Scheduling with Energy & Network Constraints

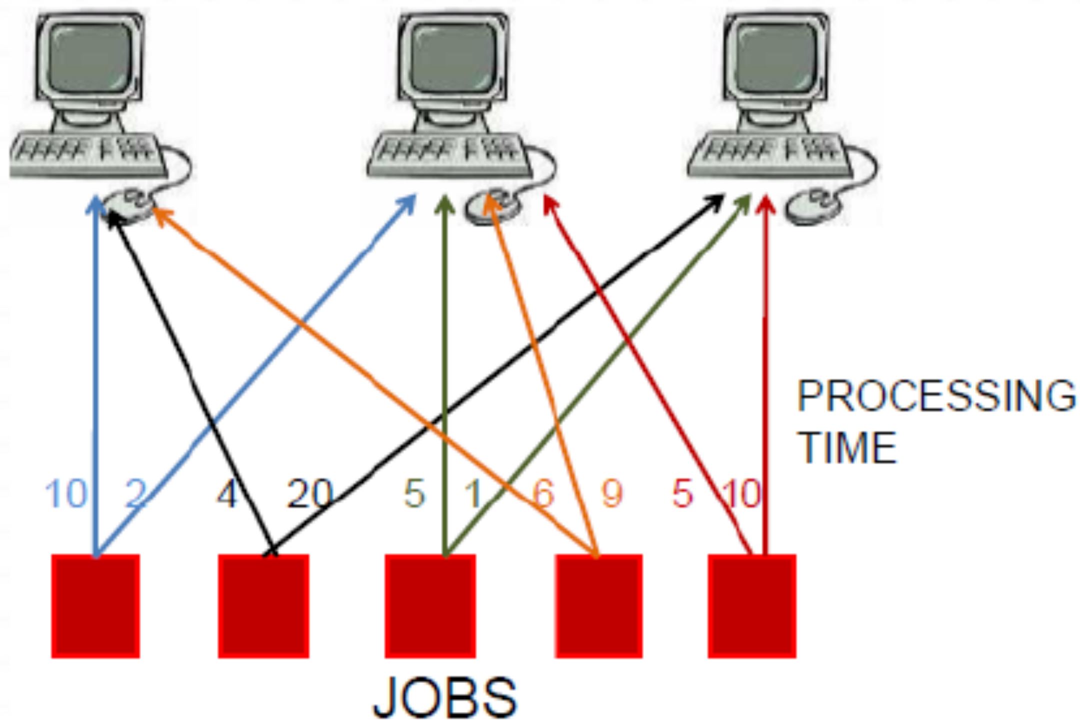
Barna Saha
College of Information & Computer Science
UMass Amherst

Collaborators: Samir Khuller, Jian Li, Manish Purohit

Job Scheduling

Unrelated Parallel Machines (UPM)

- Data centers contain heterogeneous machines varying in computing capability
- A job can only be processed in a machine, if required data is available in the machine-memory.



$p_{i,j}$: Processing time of job *j* on machine *i*.
They are unrelated

Job Scheduling

Unrelated Parallel Machines (UPM)

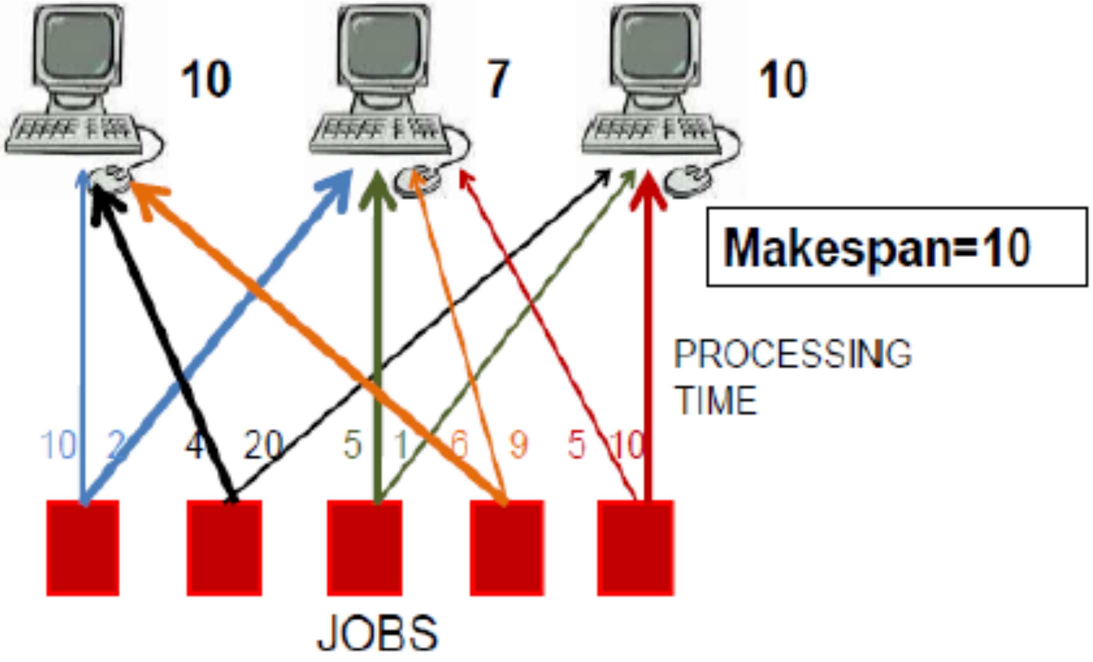
- **Makespan Minimization** [Lenstra, Shmoys, Tardos'90]

Minimize maximum load (sum of the processing time of the allocated jobs) on any machine

- **Generalized Assignment Problem (GAP)** [Shmoys, Tardos'93]

job j has a cost $c_{(i,j)}$ to be assigned on machine i.

Unrelated Parallel Machines



$p_{i,j}$: Processing time of job j on machine i. They are unrelated

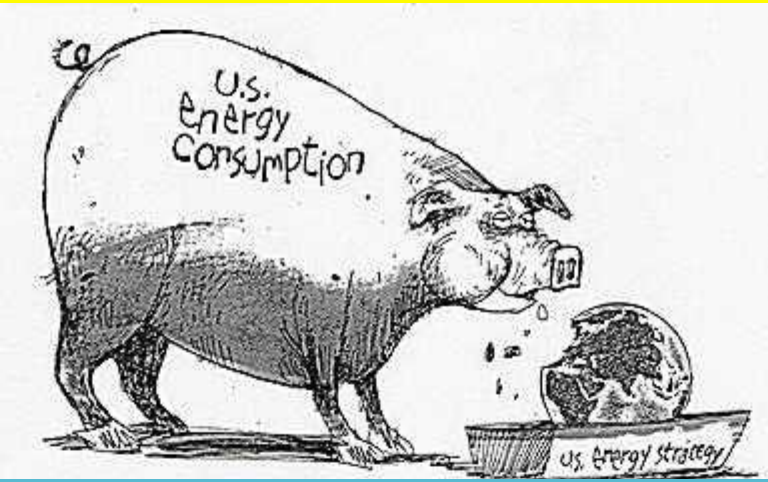
Data Center Scheduling

Unrelated Parallel Machines (UPM)

Data Center Scheduling

Infrastructure as a service

Resources are scarce and limited!

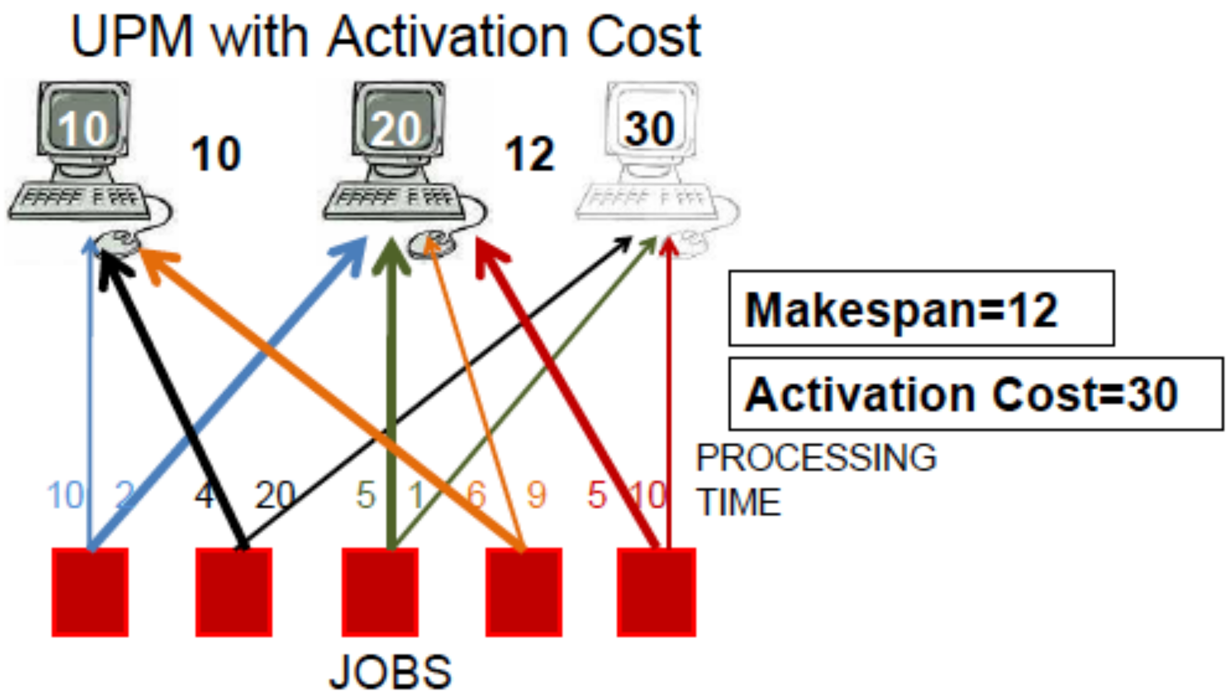


Scheduling it turns out, comes down to how to spend money !

- Limited resources must be distributed efficiently to optimize system performance, profit, social fairness etc.
- Energy savings has become a critical issue with the advent of data centers.
- Computing moving in the cloud requires optimized scheduling mechanism.

A Simple Model for Saving Energy

- Scheduling with Activation
  - Minimize energy by selectively shutting down machine [Khuller, Li, Saha, SODA 2010]



Each machine has an activation cost

Minimize the total activation cost while maintaining makespan

Scheduling with Activation

- Result

LP-Rounding: $2+\epsilon$ approximation for makespan and $(2+\frac{1}{\epsilon})(\ln\frac{n}{OPT}+1)$ approximation for activation cost.
Greedy: 2 approximation for makespan and $(1+\ln n)$ approximation for activation cost.

- Extensions:
  - GAP to consider energy consumption for job processing
  - Better bounds for related machine scheduling
  - Multi-dimensional jobs

Subsequent Works

- Generalized activation cost
  - Activation cost is a function of machine load [Li, Khuller, SODA 2011]
  - multi-dimensional jobs
- Online: jobs arrive online
  - [Azar, Bhaskar, Fleischer, Panigrahi, SODA 2013]
  - Meyerson, Roytman, Tagiku (multi-dimensional job), APPROX-RANDOM 2013]

Scheduling with Machine Activation

- Guess the optimum makespan T.

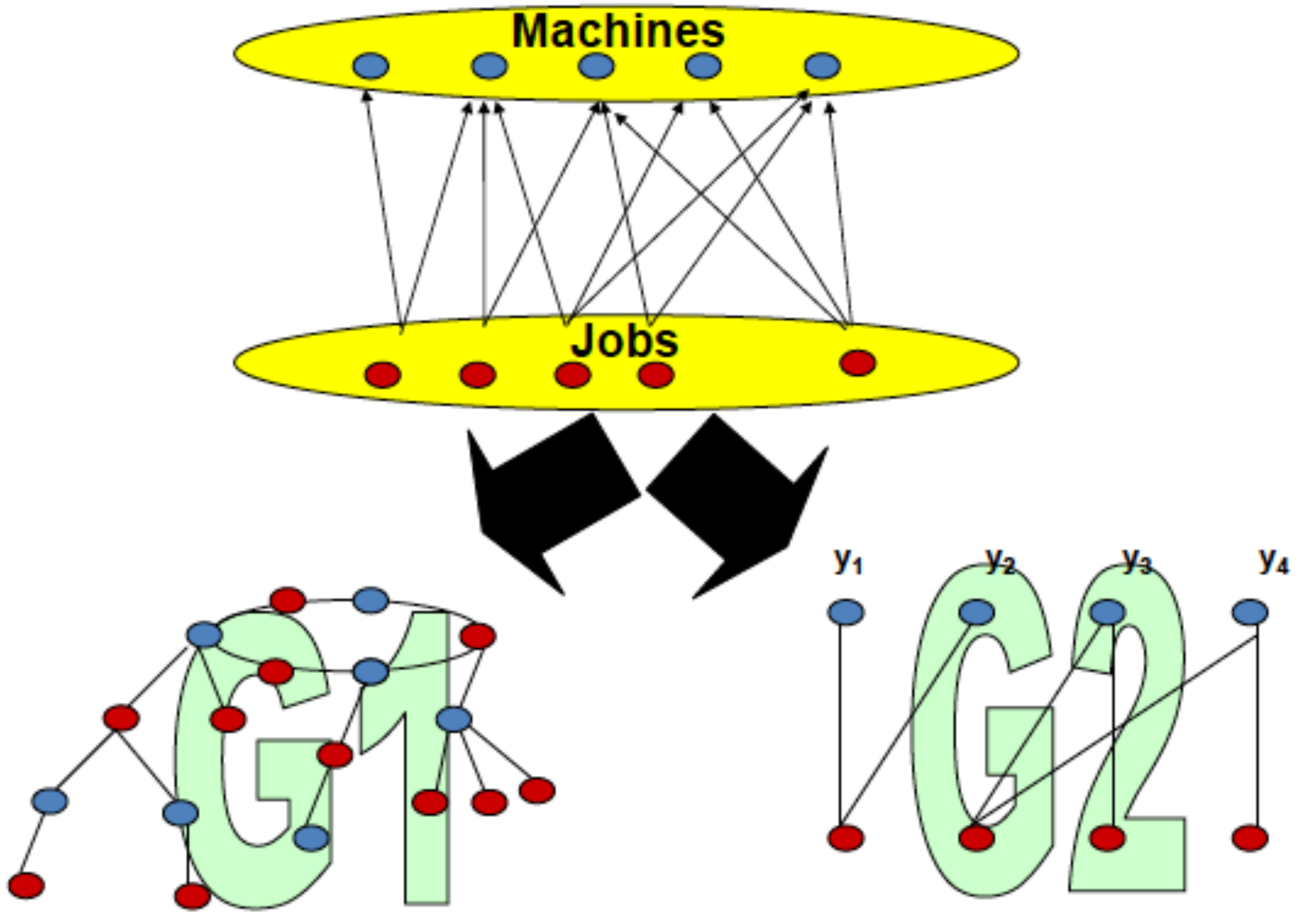
$$\min \quad \sum_{i=1}^{m} a_i y_i + \sum_{(i,j)} c_{i,j} x_{i,j}$$

$$\sum_i x_{i,j} \geq 1 \ \forall j \qquad (\text{Assign})$$

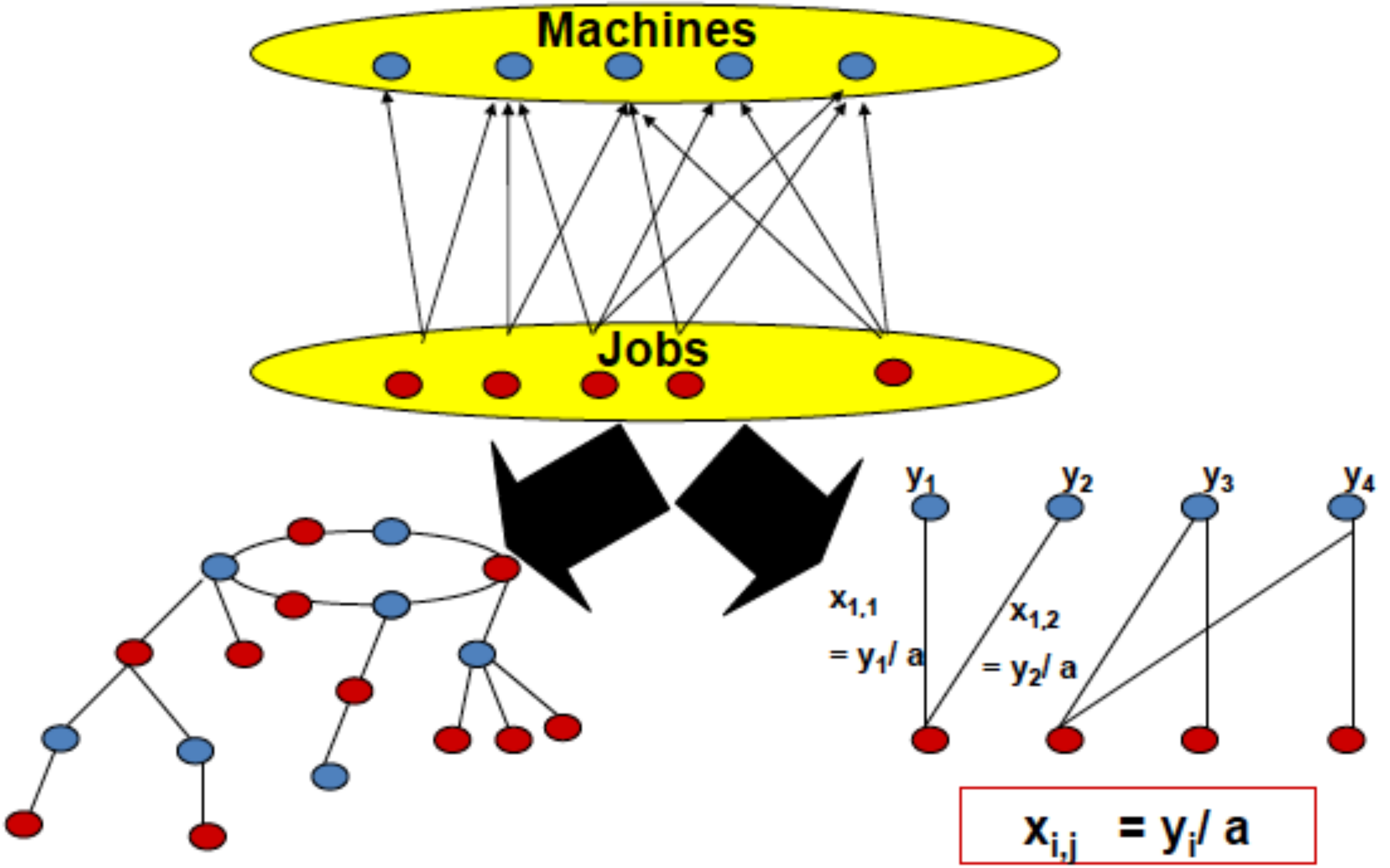
$$\sum_j p_{i,j} x_{i,j} \leq T y_i \ \forall i \qquad (\text{Load})$$

$$x_{i,j} \leq y_i \ \forall i \in M, j \in J$$

$$x_{i,j} \in \{0,1\} \ \forall i,j \ \text{ and } \ x_{i,j} = 0 \ \text{ if } p_{i,j} > T$$

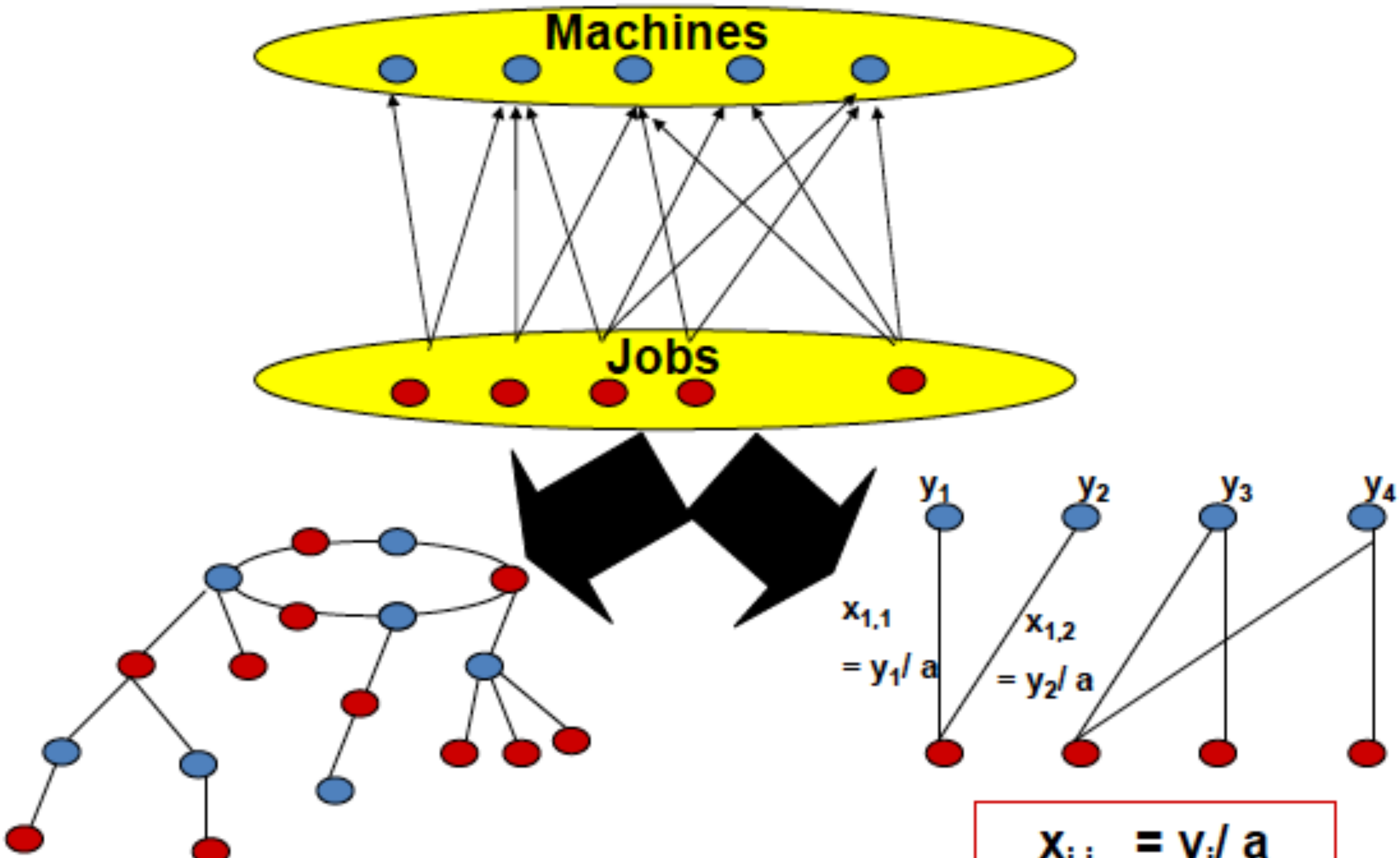


Machines
Jobs
G1
G2
y1
y2
y3
y4



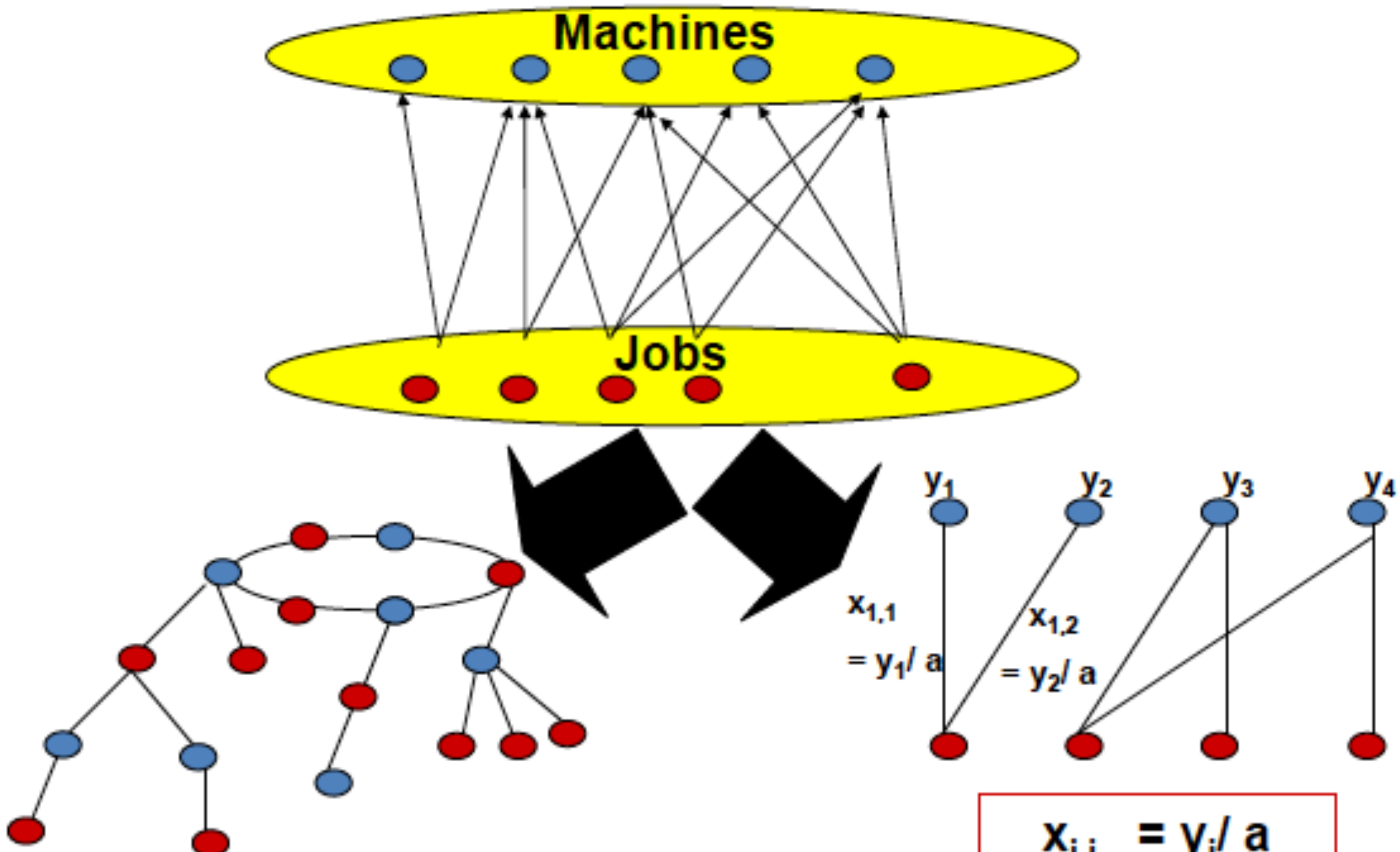
$$x_{i,j} = y_i / a$$

$$\sum_j x_{i,j} p_{i,j} <= T_2 y_i$$



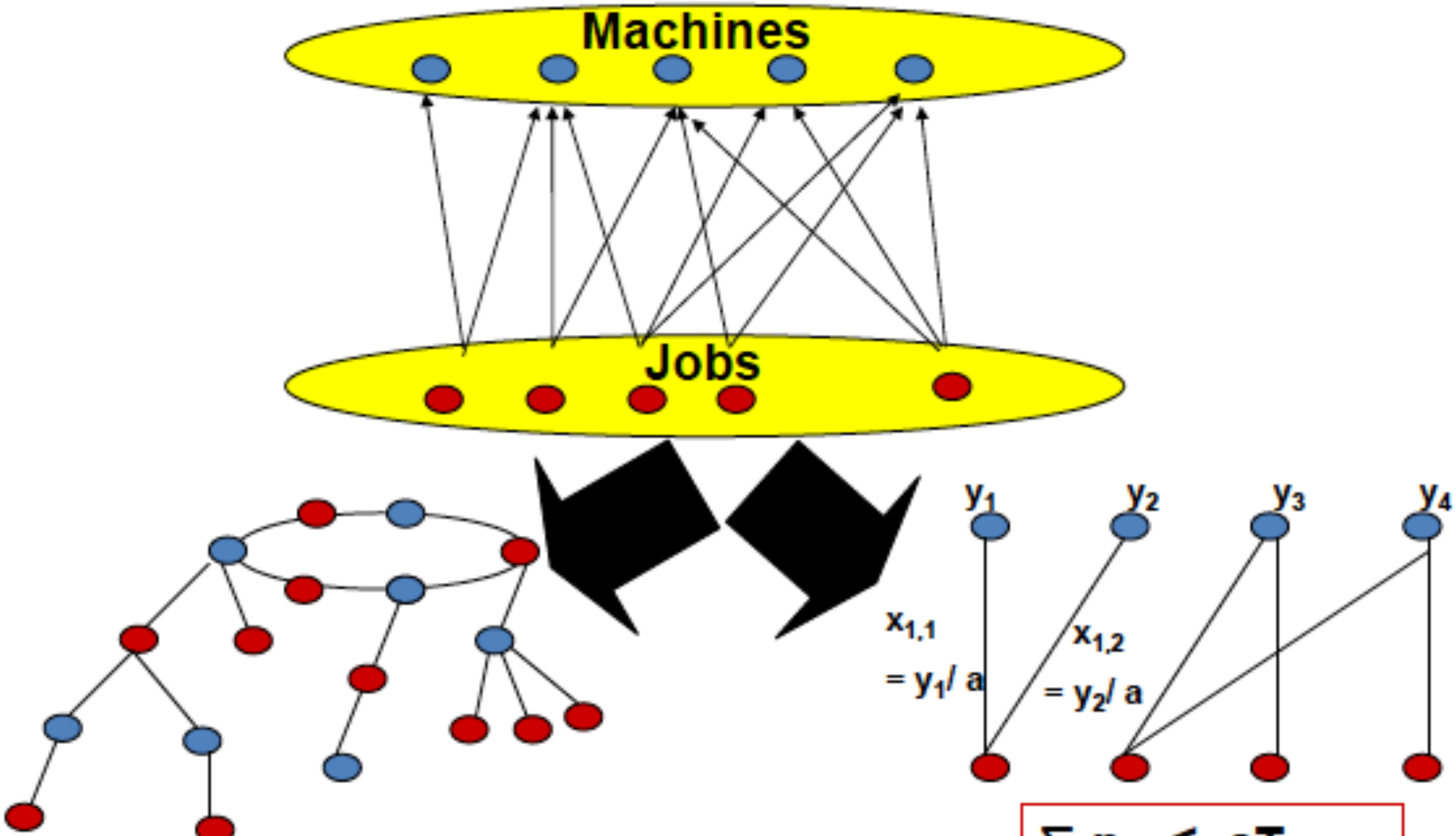
$$x_{i,j} = y_i / a$$

$$\sum_j y_i p_{i,j} / a \leq T_2 y_i$$



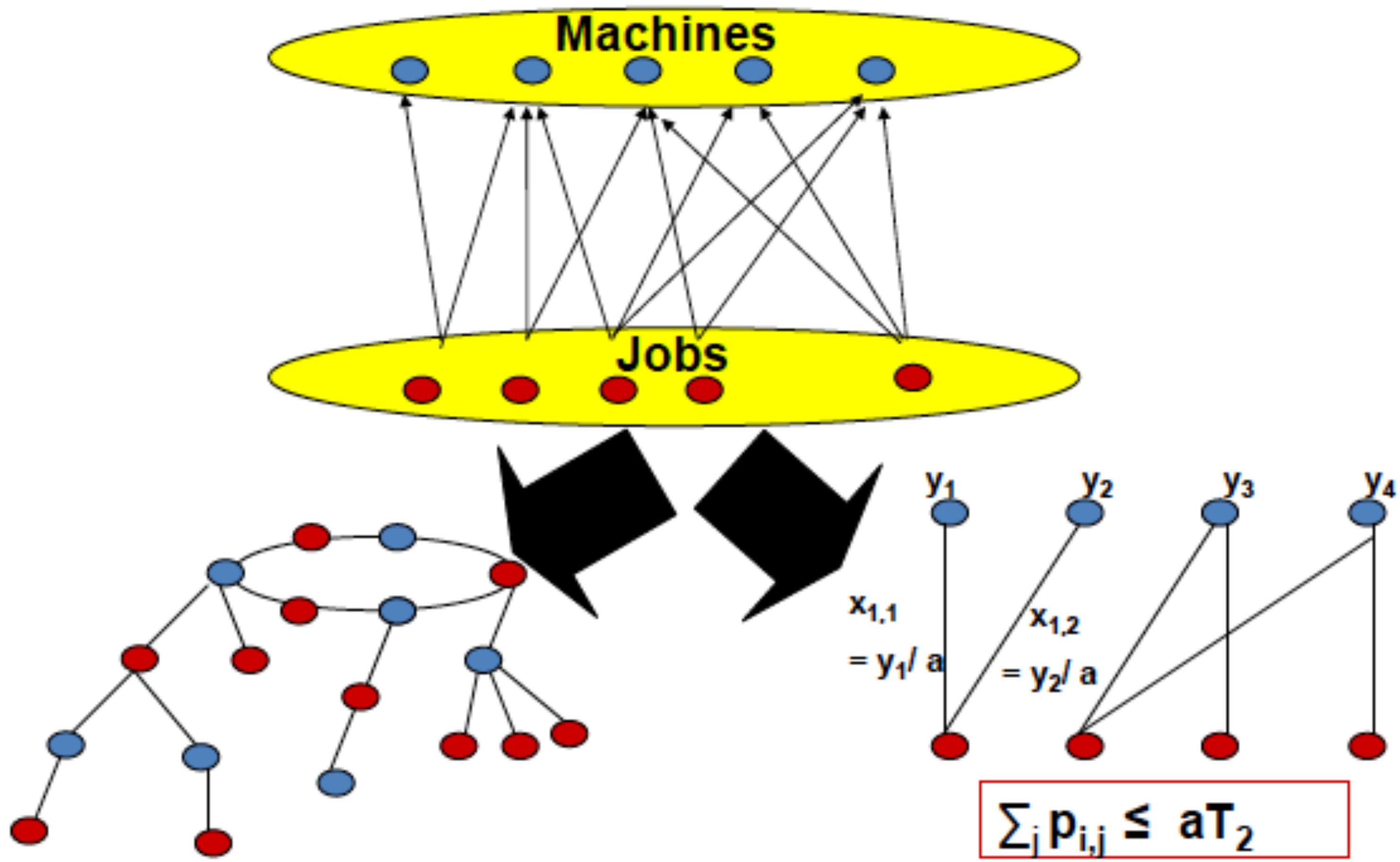
$$x_{i,j} = y_i / a$$

$$\sum_j p_{i,j} \leq aT_2$$

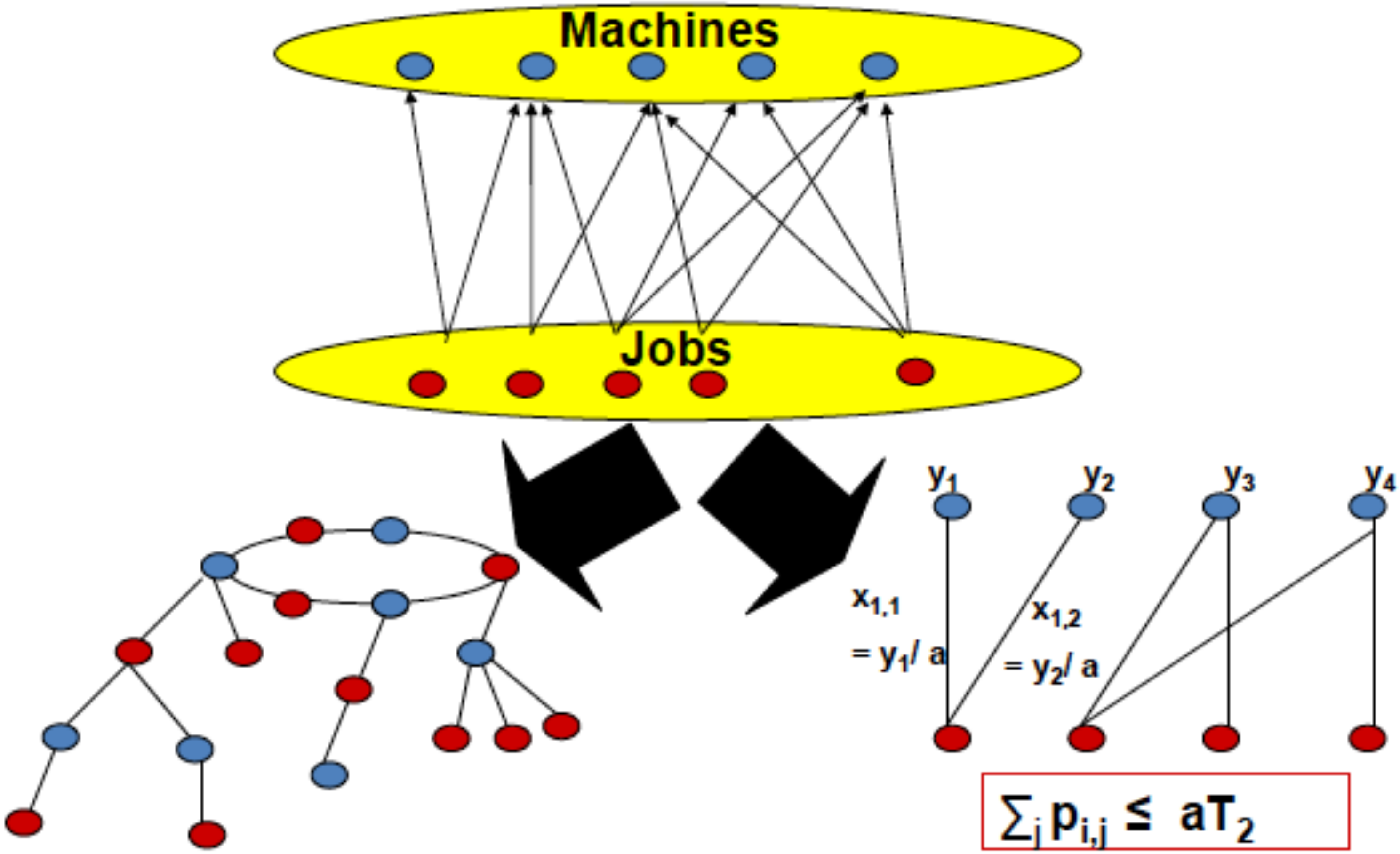


$\sum_j p_{i,j} \leq aT_2$

If a machine is opened, schedule all the jobs fractionally assigned to it



Machine=Sets, Jobs=Elements



If the total fractional assignment of the jobs is ≥ 1:

Fractional Set Cover Instance

A Special Case

- Unit jobs:
  - makespan ≈ capacity constraint on machines
- Uniform activation cost

Set Cover with Hard Capacities

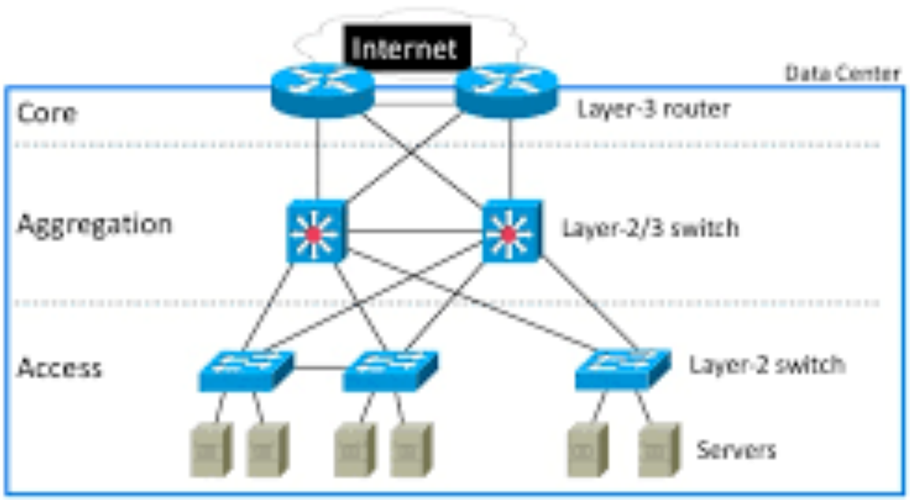
Makespan/capacity constraints are strictly maintained

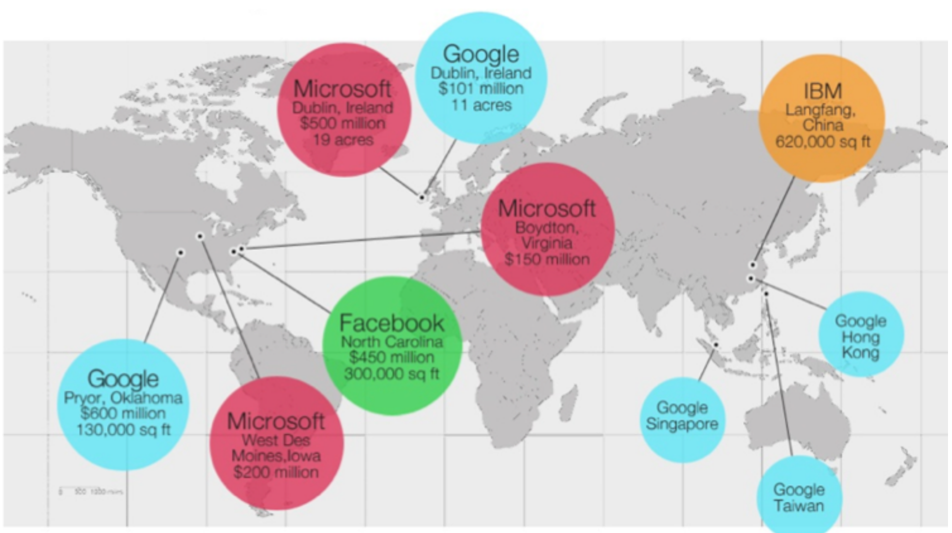
Set Cover with Hard Capacities

- Weighted version: O(log(n)) approximation follows from a classical result by Wolsey from 1982.
- Unweighted version:
  - 3-approximation for vertex cover by Chuzhoy and Naor, FOCS 2005
  - Max(6f,65)-approximation for set cover by Saha and Khuller, ICALP 2012 [f=maximum #no of sets an element belongs to]
  - Subsequent improvements in SODA 2014 by Cheung, Goemans and Wong to 2f-approximation and SODA 2017 independently by Wong and to get a f-approximation for set cover with hard capacities.

What is Missing?

Common DC Topology



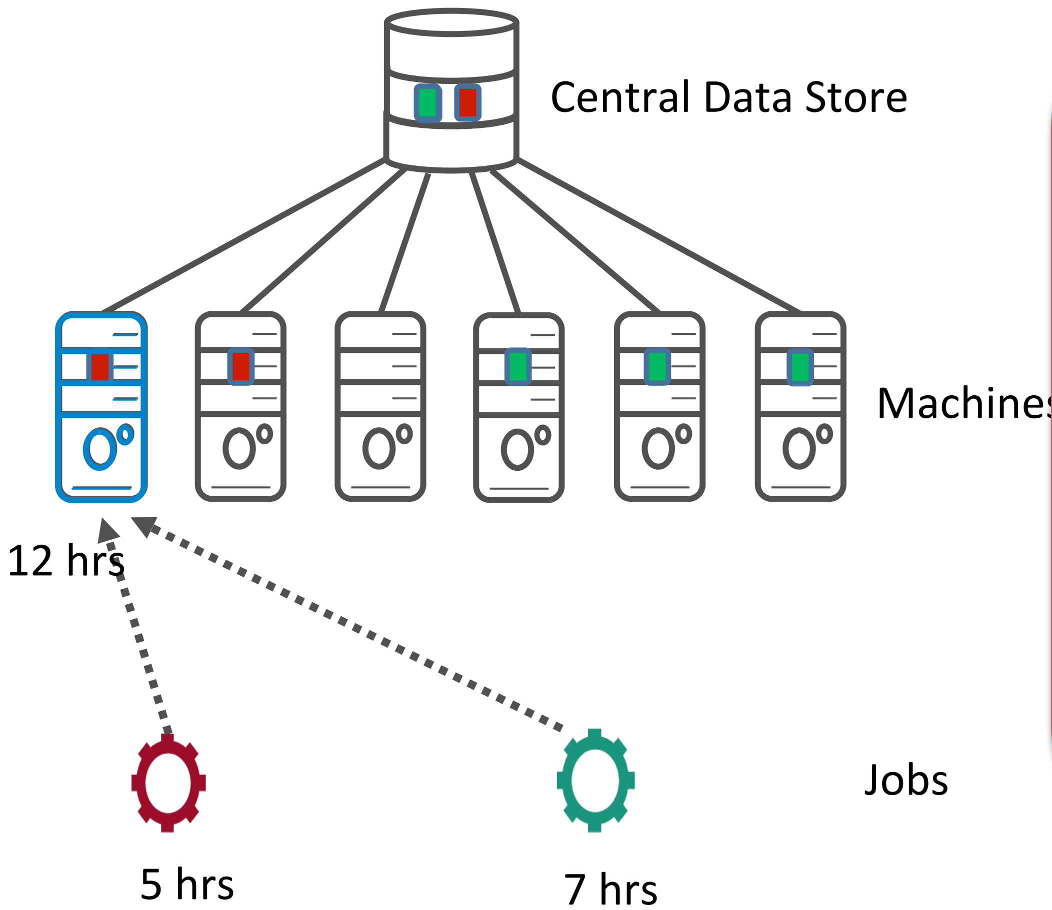


- Data center machines are inter-connected by network
- Restricting a job to run only on a few machines containing requisite data is restrictive

Scheduling with Energy and Network Constraints

- Jobs can be scheduled on any machine as long as data can be transferred to it.
- Each network link has limited bandwidth which limits how much data can be transferred on them.

Framework: Star Network



Goal:
Activate minimum # machines

Constraints:
At every active machine

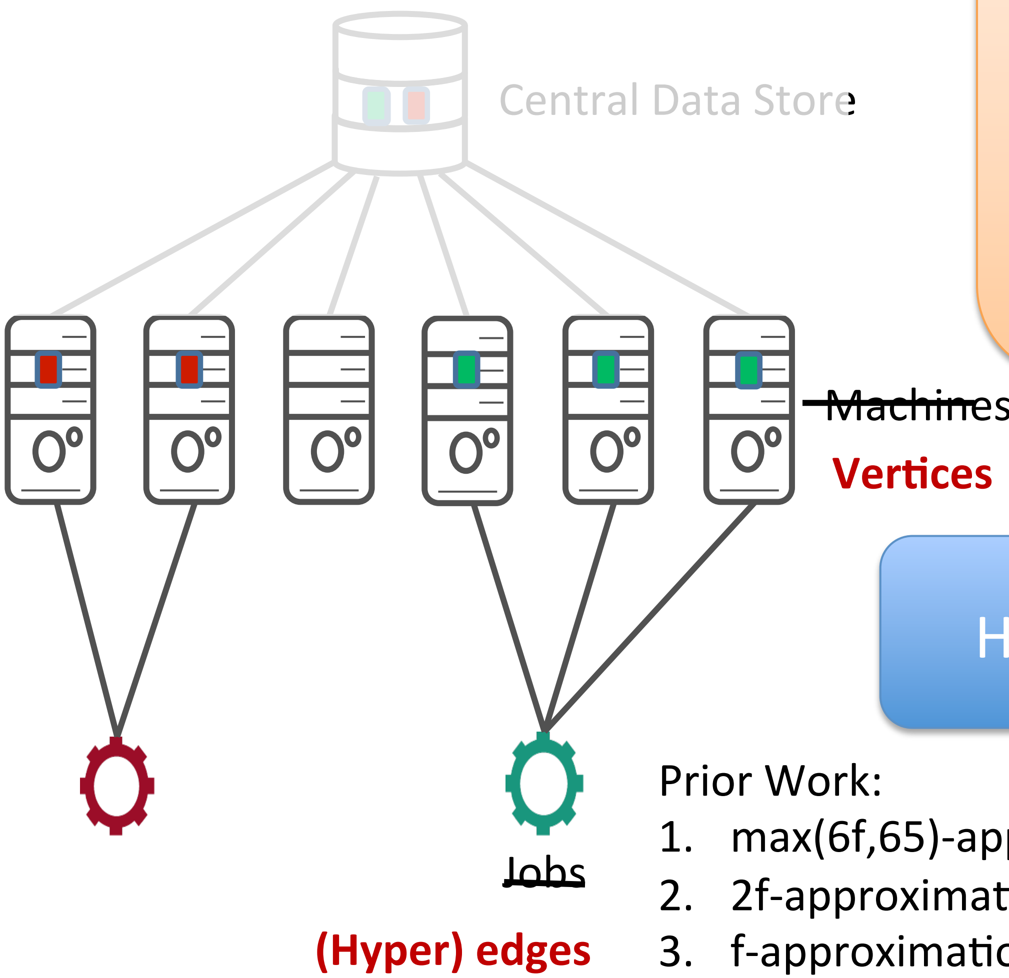
- Total processing time < T

Makespan

- Total data transfer < B

Congestion

Connections to Classical Problems



Goal:
Activate minimum # machines

Constraints:
At every active machine

- Total processing time < T
- Total data transfer < 0

Hard Capacitated Set Cover

Prior Work:

1. max(6f,65)-approximation: Saha, Khuller, ICALP 2012
2. 2f-approximation: Cheung, Goemans, Wong, SODA'14
3. f-approximation: Wong, SODA'17
4. f-approximation: Kao, SODA'17

Our Results

Network-Aware Machine Activation

Open minimum # machines s.t. makespan $\leq T$ and congestion $\leq B$

- For unit jobs
  - (4f+4)-approximation algorithm

- For jobs with arbitrary processing and data requirements
  We find a solution that opens (8f+8)OPT machines and has makespan 5T and congestion 4B

Linear Programming Relaxation

Variables: y_i : Is machine i active?

x_{ij} : Is job j assigned to machine $i \in \delta(j)$

z_{ij} : Is job j assigned to machine $i \notin \delta(j)$

$$Minimize \sum_{i \in M} y_i$$

Subject to,

$\forall\, j \in J$ $\quad \sum_{i \in \delta(j)} x_{ij} + \sum_{i \notin \delta(j)} z_{ij} \geq 1$ — Every job is assigned to a machine

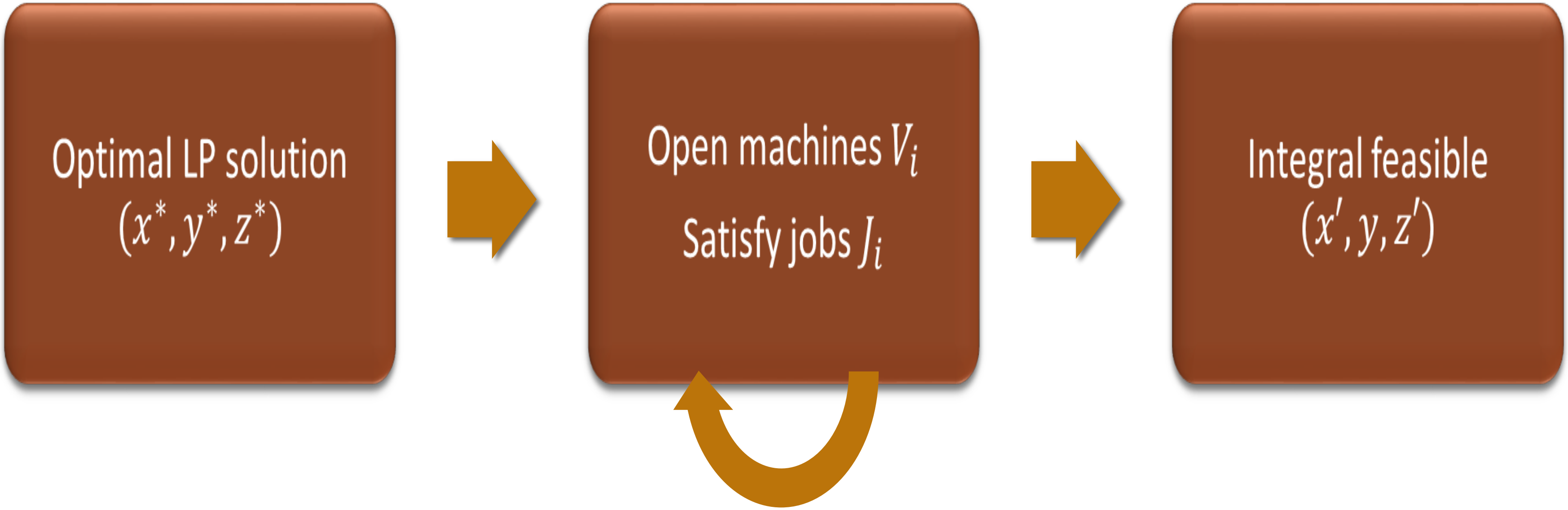
$\forall\, j \in J$ and $i \in M$ $\quad x_{ij} + z_{ij} \leq y_i$ — Machine needs to be open

$\forall\, i \in M$ $\quad \sum_{j \in J} (x_{ij} + z_{ij}) \leq C_i y_i$ — Capacity constraints

$\forall\, i \in M$ $\quad \sum_{j \in J} z_{ij} \leq B_i y_i$ — Bandwidth constraints

$$\mathbf{0 \leq x, y, z \leq 1}$$

Our Approach



Why is it harder than Hard-Capacity Set Cover?

Variables: y_i : Is vertex i active?
x_{ij} : Is edge j assigned to vertex $i \in \delta(j)$
z_{ij} : Is job j assigned to machine $i \notin \delta(j)$

$$Minimize \sum_{i \in M} y_i$$

Subject to,

At least one is $\geq \frac{1}{f}$

$\forall\, j \in J$ $$\sum_{i \in \delta(j)} x_{ij} + \sum_{i \notin \delta(j)} z_{ij} \geq 1$$

$\forall\, j \in J$ and $i \in M$ $$x_{ij} + z_{ij} \leq y_i$$

$\forall\, i \in M$ $$\sum_{j \in J} (x_{ij} + z_{ij}) \leq C_i y_i$$

$\forall\, i \in M$ $$\sum_{j \in J} z_{ij} \leq B_i y_i$$

$$\mathbf{0 \leq x, y, z \leq 1}$$

Cheung et al. (SODA 2014)

1. Open all vertices with $y_i \geq \frac{1}{f}$ (Say V_1)
2. Every edge is incident on an open vertex
3. Open $|V_1|$ more vertices to satisfy the capacity constraints
   [Auxiliary Linear Program]

Our Algorithm – A Brief Overview

Stage 1: Open all machines with $y_i \geq \alpha$ (Say V_1)

Stage 2 (Cheung et al) : Open $|V_1|$ more machines *to satisfy jobs that are incident on* V_1

Stage 3: Handle "external" demands, i.e. set $z_{ij} = 0$ for all machines not yet open

Say O

Stage 4: Open $|V_1|$ more machines to satisfy "not-tight" jobs

Stage 5: Open $|V_1|$ more machines to satisfy all remaining jobs

Reduced Instance

$\forall$ unsatisfied j $$\sum_{i \notin O} x_{ij} + \sum_{i \in O} z_{ij} \geq 1 - \sum_{i \in O} x_{ij}$$

$\forall$ unsatisfied $j \in J$ and $i \notin O$ $$x_{ij} \leq y_i$$

$\forall\, i \notin O$ $$\sum_{j \in J} x_{ij} \leq C_i y_i$$

$\forall\, i \in O$ $$\sum_{j \in J} z_{ij} \leq B_i$$

$$\mathbf{0 \leq x, y, z \leq 1}$$

Key Idea:

If all $x_{ij} \in \{0, y_i\}$, drop capacity constraints

Can we ensure all $x_{ij} \in \{0, y_i\}$?

Yes! Use iterative rounding!

Final Stage:

The LP now has a much simpler structure!

Admits an easy iterative rounding strategy

Our Algorithm – A Brief Overview

Open Machines ≤ 4$|V_1|$ = 4(f+1)Opt

Stage 1: Open all machines with $y_i \geq \alpha$ (Say V_1)

Stage 2 (Cheung et al) : Open $|V_1|$ more machines *to satisfy jobs that are incident on* V_1

Stage 3: Handle "external" demands, i.e. set $z_{ij} = 0$ for all machines not yet open

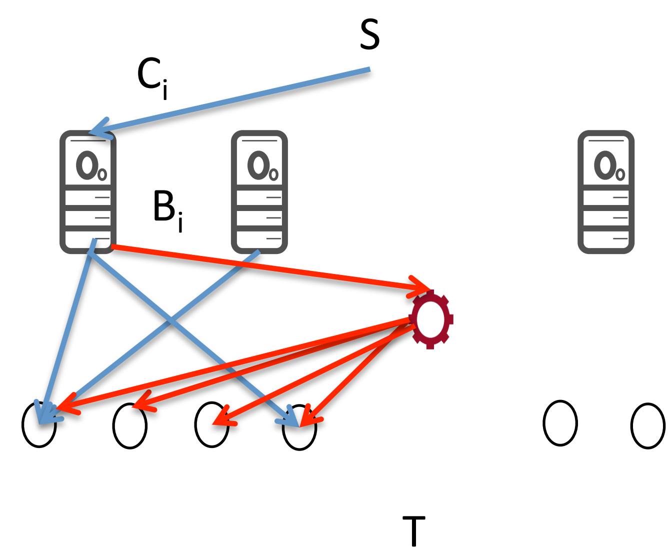
Stage 4: Open $|V_1|$ more machines to satisfy "not-tight" jobs

Stage 5: Open $|V_1|$ more machines to satisfy all remaining jobs

How can we obtain integral assignments?

Define a flow network such that

- All capacities are integers
- x_{ij} and z_{ij} define a fractional flow



Extension – Arbitrary Job Sizes

Open Machines $\leq 8|V_1| = 8(f+1)Opt$

Stage 1: Open all machines with $y_i \geq \alpha$ (Say V_1)

Stage 2 (Cheung et al) : Open $|V_1|$ more machines *to satisfy jobs that are incident on* V_1

Stage 3: Handle "external" demands, i.e. set $z_{ij} = 0$ for all machines not yet open

Stage 4: Open **3**$|V_1|$ more machines to satisfy "not-tight" jobs

Stage 5: Open **3**$|V_1|$ more machines to satisfy all remaining jobs

LP structure is more involved

How can we obtain integral assignments?

Not a flow problem any more!

Use techniques similar to the *Generalized Assignment Problem*

Makespan <= 5T, Congestion <= 4B

Next Steps

- Extend to hierarchical tree network
- Online algorithms
- Other performance criteria: completion time, flow time etc.
- Consider arbitrary activation cost, energy consumption for job processing
- .
- .

In a nut shell, to obtain the best performance it is not enough to treat machines independently—one needs to consider the underlying network and the possibility of data transfer at the time of scheduling.

This could lead to interesting questions both of theoretical and practical interest.

Thank You!