CS 744: DRF

Shivaram Venkataraman

Fall 2022
- Assignment 2 out! → ~ 2 weeks  Pytorch / Machine Learning
- Course Project
  - Project list by Oct 4  → seed ideas
  - Form groups and submit project bids by Oct 11
  - Assigned project by Oct 15
  - Introductions due Oct 25
SETTING: FAIR SHARING

Equal Share

Maximize the allocation for most poorly treated users

Max-Min Share

Maximize the minimum

3 clients in system
one resource

→ work conserving
= no resources are wanted

→ network

CPU lottery scheduling

Maximize the allocation for most poorly treated users

36%

50%

0%

50%

0%
SLOT-BASED MODEL

Slot: Fixed quantity of CPU and memory

Example: Hadoop MapReduce
Mapper: 2 CPU and 1 GB
Reducer: 1 CPU and 2 GB

Allocate in units of slots

2 Map slots in machine 1 to job 1
2 Reduce slots of job 2 in machine 1
MOTIVATION: MULTI RESOURCES

CPU intensive not memory intensive

majority of tasks

not based model

large range of resource demands

Reduce tasks memory intensive

Reduce tasks

maps

reduces

Per task memory demand (GB)

Pertask CPU demand (cores)
Users have a **demand vector**

\[ <2, 3, 1> \] means user’s task needs 2 R1, 3 R2, 1 R3

\[ \{ \text{CPU, Mem, Disk} \} \] example

Resources given in multiples of demand vector

i.e., users might get \[ <4, 6, 2> \]
PROPERTIES

Sharing Incentive
Users should not fare worse than having in cluster.

Strategy Proof
Users should not be able to lie to get a large share.

Pareto Efficiency
We cannot increase allocation of one user without decreasing others.

Envy free
Users should not desire allocation of another user.
**PROPERTIES**

**Sharing Incentive**
User is no worse off than a cluster with $\frac{1}{n}$ resources

**Strategy Proof**
User should not benefit by lying about demands

**Pareto Efficiency**
Not possible to increase one user without decreasing another

**Envy free**
User should not desire the allocation of another user
## DRF: APPROACH

<table>
<thead>
<tr>
<th>Dominant Resource</th>
<th>Dominant Share</th>
</tr>
</thead>
<tbody>
<tr>
<td>Resource user has the <strong>biggest</strong> share of</td>
<td>Fraction of the dominant resource user is allocated</td>
</tr>
<tr>
<td>Total: &lt;10 CPU, 4 GB&gt;</td>
<td>E.g., for User 1 this is 25% or 1/4</td>
</tr>
<tr>
<td>User 1: &lt;1 CPU, 1 GB&gt;</td>
<td></td>
</tr>
<tr>
<td>Dominant resource is <strong>memory</strong></td>
<td></td>
</tr>
</tbody>
</table>

\[
\frac{1}{10} < \frac{1}{4}
\]
# DRF: APPROACH

Equalize the dominant share of users

<table>
<thead>
<tr>
<th>User</th>
<th>Allocation</th>
<th>Dominant Share</th>
</tr>
</thead>
<tbody>
<tr>
<td>User1</td>
<td>&lt;0 CPU, 0 GB&gt; &lt;1 CPU, 4 GB&gt; &lt;2 CPU, 8 GB&gt; &lt;3 CPU, 12 GB&gt;</td>
<td>0; 4/18 = 2/9; 8/18 = 4/9; 12/18 = 2/3</td>
</tr>
<tr>
<td>User2</td>
<td>&lt;0 CPU, 0 GB&gt; &lt;3 CPU, 1 GB&gt;</td>
<td>0; 3/9</td>
</tr>
</tbody>
</table>

Total: <9 CPU, 18 GB>

User1: <1 CPU, 4 GB>

dom res: mem

User2: <3 CPU, 1 GB>

dom res: CPU
Total: <9 CPU, 18 GB>

User 1: <1 CPU, 4 GB> per task
<3 CPU, 12 GB> for 3 tasks
dom res: mem
dom share: 12/18 = 2/3

User 2: <3 CPU, 1 GB>
<6 CPU, 2 GB> for 2 tasks
dom res: CPU
dom share: 6/9 = 2/3
DRF ALGORITHM

Whenever there are available resources:
Schedule a task to the user with **smallest dominant share**

similar to max-min fairness but on dominant share
Algorithm 1 DRF pseudo-code

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

**pick** user \( i \) with lowest dominant share \( s_i \)

\( D_i \leftarrow \text{demand of user i's next task} \)

**if** \( C + D_i \leq R \)** then**

\[
\begin{align*}
C &= C + D_i \quad \triangleright \text{update consumed vector} \\
U_i &= U_i + D_i \quad \triangleright \text{update i's allocation vector} \\
s_i &= \max_{j=1}^{m} \{u_{i,j}/r_j\}
\end{align*}
\]

**else**

**return** \( \triangleright \text{the cluster is full} \)

**end if**
Asset Fairness: Equalize each user’s sum of resource shares

Consider total of 70 CPUs, 70 GB RAM
U1 needs <2 CPU, 2 GB RAM> per task
U2 needs <1 CPU, 2 GB RAM> per task

Asset Fair Allocation:
U1: 15 tasks: 30 CPU, 30 GB (Sum = 60)
U2: 20 tasks: 20 CPU, 40 GB (Sum = 60)
COMPARISON: ASSET FAIRNESS

Asset Fairness: Equalize each user's sum of resource shares

Violates Sharing Incentive

Consider total of 70 CPUs, 70 GB RAM
U1 needs <2 CPU, 2 GB RAM> per task
U2 needs <1 CPU, 2 GB RAM> per task

Sharing incentive?
Half of the cluster is 35 CPU, 35 GB RAM
U1: 17 tasks 34 CPU, 34 GB RAM \( \rightarrow \) better than 15 tasks with Asset Fairness
U2:
COMPARISON: CEEI

CEEI: Competitive Equilibrium from Equal Incomes

- Each user receives initially $1/n$ of every resource,
- Subsequently, each user can trade resources with other users in a perfectly competitive market
- Nash solution: Maximize product of utilities across users
COMPARISON: CEEI

Total: <9 CPU, 18 GB>  User1: <1 CPU, 4 GB>  User2: <3 CPU, 1 GB>

max (x \cdot y)
subject to
\begin{align*}
x + 3y & \leq 9 \\
4x + y & \leq 18
\end{align*}

x = 4.05

y = 1.62

\[ y = 1.62 \]

\[ x = 4.05 \]

Product of utilities
CEEI: STRATEGY PROOFNESS

Total: <9 CPU, 18 GB>

User2 Before:
CEEI: 55% CPU, 9% mem

Total: <9 CPU, 18 GB>
User1: <1 CPU, 4 GB>
User2: <3 CPU, 1 GB>
User2: <3 CPU, 2 GB>

\[ \max \ x \cdot y \]
\[ x + 3y \leq 9 \]
\[ 4x + 2y \leq 18 \]
## COMPARISON

<table>
<thead>
<tr>
<th>Property</th>
<th>Allocation Policy</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Asset</td>
</tr>
<tr>
<td>Sharing Incentive</td>
<td>✓</td>
</tr>
<tr>
<td>Strategy-proofness</td>
<td>✓</td>
</tr>
<tr>
<td>Envy-freeness</td>
<td>✓</td>
</tr>
<tr>
<td>Pareto efficiency</td>
<td>✓</td>
</tr>
<tr>
<td>Single Resource Fairness</td>
<td>✓</td>
</tr>
<tr>
<td>Bottleneck Fairness</td>
<td></td>
</tr>
<tr>
<td>Population Monotonicity</td>
<td>✓</td>
</tr>
<tr>
<td>Resource Monotonicity</td>
<td>✓</td>
</tr>
</tbody>
</table>

Table 2: Properties of Asset Fairness, CEEI and DRF.
SUMMARY

DRF: Dominant Resource Fairness
Allocation policy for scheduling
Provides multi-resource fairness
Ensures sharing incentive, strategy proofness
DISCUSSION

https://forms.gle/n97b12Qcs8Xv3C6L6
- Task requirements change and DRF adapts to the changes.
- There is a gap between Job 1 / Job 2 (discretization).
- Both jobs have dom. share 70.5.
- Task arrival / discretization?
What could be one workload / cluster scenario where DRF implemented on Mesos will NOT be optimal?

- Users diff between dominant resource & next dom. resource is small → will this lead to strategy proof violations??
- If you cannot fit the user with smallest dominant share, resource waste?
- Handling locality / affinity preferences is challenging
NEXT STEPS

Next Week: Machine Learning
Assignment 2 out!