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)

0 Data centers contain heterogeneous machines varying in computing capability.

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

Unrelated Parallel Machines (UPM)
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 
\((2 + \frac{1}{\epsilon})(\ln \frac{n}{\text{OPT}} + 1)\) 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 $T$.

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

\[
\sum_{i} x_{i,j} \geq 1 \quad \forall j \quad \text{(Assign)}
\]

\[
\sum_{j} p_{i,j} x_{i,j} \leq T y_i \quad \forall i \quad \text{(Load)}
\]

\[
x_{i,j} \leq y_i \quad \forall i \in M, j \in J
\]

\[
x_{i,j} \in \{0, 1\} \quad \forall i,j \text{ and } x_{i,j} = 0 \text{ if } p_{i,j} > T
\]
\[ x_{i,j} = \frac{y_i}{a} \]

\[ \sum_j x_{i,j} p_{i,j} \leq T_2 y_i \]
\[ x_{i,j} = \frac{y_i}{a} \]
\[ \sum_j p_{i,j} \leq \alpha T_2 \]
If a machine is opened, schedule all the jobs fractionally assigned to it.

\[ \sum_j p_{i,j} \leq aT_2 \]
Machine=Sets, Jobs=Elements

\[
\sum_j p_{i,j} \leq aT_2
\]
If the total fractional assignment of the jobs is $\geq 1$:

**Fractional Set Cover Instance**

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

$x_{1,1} = \frac{y_1}{a}$

$x_{1,2} = \frac{y_2}{a}$
A Special Case

• Unit jobs:
  – makespan \approx \text{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?

- 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

Central Data Store

Goal:
Activate minimum # machines

Constraints:
At every active machine
- Total processing time < T
- Total data transfer < B

Makespan
Congestion
Connections to Classical Problems

**Goal:**
Activate minimum # machines

**Constraints:**
- Total processing time < T
- Total data transfer < 0

**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

Minimize \( \sum_{i \in M} y_i \)

Subject to,

\( \forall j \in J \)

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

\( \forall j \in J \) and \( i \in M \)

\( x_{ij} + z_{ij} \leq y_i \)

\( \forall i \in M \)

\( \sum_{j \in J} (x_{ij} + z_{ij}) \leq C_i y_i \)

\( \forall i \in M \)

\( \sum_{j \in J} z_{ij} \leq B_i y_i \)

\( 0 \leq x, y, z \leq 1 \)

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 \( i \notin \delta(j) \)

Every job is assigned to a machine

Machine needs to be open

Capacity constraints

Bandwidth constraints
Our Approach

Optimal LP solution $(x^*, y^*, z^*)$

Open machines $V_i$
Satisfy jobs $J_i$

Integral feasible $(x', y, z')$
Why is it harder than Hard-Capacity Set Cover?

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

Subject to,

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

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

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

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

$0 \leq x, y, z \leq 1$

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 $i \notin \delta(j)$

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

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

\[
\forall \text{ unsatisfied } j \quad \sum_{i \in \emptyset} x_{ij} + \sum_{i \in \emptyset} z_{ij} \geq 1 - \sum_{i \in \emptyset} x_{ij}
\]

\[
\forall \text{ unsatisfied } j \in J \text{ and } i \not\in \emptyset \quad x_{ij} \leq y_i
\]

\[
\forall i \not\in \emptyset \quad \sum_{j \in J} x_{ij} \leq C_i y_i
\]

\[
\forall i \in \emptyset \quad \sum_{j \in J} z_{ij} \leq B_i
\]

\[0 \leq x, y, z \leq 1\]

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

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

# Open Machines $\leq 4|V_1| = 4(f+1)Opt$

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

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

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

How can we obtain integral assignments?

Not a flow problem any more!

Use techniques similar to the Generalized Assignment Problem

Makespan $\leq 5T$, Congestion $\leq 4B$

LP structure is more involved
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 nutshell, 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!