Cooper: Task Colocation with Cooperative Games

Qiuyun Llull, Songchun Fan, Seyed Majid Zahedi, Benjamin C. Lee
Presented by: Qiuyun Llull

Duke University

HPCA – Feb 7, 2017
Task Colocation in Datacenters

Datacenters collocate applications to increase server utilization
Colocation Contention

Colocation interference can lead to performance degradation
System Setting

- Alvin, Ben, and Dan are working towards HPCA papers.
- They share a cluster and divide processors equally.
- Ben’s applications are memory intensive.
- Alvin and Dan’s applications are not memory intensive.
System Setting
Strategic Behavior

- Alvin, Ben, and Dan are strategic.
- Can smaller, separate clusters improve performance?
- Alvin and Dan share separate cluster to improve performance.
Strategic Behavior
Strategic Behavior

Without incentives, strategic users may...

- Bypass common management policy
- Migrate tasks for better colocations
- Procure private machines

Strategic action fragments cluster and harms efficiency
Prior Research

Pursues Performance
- Predicts contention quickly and accurately
- Colocates tasks for system performance
- Colocates tasks with complementary demands

Neglects Incentives
- Overlooks strategic behavior
- Fails to encourage users to colocate
Incentivizing Colocation

Stability
- No group of users break away to form separate system

Satisfied Preferences
- More users collocate with preferred tasks

Fair Attribution of Costs
- Users that contribute more to contention suffer higher losses
### Example

#### Preferences

- A: B > C > D
- B: A > C > D
- C: A > B > D
- D: C > A > B

#### Disutility (Slow-Down %)

<table>
<thead>
<tr>
<th>User</th>
<th>Co-Runner</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>A</td>
</tr>
<tr>
<td>A</td>
<td></td>
</tr>
<tr>
<td>B</td>
<td>1.8</td>
</tr>
<tr>
<td>C</td>
<td>0.0</td>
</tr>
<tr>
<td>D</td>
<td>3.8</td>
</tr>
</tbody>
</table>
Example

<table>
<thead>
<tr>
<th>User</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>1.5</td>
<td>4.9</td>
<td>9.3</td>
<td></td>
</tr>
<tr>
<td>B</td>
<td>1.8</td>
<td>3.9</td>
<td>12.7</td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>0.0</td>
<td>0.3</td>
<td>32.8</td>
<td></td>
</tr>
<tr>
<td>D</td>
<td>3.8</td>
<td>5.3</td>
<td>1.0</td>
<td></td>
</tr>
</tbody>
</table>

Preferences:
- A: B > C > D
- B: A > C > D
- C: A > B > D
- D: C > A > B

Performance:
- A
- D
- B
- C

Fairness:
- A
- B
- C
- D
A framework that incentivizes strategic users to collocate by providing desirable system outcomes:

• Stability
• Satisfied Preferences
• Fair Attribution of Costs
Agenda

• System Setting
• Incentivizing Colocation
• Cooper Colocation Framework
• Evaluation
Cooperative Game

- Strategic agents are users and tasks
- Utility is task performance
- Colocation preferences describe preferred co-runners
  - If $u(A, B) > u(A, C)$, then A prefers B over C
- Actions are -- participate or break away
Game Equilibrium

Colocations are stable when no group of users can improve their performance by changing colocation.
Cooper Framework

Agents
- Query Interface
- Preference Predictor
- Action Recommender

Coordinator
- System Profiler
- Colocation Policies
- Job Dispatcher

Machines

Users
Colocation Policies

Agents
- Query Interface
- Preference Predictor
- Action Recommender

Coordinator
- System Profiler
- Colocation Policies
- Job Dispatcher

Machines

Users
Matching people in life

It’s a Match!
You and Allison have liked each other.
Stable Matching

Algorithm partitions tasks into two sets
- Tasks in one set propose.
- Tasks in other set accepts, rejects.

Task updates co-runners
- Accept proposal if performance improves

Algorithm terminates when all tasks matched

Stable Matching

D>F>E

A

B

E>F>D

C

E>D>F

F

A>B>C

A>C>B

C>B>A
Stable Policies

Stable Marriage Random (SMR)
  • Partition tasks randomly

Stable Marriage Partition (SMP)
  • Partition tasks with domain-specific knowledge
  • Memory-intensive tasks propose

Stable Roommate (SR)
  • No partition
  • Any task proposes to any other.
Baseline Policies

Greedy (GR)
  • Colocate tasks to minimize performance loss

Complementary (CO)
  • Colocate tasks with complementary resource demands
Preference Predictor

Agents

Query Interface

Preference Predictor

Action Recommender

Coordinator

System Profiler

Colocation Policies

Job Dispatcher

Machines

Users
Preference Predictor

• Profile colocation performance with sparse samples

• Rate co-runners with profiles

• Predict ratings with collaborative filtering
  • Infer ratings based on task similarity
  • Suppose A: B > C and A is similar to D
  • Then D: B > C

• Construct preference list per task based on ratings
Action Recommender

Agents
- Query Interface
- Preference Predictor
- Action Recommender

Coordinator
- System Profiler
- Colocation Policies
- Job Dispatcher

Machines

Users

Duke Architecture
Action Recommender

• Assess assigned matches for each task

• Search preference list for better co-runners
  • Suppose X: A > B, and X matched to B
  • X messages A to suggest new match

• Recommend break away
  • Suppose A also prefers X over assigned match.
  • X, A should break away
Cooper Recap

- Agents:
  - Query Interface
  - Preference Predictor
  - Action Recommender

- Coordinator:
  - System Profiler
  - Colocation Policies
  - Job Dispatcher

- Machines
Agenda

• System Setting
• Incentivizing Colocation
• Cooper Colocation Framework
• Evaluation
Experimental Methods

Workloads
- PARSEC for multithreaded benchmarks
- Spark for task-parallel machine learning

System Measurements
- 10 nodes, each with 2 processors and 24 cores
- Two tasks share a processor each with half the cores

System Simulation
- 500 nodes with varied task populations
- Simulate colocations with system profiles
Fair Attribution of Costs

Tasks that contribute more to contention suffer higher penalties

x-axis sorts applications by memory intensity
Satisfied Preferences

More users collocate with preferred tasks.
Stability

Fewer users break away to form separate system
Performance

Stable colocations preserve system performance

Loss Relative to Stand-Alone

stable matches

GR
CO
SMP
SMR
SR
More in the paper …

Cooper Implementation

- Profiler and preference predictor
- Adapted matching algorithms
- Action recommender and job dispatcher

Cooperative Game Theory

- Shapely value for fair division
- Extending beyond pairs

Experimental Results

- Sensitivity to system scale and job mix
- Comprehensive policy comparisons
Conclusion

Cooperative Games for Shared Systems
- Formalize interactions between strategic users
- Incentivize user participation
- Enable fair task colocation

Management Desiderata
- Fair attribution of costs
- Satisfied preferences
- Stability

Fairness versus Performance
- Stable colocations satisfy more users
- Stable colocations preserve system performance
Thank you!