SCHEDULING - II

Fairness?

Plan
- Some Desirable Properties
- Scheduling Mechanisms
- Simple Algorithms
- Lottery Scheduling
- LFS (?)
- Multi Resource (DRF)
- Benchmarking + Workloads

Fairness: Why?
Why Share:
- Sharing Incentive: No Task Better If Run Alone With
  - R/N Resources

How Share:
- Pareto Efficiency: No Unused Resource

Relative Value of Resource/Allocation Depends On

Lottery
Other Considerations (Revisit)
  - Abstract/Homogenous (Lottery)
  - Strategy Proof (DRF)

Mechanics

Consider Two Types of Problem
  - Time Sharing

\[ 0 \ 1 \ 2 \ 3 \ 4 \ 5 \ldots \eta \]
\[ \eta \text{ Jobs} \]

\[ \text{SCHED} \]
\[ \text{K-Resources} \]

Space Partitioning

\[ \begin{array}{c|c|c}
0 & 0 & 0 \\
1 & 0 & 1 \\
2 & 1 & 1 \\
\end{array} \]
**Simple Scheduling**

**Round Robin**

**TIME**

0 1 2 3 ... n

1 Resource

0 1 2 3 ... n

0 1 2 RES
P0 5554 ... 1
0 1 2 3 ... n
(Time)
(HIGHER BETTER)

Priority Scheduling

- Relative?
- Pareto Efficiency?
- Envy Freedom?
- Sharing Incentive?

Resource
0 0 0 0
0 1 2 0 1
0 1 2 0 1

\[ \text{Resource} \]

\[ \text{Time} \]
0 1 2 3 ... n
\[ P^0 \overset{5}{5} \overset{5}{5} \overset{1}{1} \ldots \overset{1}{1} \]
(HIGHER BETTER)

- Sharing Incentive
- Envy Freedom
- Pareto Efficiency
- Relative
Do Round Robin or Priority Matter in Practice?

- Simple
- Low overhead

Lottery Scheduling

- Goals
  - Relative
  - Abstract/Homogenous: Algorithm is not resource specific?

8 cores 16 GB
Job | A | B | C | D
---|---|---|---|---
Ticket | 1 | 2 | 1 | 1
1 | 2 | 3 | 4

Time

Resource

- **Sharing Incentive?**
  - ✔️
- **Envy Freedom?**
  - ✔️
- Pareto Efficiency?  

Weights

\[ \frac{n_i}{N} = \sum n_i \]

1 - 14

Job

A | B | C | D

Tickets 2 | 7 | 3 | 2

2 | 9 | 12 | 14

\[ \frac{n_i}{N} \]

1 - 14

How Do We Fit Weights Into Fairness?

- Sharing Incentive?

\[ \frac{n_i}{\sum n_i} \]

- Envy Freedom?

- Pareto Efficiency?
**Algorithmic Complexity & Scalability**

---

**Completely Fair Scheduler**

<table>
<thead>
<tr>
<th>TIME</th>
<th>Job</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PRI</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>RUN 1</td>
<td>1.5</td>
<td>0.1</td>
<td>0.5</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>RUN 2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>
Algorithmic Complexity?

Problems with Multicore
Multi Resource Fairness

Switch Sheets
Problem with Different Ratios

Job 0 <1 core, 4 GB>

Job 1 <3 core, 1 GB>
Different Resources and Ratios

Memory and CPU demands for Hadoop at Facebook. (From DRF paper)
DRF Approach

Algorithm 1 DRF pseudo-code

\[ R = \langle r_1, \ldots, r_m \rangle \quad \triangleright \text{total resource capacities} \]
\[ C = \langle c_1, \ldots, c_m \rangle \quad \triangleright \text{consumed resources, initially 0} \]
\[ s_i \quad (i = 1..n) \quad \triangleright \text{user } i\text{'s dominant shares, initially 0} \]
\[ U_i = \langle u_{i,1}, \ldots, u_{i,m} \rangle \quad (i = 1..n) \quad \triangleright \text{resources given to user } i, \text{ initially 0} \]

pick user \( i \) with lowest dominant share \( s_i \)
\( D_i \leftarrow \text{demand of user } i\text{'s next task} \)
if \( C + D_i \leq R \) then
  \( C = C + D_i \quad \triangleright \text{update consumed vector} \)
  \( U_i = U_i + D_i \quad \triangleright \text{update } i\text{'s allocation vector} \)
  \( s_i = \max_{j=1}^{m} \{u_{i,j}/r_j\} \)
else
  return \quad \triangleright \text{the cluster is full} \)
end if
BENCHMARKING

PUT VS OVERHEAD

Basic Rules

- **Know what you are trying to measure**
- **What factors impact the quantity**
- **When possible**
  - Measure directly
- **If using indirect measurement**
  - Be careful!

The Problem at Hand

Baseline System B has throughput \( T \) with resources \( R \)
SYSTEM X adds features to B, and throughput goes to $T'$ with resource R.

CLAIM: CAN USE $T$, $T'$ TO DETERMINE "OVERHEAD" (COST)

WHAT WE ACTUALLY WANT TO KNOW:

HOW MUCH DOES R NEED TO INCREASE SO THAT X HAS TPUT T.

ASSUMPTION R processes $T'$ requests/sec

\[ \Rightarrow R \cdot \frac{T'}{T} \text{ PROCESSES } T \text{ REQ} \]

\[ \Rightarrow R \left( \frac{T - T'}{T} \right) < \text{ OVERHEAD?} \]

ASSUMPTIONS

1) B needs R to process T steps/sec

2) X = $T$, $T'$ = new Treq/sec
(2) A give \( \frac{RL}{I} \), can process this.