(Jog Falls, Jog, India)
Algorithmic Challenges in Building Efficient Data Center/Cloud Infrastructure

Janardhan Kulkarni, MSR Redmond.
1. Minimum Birkhoff-von Neumann Decompositions
   K., Lee, Singh. IPCO 2017

2. ProjecTor: Agile Reconfigurable Data Center Interconnect
   Ghobadi, Mahajan, Phanishayee, Devanur, K., Ranade, Blanche, Rastegarfar, Glick, Kilper. SIGCOMM'16.

3. Truth and Regret in Online Scheduling
   Chawla, Devanur, K., Niazaeh. EC 2017
Two Problems in Resource Allocation

Problem 1: Matching Decomposition

Reconfigurable Data Centers/ SDNs

Problem 2: Pricing and Scheduling VMs

Cloud Services such as Azure
The designers must decide in advance how much capacity to provision between top-of-rack (ToR) switches.

- Full interconnect is expensive
- Limits application performance when demand between two ToRs exceeds capacity
Reconfigurable Topologies

Change the topology based on traffic!

ProjecTor, MSR. (SIGCOMM'16)

Mordia, Google. (SIGCOMM'13)
Reconfigurable Topologies

Matching between senders and receivers
## Reconfigurable Topologies

<table>
<thead>
<tr>
<th></th>
<th>Senders</th>
<th>Receivers</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>100</td>
</tr>
<tr>
<td>0</td>
<td>10</td>
<td>60</td>
</tr>
<tr>
<td>0</td>
<td>20</td>
<td>10</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>70</td>
</tr>
<tr>
<td>0</td>
<td>20</td>
<td>0</td>
</tr>
</tbody>
</table>

Matching between senders and receivers
Matching Decomposition

Given a traffic matrix, find an efficient way route the traffic.

Traffic Matrix

\[
\begin{pmatrix}
10 & 20 & 0 & 0 & 5 \\
0 & 30 & 0 & 10 & 0 \\
0 & 0 & 90 & 0 & 0 \\
70 & 20 & 60 & 40 & 10 \\
\end{pmatrix}
\]

A sequence of matchings between senders and receivers

\[= \quad + \quad + \]

QUESTION

Example

\[
\begin{pmatrix}
  \frac{1}{2} & \frac{1}{4} & \frac{1}{4} \\
  \frac{1}{4} & \frac{1}{2} & \frac{1}{4} \\
  \frac{1}{4} & \frac{1}{4} & \frac{1}{2}
\end{pmatrix}
\]

\[
\begin{pmatrix}
  A & B & C \\
  A & B & C \\
  X & Y & Z
\end{pmatrix}
\]

\[
\frac{1}{2} + \frac{1}{4} + \frac{1}{4}
\]
Example

\[
\begin{bmatrix}
    X & Y & Z \\
    A & \frac{1}{2} & \frac{1}{4} & \frac{1}{4} \\
    B & \frac{1}{4} & \frac{1}{2} & \frac{1}{4} \\
    C & \frac{1}{4} & \frac{1}{4} & \frac{1}{2}
\end{bmatrix}
\]
Example

QUESTION: How to decompose a given traffic matrix using smallest number of matchings?
Example

\[
\begin{pmatrix}
A & 2 & 0 & 0 \\
B & 1 & 1 & 0 \\
C & 0 & 2 & 1 \\
\end{pmatrix}
\]

Birkhoff-von Neumann Theorem

Algorithmic Carathéodory's Theorem
There is a logarithmic approximation to the minimum Birkhoff-von Neumann decomposition problem.

THEOREM: (K.- Lee- Singh’17)

- Solve a linear program.
- Do randomized rounding.
- Apply Lovasz Local Lemma (LLL) to prove the theorem.
Minimum Birkhoff-von Neumann Decompositions

**GOAL:** \( M = \lambda_1 P_1 + \lambda_2 P_2 + \ldots + \lambda_k P_k \)

**THEOREM:** (K.- Lee- Singh’17)

- Solve a linear program.
- Do randomized rounding.
- Apply Lovasz Local Lemma (LLL) to prove the theorem.

There is a logarithmic approximation to the minimum Birkhoff-von Neuman decomposition problem.

*BVN Decomposition algorithm can be exponentially bad.*
Online + Decentralized Algorithm?

ProjecTor (MSR)

➢ Online
➢ Decentralized

ProjecTor:
Ghobadi, Mahajan, Phanishayee, Devanur, K., Ranade, Blanche, Rastegarfar, Glick, Kilper. 16.
Online + Decentralized Algorithm?

Sender id

receivers

senders
Online + Decentralized Algorithm?

*Online algorithm decides on the matching*
Online + Decentralized Algorithm?

Online algorithm decides on the matching
Online + Decentralized Algorithm?

How to reconfigure topology to minimize the average latency of the packets?

Online algorithm decides on the matching
Distributed Stable Marriage

**Theorem.** (Ghobadi, Mahajan, Panishree, Devanur, K., Ranade ‘16)
The Stable marriage algorithm is constant competitive to the objective of average latency of packets.

**Preference = function of number of packets in queue**
Takeaway

Fundamental Questions in Matching Theory

➢ Stable marriage algorithm
➢ Algorithmic version of Birkhoff–von Neumann
➢ Algorithmic Carathéodory's Theorem

Change the topology based on traffic!

ProjecTor, MSR.
Pricing and Scheduling VMs
Pricing and Scheduling in Azure

➢ How to price Virtual Machines?
➢ How to pack/schedule VMs on a cluster?
Attempt 1: Modeling the Problem

- A set of jobs arrive online
- Each job has value, and interval of time where it demands a set of resources.
  - Demands a unit of CPU for some duration $\ell$
- Service provider accepts/rejects jobs based on two factors:
  1) Amount of resources available in the system
  2) Value of job
Attempt 1: Modeling the Problem

- A set of jobs arrive online
- Each job has *value*, and *interval of time* where it demands a set of resources.
  - Demands a unit of CPU for some duration $\ell$
- Service provider accepts/rejects jobs based on two factors:
  1) Amount of resources available in the system
  2) Value of job

Schedule as many jobs as possible to maximize the total value.
A set of jobs arrive online. Each job has value, and interval of time where it demands a set of resources. Demands a unit of CPU for some duration $\ell$

Service provider accepts/rejects jobs based on two factors:

1) Amount of resources available in the system
2) Value of job

Schedule as many jobs as possible to maximize the total value.

Large literature in online scheduling to maximize throughput. [ILM’16, JMNY’15, LMNY’13, … CI’98, KSM’94, KS’92…]

**Strong lower bounds:**

No algorithm can do better than logarithmic factors in the worst case analysis.
Azure Virtual Machines gives you the flexibility of virtualization for a wide range of computing solutions with support for Linux, Windows Server, SQL Server, Oracle, IBM, SAP, and more. Select from a wide variety of virtual machine sizes. Virtual machines are billed on per-minute basis and most include load-balancing and auto-scaling free of charge.

### Virtual machines categories

#### General Purpose

Balanced CPU-to-memory ratio. Ideal for testing and development, small to medium databases, and low to medium traffic web servers.

<table>
<thead>
<tr>
<th>INSTANCE</th>
<th>CORES</th>
<th>RAM</th>
<th>DISK SIZES</th>
<th>PRICE</th>
</tr>
</thead>
<tbody>
<tr>
<td>A0</td>
<td>1</td>
<td>0.75 GB</td>
<td>20 GB</td>
<td>$0.018/hr</td>
</tr>
<tr>
<td>A1</td>
<td>1</td>
<td>1.75 GB</td>
<td>40 GB</td>
<td>$0.024/hr</td>
</tr>
<tr>
<td>A2</td>
<td>2</td>
<td>3.50 GB</td>
<td>60 GB</td>
<td>$0.068/hr</td>
</tr>
<tr>
<td>A3</td>
<td>4</td>
<td>7.00 GB</td>
<td>120 GB</td>
<td>$0.176/hr</td>
</tr>
<tr>
<td>A4</td>
<td>8</td>
<td>14.00 GB</td>
<td>240 GB</td>
<td>$0.352/hr</td>
</tr>
</tbody>
</table>

*Storage values for disk sizes use a legacy “GB” label. They are actually calculated in gibabytes, and all values should be read as “X GiB”.*

A0-4 – Basic [More information >](#)
Attempt 2: Modeling the Problem

Declares a price $p$.

A job that has value per unit length greater than $p$ is accepted and scheduled in FIFO order.

Best hindsight price that maximizes the total value of jobs.
Example

Benchmark: Best hindsight price that maximizes the total value of jobs.
Scheduling Policy: First in First Out (FIFO)

Value = 1, deadline \([t, t+1]\)

Value = 2, deadline \([t, t+2]\)

Value = 2, deadline \([t, t+1]\)

All jobs need one unit of CPU
Example

Benchmark: Best hindsight price that maximizes the total value of jobs.
Scheduling Policy: First in First Out (FIFO)

- Value = 1, deadline \([t, t+1]\)
- Value = 2, deadline \([t, t+2]\)
- Value = 2, deadline \([t, t+1]\)

All jobs need one unit of CPU

Price = 1

Total Value (Price = 1) = 3T
Example

Benchmark: Best hindsight price that maximizes the total value of jobs.
Scheduling Policy: First in First Out (FIFO)

Value = 1, deadline \([t, t+1]\)

Value = 2, deadline \([t, t+2]\)

Value = 2, deadline \([t, t+1]\)

All jobs need one unit of CPU

Price = 2

Total Value (Price = 2) = 4T
Example

Benchmark: Best hindsight price that maximizes the total value of jobs.
Scheduling Policy: First in First Out (FIFO)

Value = 1, deadline [t, t+1]
Value = 2, deadline [t, t+2]

Both jobs need one unit of CPU

Price = 2

Total Value (Price = 2) = 2T
Example

Benchmark: Best hindsight price that maximizes the total value of jobs.
Scheduling Policy: First in First Out (FIFO)

Value = 1, deadline [t, t+1]
Value = 2, deadline [t, t+2]

Both jobs need one unit of CPU

Price = 1

Total Value (Price = 1) = 3T
Attempt 2: Modeling the Problem

Benchmark: Best hindsight price that maximizes the total value of jobs.

Can we learn the optimal price?
Regret Analysis

*The online algorithm can change/adapt its price over time.*

Benchmark: Best hindsight price that maximizes the total value of jobs.

\[
\text{Regret} = \text{Total Value (} p^* \text{)} - \text{Total Value of ALG}
\]
Regret Analysis

*The online algorithm can change/adapt its price over time.*

Benchmark: Best hindsight price that maximizes the total value of jobs.

\[
\text{Regret} = \frac{\text{Total Value} \ (p^*)}{T} - \frac{\text{Total Value of ALG}}{T}
\]

Good Learning Algorithm: *Average regret approaches zero as time increases.*
Regret Analysis

The online algorithm can change/adapt its price over time.

Benchmark: Best hindsight price that maximizes the total value of jobs.

Good Learning Algorithm: Average regret approaches zero as time increases.

Chawla et al ’17: For iid distributions, optimal solution is a pricing algorithm.
Optimal Learning Algorithm

There is an online learning algorithm that achieves optimal regret for the problem of scheduling to jobs to maximize total value.

THEOREM: Chawla-Devanur-K.-Niazadeh’17
Optimal Learning Algorithm

THEOREM: Chawla-Devanur-K.-Niazadeh’17

There is an online learning algorithm that achieves optimal regret for the problem of scheduling to jobs to maximize total value.

Truthful!

(Jobs have no incentive to lie about their value, deadlines and arrivals.)
Optimal Learning Algorithm

There is an online learning algorithm that achieves optimal regret for the problem of scheduling to jobs to maximize total value.

THEOREM: Chawla-Devanur-K.-Niazadeh’17

Regret in the case when job lengths are known in advance

\[ O(\sqrt{T}) \]

Regret in the case when job lengths are not known in advance

\[ O(T^{2/3}) \]
Optimal Learning Algorithm

There is an online learning algorithm that achieves optimal regret for the problem of scheduling to jobs to maximize total value.

MAIN IDEA

➢ Think of each price as an expert.
➢ Scheduling problem has a state.
➢ Adaptations of algorithms from experts/bandit with switching cost model.
Interesting Scheduling Problems at the Intersection
Takeaway

Interesting Scheduling Problems at the Intersection

Thank You