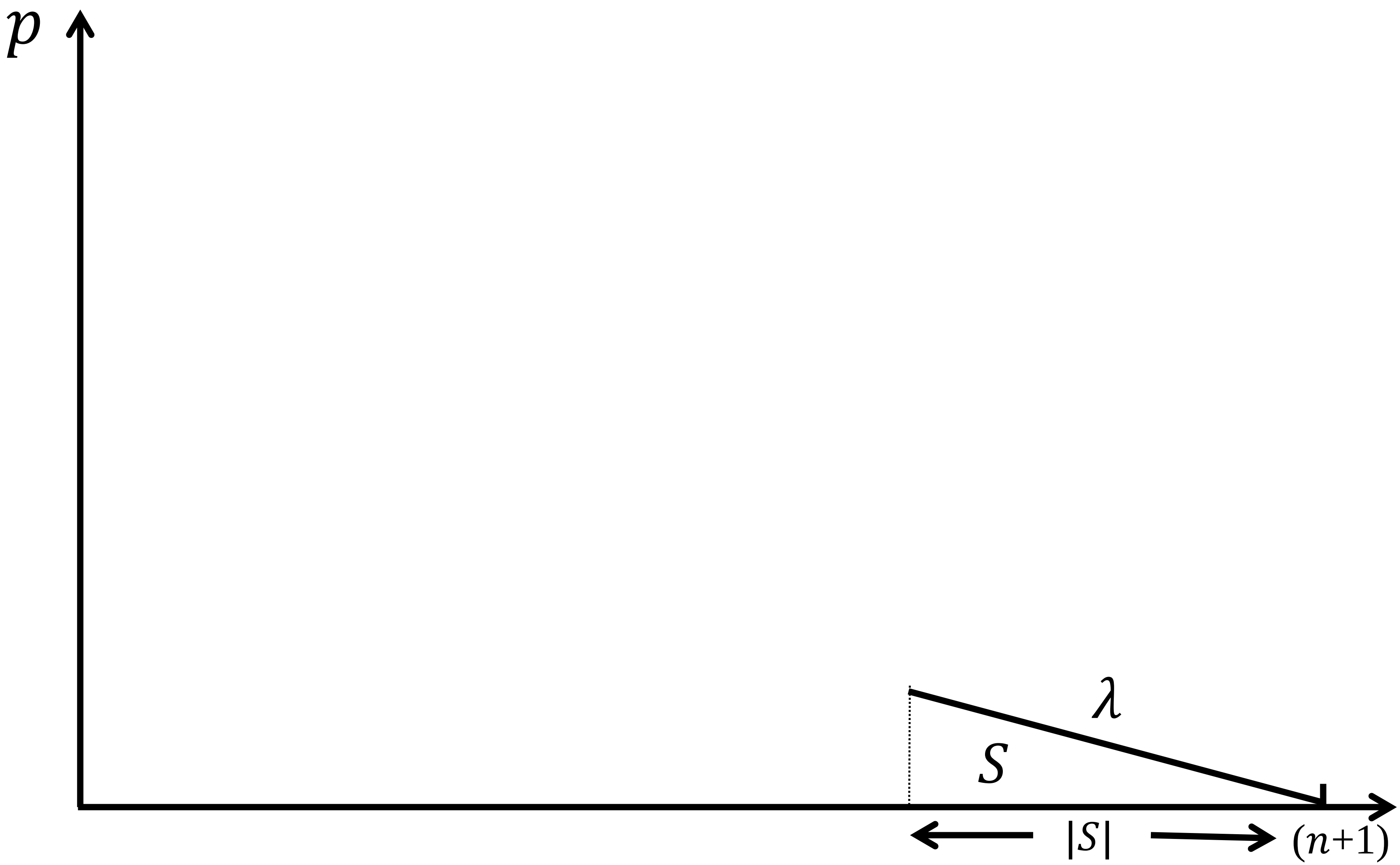
Algorithm: Single good case

r_i=1 for each agent i

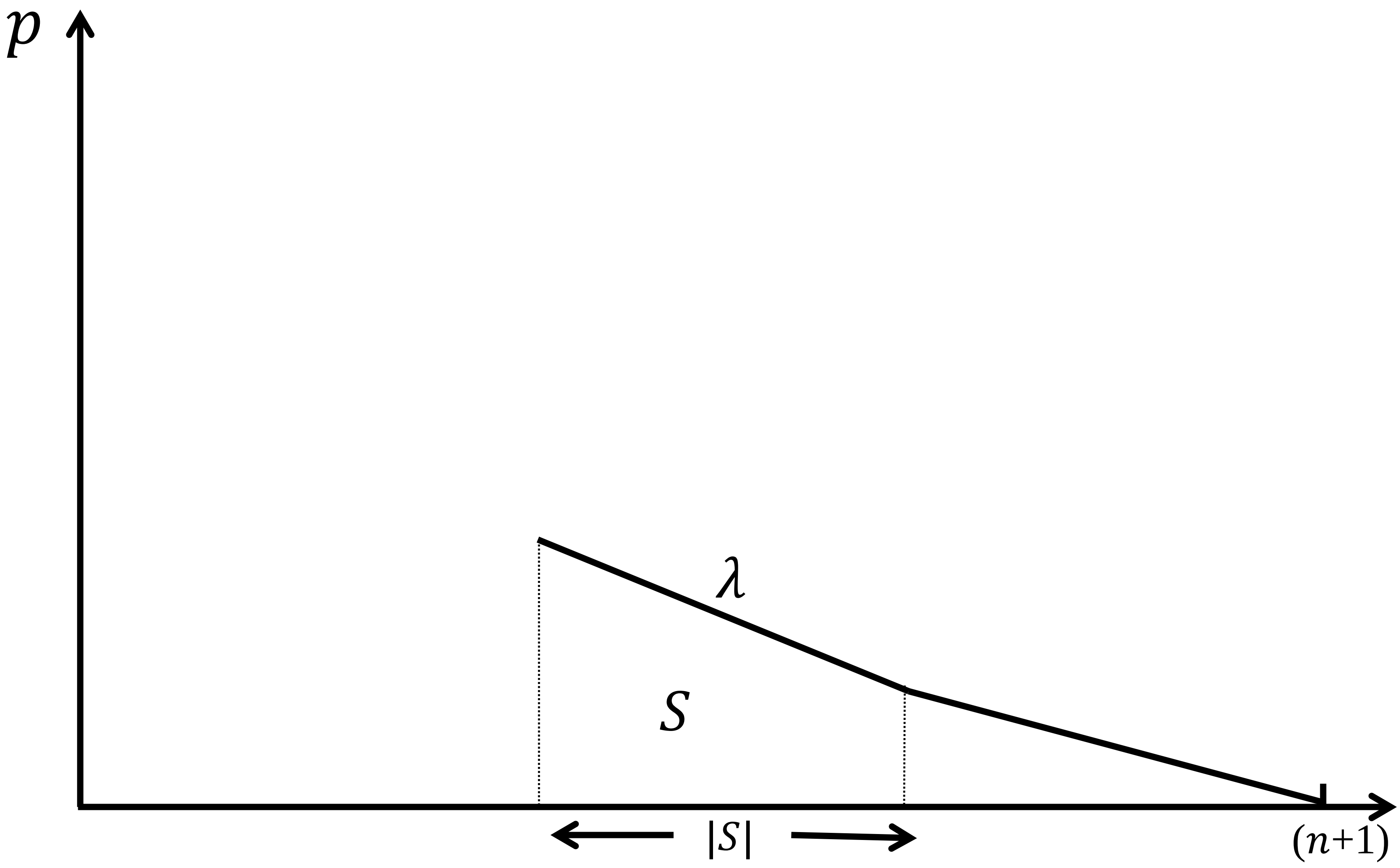
Algorithm



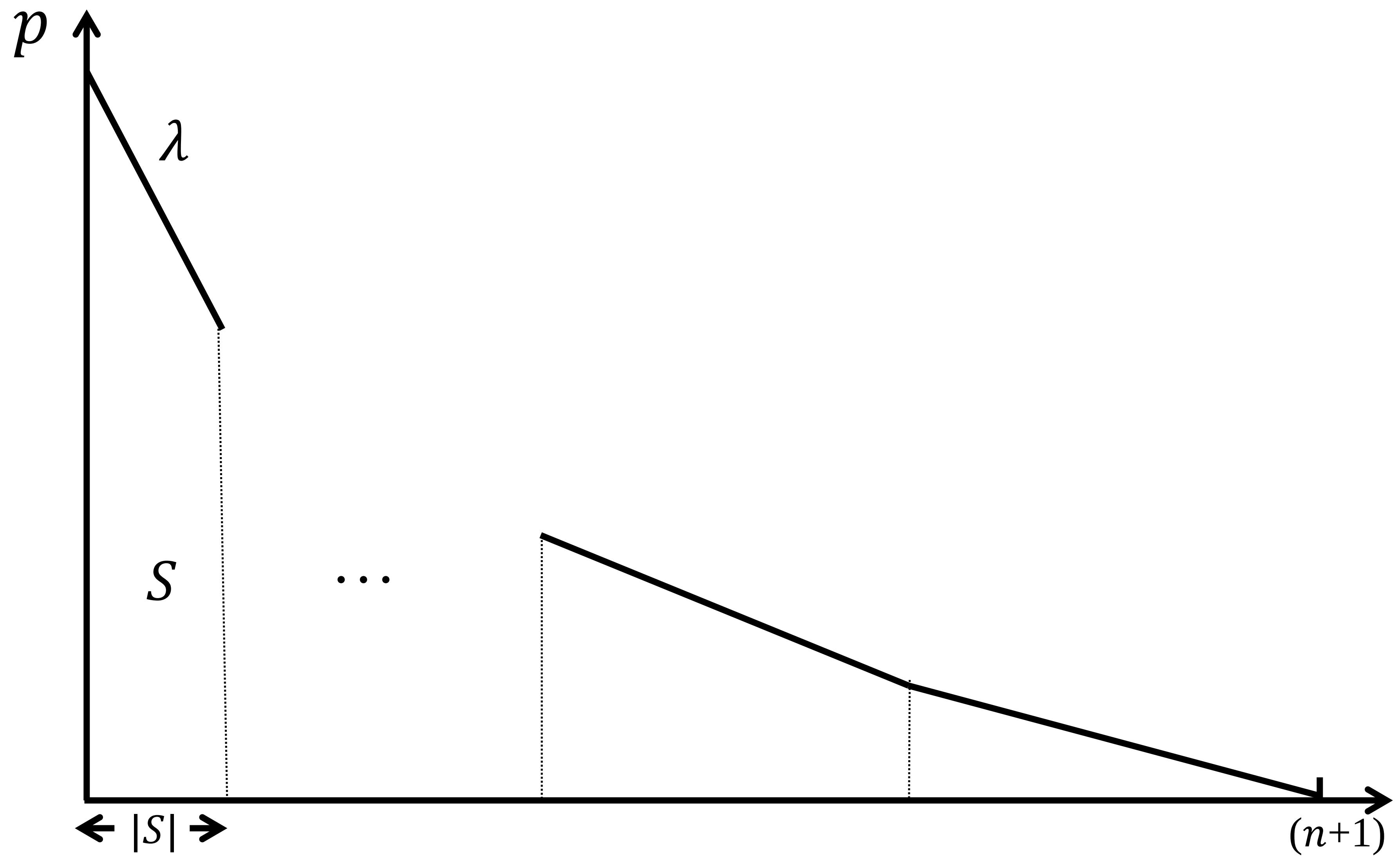
Algorithm



Algorithm

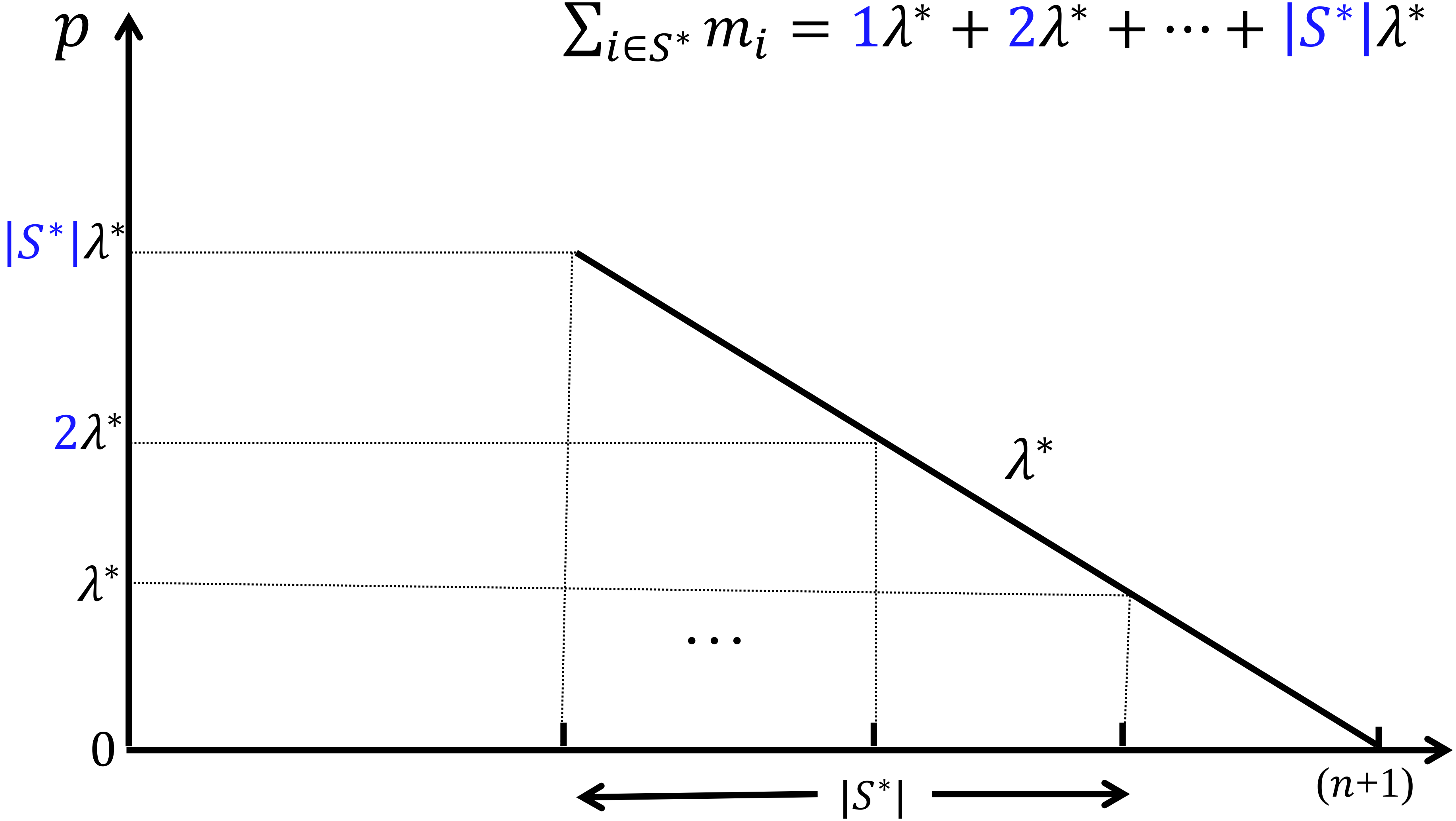


Algorithm



Algorithm: Last segment

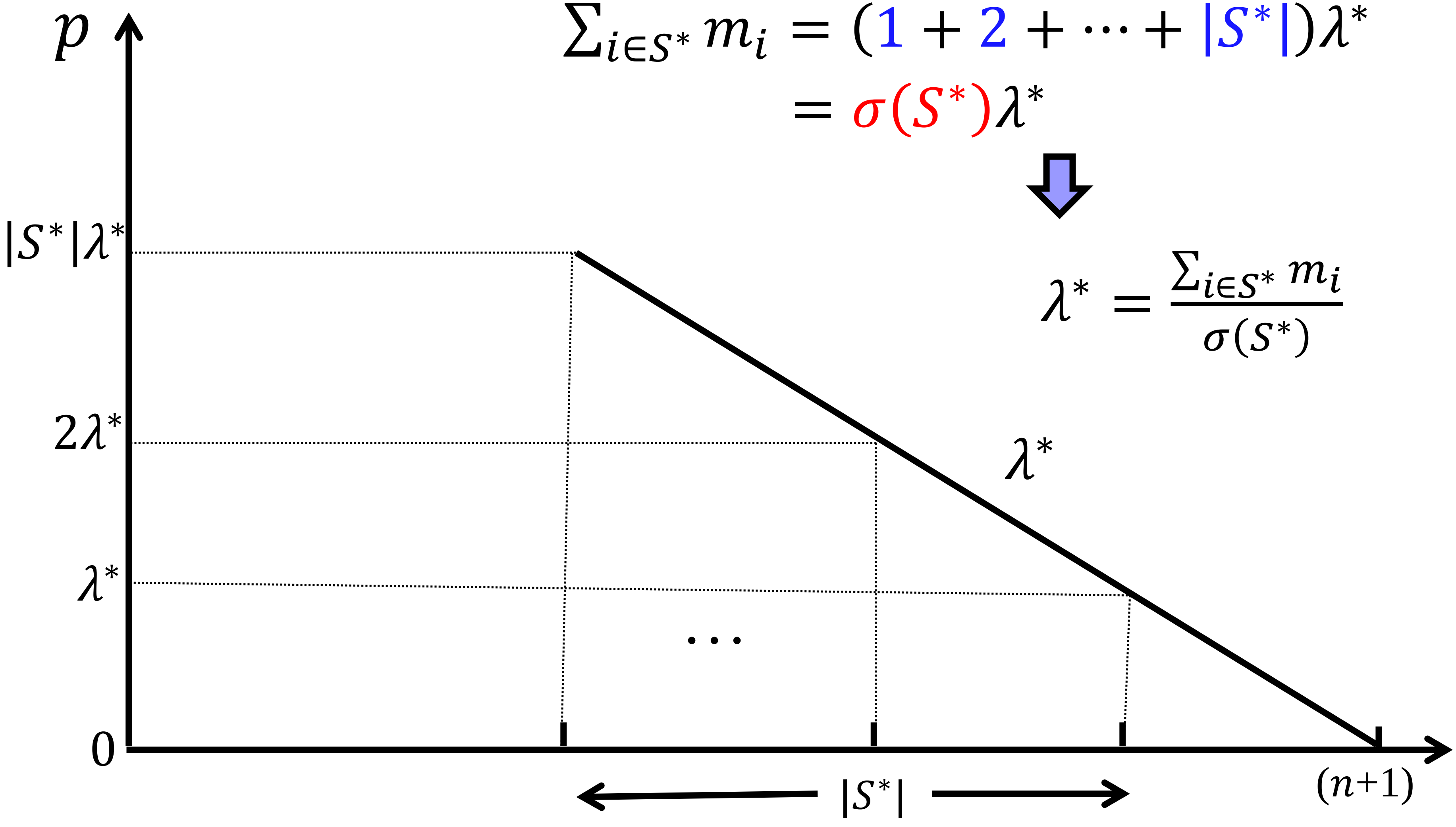
$$\sum_{i \in S^*} m_i = 1\lambda^* + 2\lambda^* + \cdots + |S^*|\lambda^*$$



Algorithm: Last segment

$$\sum_{i \in S^*} m_i = (1 + 2 + \cdots + |S^*|)\lambda^*$$
$$= \sigma(S^*)\lambda^*$$

$$\lambda^* = \frac{\sum_{i \in S^*} m_i}{\sigma(S^*)}$$



Last segment: Find λ^*, S^*

$$\lambda_{S^*} = \lambda^*$$

$$\lambda_S \stackrel{\text{def}}{=} \frac{\sum_{i \in S} m_i}{\sigma(S)}$$

Want: $\forall S, \quad \lambda_{S^*} \leq \lambda_S$

$$\lambda^* = \frac{\sum_{i \in S^*} m_i}{\sigma(S^*)}$$

$$S^* = \operatorname*{argmin}_{S \subseteq A} \lambda_S$$

Exponentially many sets!

Last segment: Find λ^*, S^*

$$\forall S, \quad \lambda_S \left(= \frac{\sum_{i \in S} m_i}{\sigma(S)} \right) \geq \lambda^* = \lambda_{S^*}$$

$$\forall S, \ \sum_{i \in S} m_i - \lambda^* \sigma(S) \geq 0$$
$$= 0 \ \text{ if } \ S = S^*$$

Given λ^*

$$S^* = \text{argmin}_{S \subseteq A} \sum_{i \in S} m_i - \lambda^* \sigma(S)$$

Polytime!

Sub-modular function

How to find λ^*?

Binary search!

Last segment: Find λ^*, S^*

$$\forall S, \qquad \lambda_S \left(= \frac{\sum_{i \in S} m_i}{\sigma(S)} \right) \geq \lambda^* = \lambda_{S^*}$$

$$\forall S,\ \sum_{i \in S} m_i - \lambda^* \sigma(S) \geq 0$$
$$= 0 \ \text{ if } \ S = S^*$$

$$S^* = \arg\min_{S \subseteq A} \sum_{i \in S} m_i - \lambda^* \sigma(S) \stackrel{\text{def}}{=} g(\lambda^*)$$

How to find λ^*?

Binary search!

Last segment: Find λ^*, S^*

$$\forall S, \qquad \lambda_S \left(= \frac{\sum_{i \in S} m_i}{\sigma(S)} \right) \geq \lambda^* = \lambda_{S^*}$$

$$\forall S,\ \sum_{i \in S} m_i - \lambda^* \sigma(S) \geq 0$$

How to find λ^*?

$g(\lambda^*) = 0$

Binary search!

Sub-modular minimization

Last segment: Find λ^*, S^*

$$\forall S, \quad \lambda_S \left(= \frac{\sum_{i \in S} m_i}{\sigma(S)} \right) \geq \lambda^* = \lambda_{S^*} > \lambda$$

$$\forall S, \ \sum_{i \in S} m_i - \lambda \sigma(S) > 0$$ for $\lambda < \lambda^*$

$$g(\lambda) > 0$$

How to find λ^*?

$$g(\lambda^*) = 0$$

Binary search!

Sub-modular minimization

Last segment: Find λ^*, S^*

$$\forall S, \quad \lambda_S \left(= \frac{\sum_{i \in S} m_i}{\sigma(S)} \right) \geq \lambda^* = \lambda_{S^*}$$

$$\sum_{i \in S^*} m_i - \lambda \sigma(S^*) < 0 \qquad \text{for } \lambda > \lambda^* = \frac{\sum_{i \in S^*} m_i}{\sigma(S^*)}$$

$$g(\lambda) < 0$$

for $\lambda > \lambda^*$ $\quad g(\lambda) < 0$

for $\lambda < \lambda^*$ $\quad g(\lambda) > 0$

$$g(\lambda^*) = 0$$

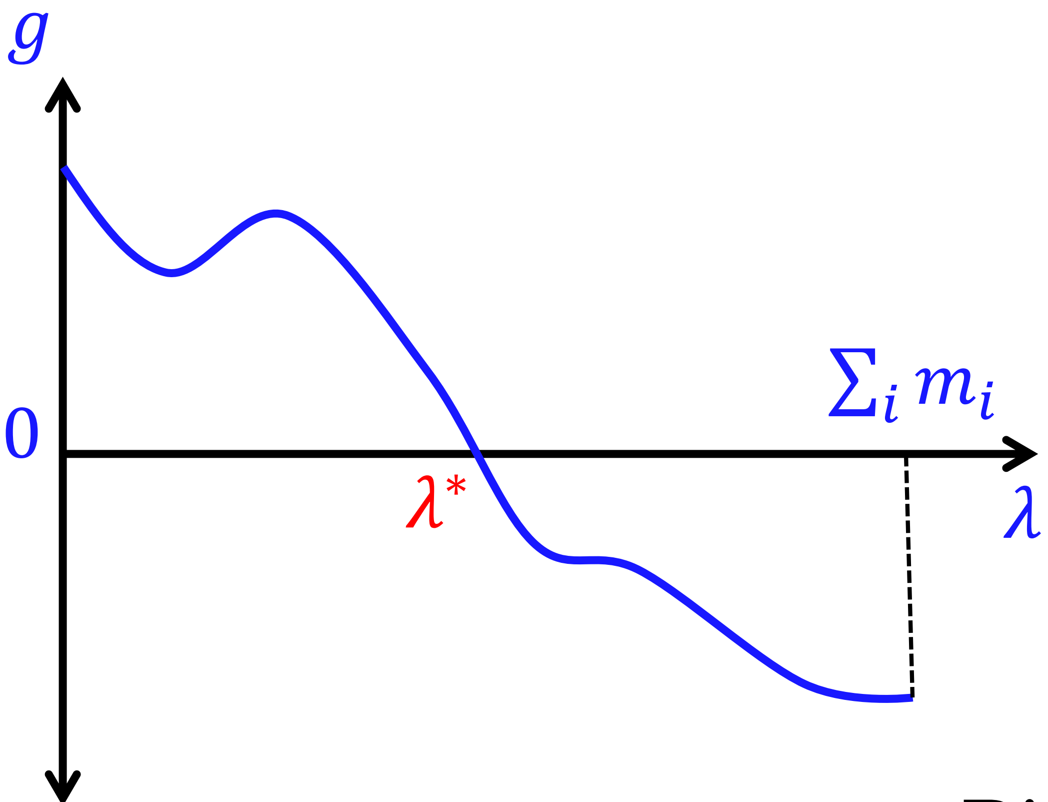
How to find λ^*?

Binary search!

Sub-modular minimization

Last segment: Find λ^*, S^*

$$\forall S, \qquad \lambda_S \left(= \frac{\sum_{i \in S} m_i}{\sigma(S)} \right) \geq \lambda^* = \lambda_{S^*}$$



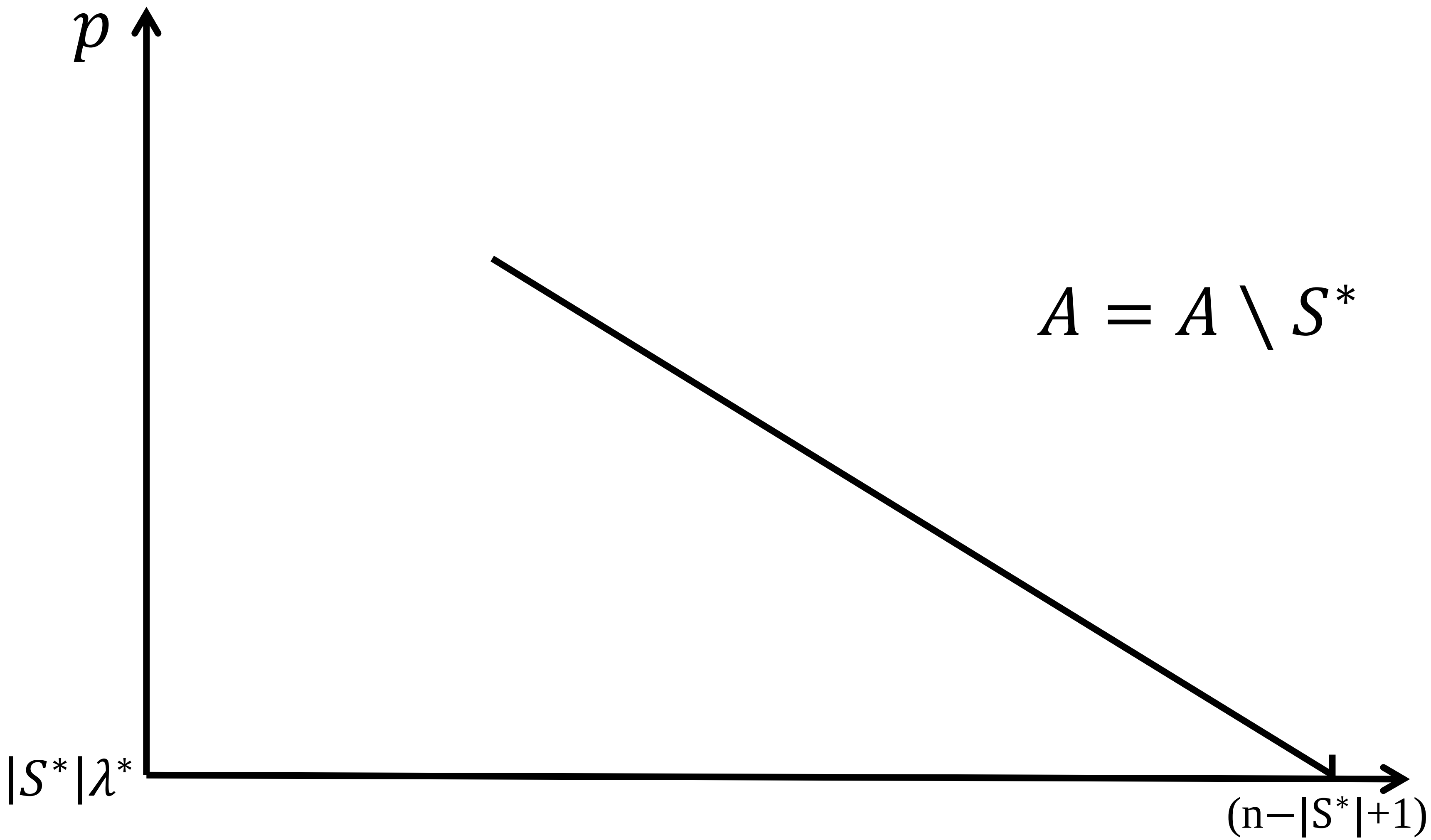
for $\lambda > \lambda^*$ $\quad g(\lambda) < 0$

for $\lambda < \lambda^*$ $\quad g(\lambda) > 0$

$g(\lambda^*) = 0$

Binary search!

Algorithm: **Second** last segment

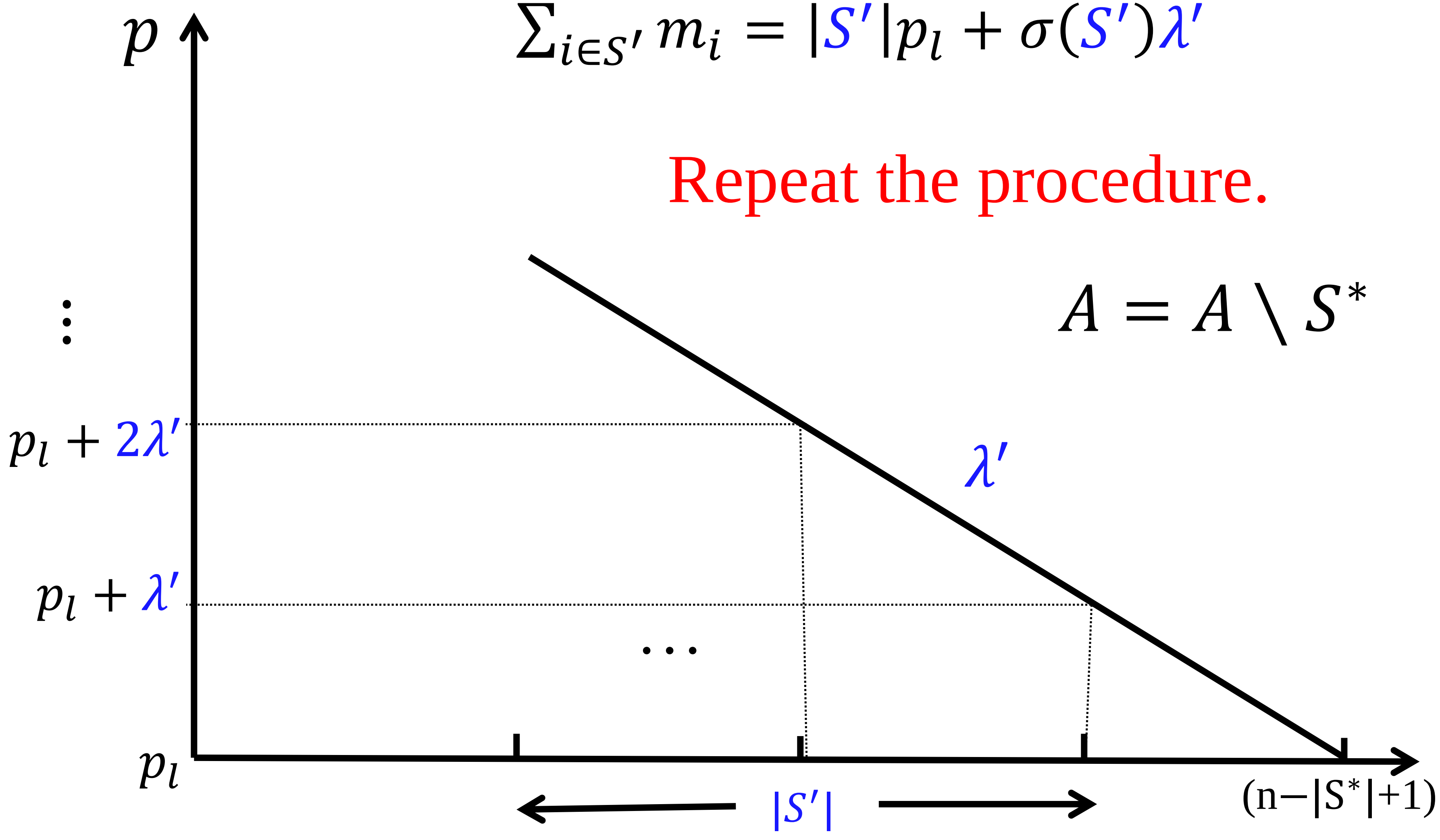


Algorithm: **Second** last segment

$\sum_{i \in S'} m_i = |S'| p_l + \sigma(S') \lambda'$

Repeat the procedure.

$A = A \setminus S^*$



Algorithm

$p_l = 0, A' = A$
While $A' \neq \emptyset$
    $[S, \lambda]$ =Find-Last-Segment(p_l, A')
    Store (S, λ)
    $p_l = p_l + |S|\lambda; \quad A' = A' \setminus S$

General markets: Parameterized LP with $(\lambda_1, \ldots, \lambda_n)$ as parameters + submodular optimization

Major Challenges: Monotonic prices, hold payments of allocated agents, existence of final allocation.

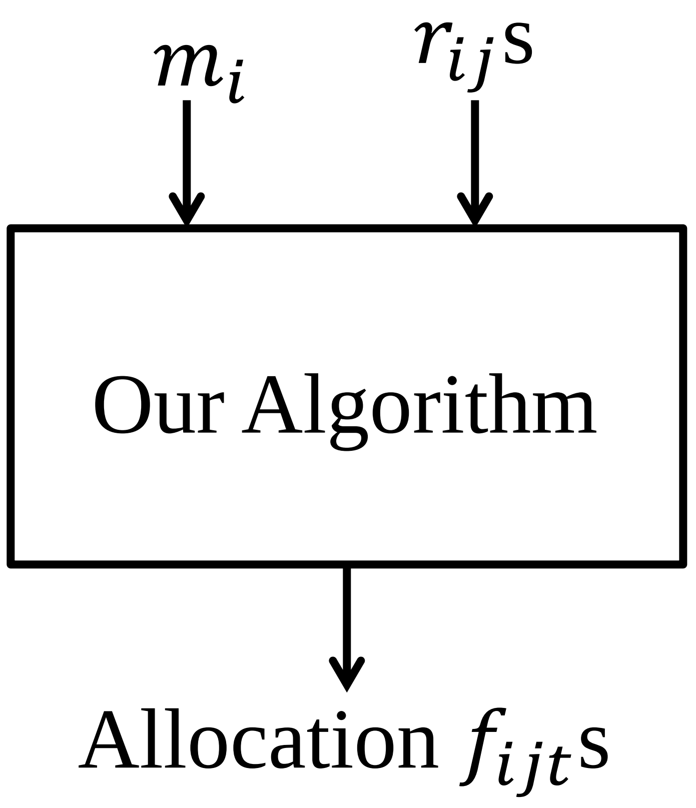
Extensions (Scheduling market)

- Multiple goods and arbitrary requirements
- Arbitrary monotonic delay cost
- Arbitrary arrivals of agents
- Arbitrary capacity of resources across time
- Etc.

Fairness Properties

- Pareto-optimal allocation.
- Envy-free
- Every buyer gets at least her "fair share"
  - The allocation Pareto dominates $\frac{m_i}{\sum_k m_k}$-share allocation.

Algorithm: A Mechanism (scheduling)



Truthful

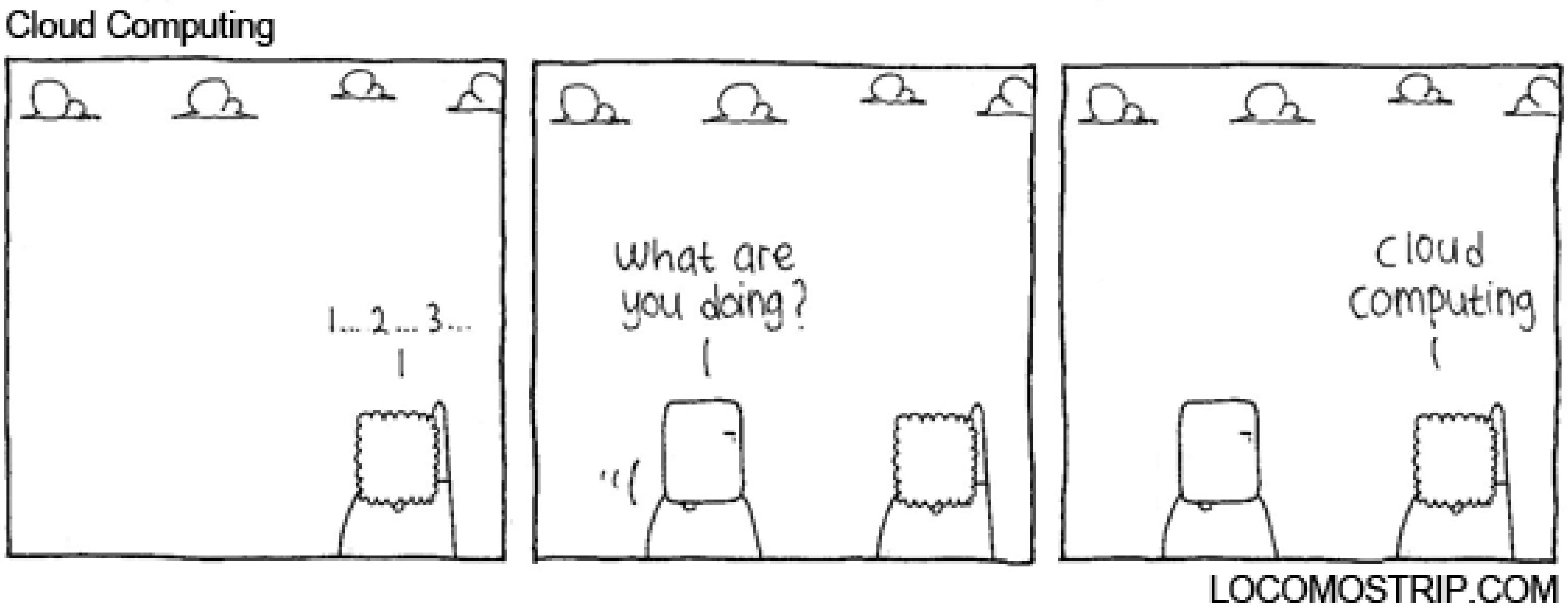
Quasi-Linear: delay-cost + $\eta_i$payment

Theorem: There is no truthful, Pareto optimal, and anonymous auction, for the case of a single good and two agents.

Uses Dobzinski, Lavi, Nisan (2012) construction.

Open Problems.

- Efficient algorithm for other sub-classes
- Online setting
- Discrete goods
- …


Cloud Computing
1... 2... 3...
what are you doing?
cloud computing
LOCOMOSTRIP.COM

THANK YOU

Images Curtsey

http://ogrisel.github.io/scikit-learn.org/sklearn-tutorial/modules/clustering.html
http://ordination.okstate.edu/MULTIPLE.htm
http://www.dreamstime.com/
http://www.grabstats.com/
http://www.smartdatacollective.com/junedghanchi
http://javarevisited.blogspot.de/2014/05/top-5-cloud-service-providers-companies-Java-IT-professional-know.html
http://www.forbes.com/forbes/welcome/
http://www.wired.com/2014/03/urs-google-story/
http://www.investopedia.com/articles/investing/032715/cloudcomputing-industry-exponential-growth.asp?header_alt=c
http://www.enterprisetech.com/2014/11/03/forecasts-call-cloud-burst-2018/
http://theearthplanet.com/advantages-of-cloud-computing/
http://www.datacentertalk.com/2015/10/the-next-generation-enterprises-will-not-stop-at-corporate-firewall/
http://spinlab.wpi.edu/projects/clusterflop/clusterflop.html