Carbyne

Altruistic Scheduling in Multi-Resource Clusters

Robert Grandl, Mosharaf Chowdhury, Aditya Akella, Ganesh Ananthanarayanan
We observe that:

- *Existing cluster schedulers focus on instantaneous fairness*
We observe that:

- Existing cluster schedulers focus on *instantaneous fairness*
- *Long-term fairness enables larger scheduling flexibility*
Performance of Cluster Schedulers

We observe that:

- Existing cluster schedulers focus on *instantaneous fairness*
- Long-term fairness enables larger scheduling flexibility
- Data-parallel jobs provide ample *opportunities for long-term optimizations*
We observe that:

- **Existing cluster schedulers focus on instantaneous fairness**
- **Long-term fairness enables larger scheduling flexibility**
- **Data-parallel jobs provide ample opportunities for long-term optimizations**

**Carbyne**

- 1.3x higher cluster efficiency;
- 1.6x lower average job completion time;
- near-perfect fairness
Scheduling in Data Analytics Clusters

Jobs
Scheduling in Data Analytics Clusters

Jobs → Cluster Wide Resource Manager
Scheduling in Data Analytics Clusters

Jobs

Cluster Wide Resource Manager

Inter-Job Scheduler

Cluster resources
Scheduling in Data Analytics Clusters

Jobs

Intra-Job Scheduler

Cluster Wide Resource Manager

Inter-Job Scheduler

Cluster resources
Scheduling in Data Analytics Clusters

Objectives:

#1: Fairness
Scheduling in Data Analytics Clusters

Objectives:

#1: Fairness
#2: Job Performance
#3: Cluster Efficiency
Scheduling in Data Analytics Clusters

Objectives:

#1: Fairness

#2: Job Performance

#3: Cluster Efficiency = is difficult!
Scheduling in Data Analytics Clusters

Objectives:

#1: Fairness
#2: Job Performance
#3: Cluster Efficiency

= is difficult!

TPC-DS workload on a 100-machine cluster
Scheduling in Data Analytics Clusters

Objectives:

1: Fairness
   - Dominant Resource Fairness
     - Max-min fair sharing across multiple dimensions

2: Job Performance

3: Cluster Efficiency
   - = is difficult!

TPC-DS workload on a 100-machine cluster
Scheduling in Data Analytics Clusters

Objectives:

#1: Fairness
- DRF [NSDI'11]

#2: Job Performance
- Shortest Job First (SJF)
  - Schedule jobs near completion first

#3: Cluster Efficiency
  = is difficult!

TPC-DS workload on a 100-machine cluster
Scheduling in Data Analytics Clusters

**Objectives:**

#1: Fairness
- DRF [NSDI’11]

#2: Job Performance
- Shortest Job First (SJF)

#3: Cluster Efficiency
- Tetris [SIGCOMM’14]
  - Efficiently pack resources

TPC-DS workload on a 100-machine cluster
Scheduling in Data Analytics Clusters

Objectives:

#1: Fairness
- DRF [NSDI’11]

#2: Job Performance
- Shortest Job First (SJF)

#3: Cluster Efficiency
- Tetris [SIGCOMM’14]

= is difficult!

TPC-DS workload on a 100-machine cluster
Scheduling in Data Analytics Clusters

Objectives:

#1: Fairness
- Jain's Fairness Index
  - Tetris
  - DRF
  - SJF

#2: Job Performance
- Inter-job fairness
- Avg. JCT (seconds)
  - Tetris: 1123
  - DRF: 1224
  - SJF: 769

#3: Cluster Efficiency
- Makespan (seconds)
  - Tetris: 4356
  - DRF: 5478
  - SJF: 6210

TPC-DS workload on a 100-machine cluster
Scheduling in Data Analytics Clusters

Objectives:

#1: Fairness
- DRF [NSDI’11]
- Shortest Job First (SJF)

#2: Job Performance
- Tetris [SIGCOMM’14]

#3: Cluster Efficiency

= is difficult!

Outperforms
- Preferred metric

Underperforms
- Secondary metrics

TPC-DS workload on a 100-machine cluster
Scheduling in Data Analytics Clusters

Objectives:

#1: Fairness  
- DRF [NSDI’11]  
- Jain’s Fairness Index: Tetris = 0.74, DRF = 0.86, SJF = 0.64

#2: Job Performance  
- Shortest Job First (SJF)  
- Makespan (seconds):
  - Tetris: 4356, DRF: 5478, SJF: 6210
  - Avg. JCT (seconds):
    - Tetris: 1123, DRF: 1224, SJF: 769

#3: Cluster Efficiency  
- Tetris [SIGCOMM’14]  
- Outperforms: Preferred metric
- Underperforms: Secondary metrics

= is difficult!

TPC-DS workload on a 100-machine cluster
Scheduling in Data Analytics Clusters

Is it possible to ensure fairness and still be competitive with the best approaches for the secondary metrics (job performance and cluster efficiency)?
Key observation

#1

Modern cluster schedulers focus on *instantaneous fairness* and force *short-term optimizations*
Key observation #1

Modern cluster schedulers focus on \textit{instantaneous fairness} and force \textit{short-term optimizations}
Key observation #1

Modern cluster schedulers focus on *instantaneous fairness* and force *short-term optimizations*.

Assumption:
- Know tasks demands and durations
Key observation #1

Modern cluster schedulers focus on **instantaneous fairness** and force **short-term optimizations**

**Assumption:**
- Know tasks demands and durations
Key observation #1

Modern cluster schedulers focus on \textit{instantaneous fairness} and force \textit{short-term optimizations}.

\begin{itemize}
  \item Assumption: Know tasks demands and durations
\end{itemize}
Key observation #1

Modern cluster schedulers focus on **instantaneous fairness** and force **short-term optimizations**

Assumption:
- Know tasks demands and durations
Key observation

#1

Modern cluster schedulers focus on \textit{instantaneous fairness} and force \textit{short-term optimizations}.

Traditional scheduling

Assumption:
- Know tasks demands and durations
Key observation #1

Modern cluster schedulers focus on 
*instantaneous fairness* 
and force *short-term optimizations*
Key observation

#1

Modern cluster schedulers focus on instantaneous fairness and force short-term optimizations.

Users care less for instantaneous fairness.

Assumption:
- Know tasks demands and durations
Key observation 
#1

Modern cluster schedulers focus on **instantaneous fairness** and force **short-term optimizations**

*Users care less for instantaneous fairness*

**Assumption:**
- Know tasks demands and durations
Key observation #1

Modern cluster schedulers focus on instantaneous fairness and force short-term optimizations.

Users care less for instantaneous fairness.

---

Leftover: donated unnecessary resources

Altruism: an action to contribute leftover resources
Key observation #1

Modern cluster schedulers focus on **instantaneous fairness** and force **short-term optimizations**

Users care less for **instantaneous fairness**

---

**Leftover:** donated unnecessary resources

**Altruism:** an action to contribute leftover resources

---

![Diagram](attachment:diagram.png)

- **Stage ID**
  - #Tasks x <Res. req> @Dur

- **Jobs**
  - Job 1
    - S0: 2 x <.08> @.5
    - S1: 3 x <.21> @1
    - S2: 1 x <.1> @.1
  - Job 2
    - S0: 2 x <.29> @1

- **Capacity**
  - 0.5
  - 1.0

- **Time**
  - 0
  - 1
  - 2

- **Avg. JCT:** 1.33 x better

---
Key observation

#2

Jobs in data analytics clusters have ample opportunities for altruism.
Key observation

#2

Jobs in data analytics clusters have ample opportunities for altruism.

What increases opportunities?
Key observation

#2

Jobs in data analytics clusters have ample opportunities for altruism.

What increases opportunities?

- Complex DAG structures
- Longer DAGs
Key observation #2

Jobs in data analytics clusters have ample opportunities for altruism

What increases opportunities?
- Complex DAG structures
- Longer DAGs

How much opportunities?
Key observation #2

Jobs in data analytics clusters have ample opportunities for altruism

What increases opportunities?
- Complex DAG structures
- Longer DAGs

How much opportunities?
50% of the time at least 20% of the resources can be used as leftover
Carbyne

Altruistic multi-resource scheduling technique
Carbyne

Altruistic multi-resource scheduling technique

Leftover
Carbyne

#1. How to maximize the amount of leftover resources?
1. How to maximize the amount of leftover resources?

2. How much leftover should contribute?

Altruistic multi-resource scheduling technique

Carbyne

- Inter-Job Scheduler
- Intra-Job Scheduler
- Leftover
Altruistic multi-resource scheduling technique

#1. How to maximize the amount of leftover resources?
#2. How much leftover should contribute?
#3. How to redistribute the leftover?
Maximize the amount of leftover resources
Maximize the amount of leftover resources

Instantaneous fairness elongates job completion time the most and increases altruism opportunities
Maximize the amount of leftover resources

Instantaneous fairness elongates job completion time the most and increases altruism opportunities

Carbyne uses DRF for inter-job scheduling
Maximize the amount of leftover resources

Instantaneous fairness elongates job completion time the most and increases altruism opportunities

Carbyne uses DRF for inter-job scheduling
- Any fair scheduler technique can be used
Maximize the amount of leftover resources

Instantaneous fairness elongates job completion time the most and increases altruism opportunities

Carbyne uses DRF for inter-job scheduling
- Any fair scheduler technique can be used

<table>
<thead>
<tr>
<th>Stage ID</th>
<th>#Tasks x &lt;Res. req&gt; @Dur</th>
</tr>
</thead>
<tbody>
<tr>
<td>S0</td>
<td>2 x &lt;.08&gt; @.5</td>
</tr>
<tr>
<td>S1</td>
<td>3 x &lt;.21&gt; @1</td>
</tr>
<tr>
<td>S2</td>
<td>1 x &lt;.1&gt; @.1</td>
</tr>
</tbody>
</table>
Maximize the amount of leftover resources

Instantaneous fairness elongates job completion time the most and increases altruism opportunities

Carbyne uses DRF for inter-job scheduling
- Any fair scheduler technique can be used

<table>
<thead>
<tr>
<th>Stage ID</th>
<th>#Tasks x &lt;Res. req&gt; @Dur</th>
</tr>
</thead>
<tbody>
<tr>
<td>S0</td>
<td>2 x &lt;.08&gt; @.5</td>
</tr>
<tr>
<td>S1</td>
<td>3 x &lt;.21&gt; @1</td>
</tr>
<tr>
<td>S2</td>
<td>1 x &lt;.1&gt; @.1</td>
</tr>
</tbody>
</table>
How much leftover to contribute?
How much leftover to contribute?

Traditional scheduling to compute expected completion time
How much leftover to contribute?

Traditional scheduling to compute expected completion time
How much leftover to contribute?

Traditional scheduling to compute expected completion time
How much leftover to contribute?

Traditional scheduling to **compute expected completion time**
How much leftover to contribute?

Traditional scheduling to compute expected completion time

Scheduling in the future from finish to current time

<table>
<thead>
<tr>
<th>Stage ID</th>
<th>#Tasks x &lt;Res. req&gt; @Dur</th>
</tr>
</thead>
<tbody>
<tr>
<td>S0</td>
<td>2 x &lt;.08&gt; @.5</td>
</tr>
<tr>
<td>S1</td>
<td>3 x &lt;.21&gt; @1</td>
</tr>
<tr>
<td>S2</td>
<td>1 x &lt;.1&gt; @.1</td>
</tr>
</tbody>
</table>

Job 1

<table>
<thead>
<tr>
<th>Stage</th>
<th>#Tasks</th>
<th>&lt;Res. req&gt;</th>
<th>@Dur</th>
</tr>
</thead>
<tbody>
<tr>
<td>S0</td>
<td>2</td>
<td>&lt;.08&gt;</td>
<td>.5</td>
</tr>
<tr>
<td>S1</td>
<td>3</td>
<td>&lt;.21&gt;</td>
<td>1</td>
</tr>
<tr>
<td>S2</td>
<td>1</td>
<td>&lt;.1&gt;</td>
<td>1</td>
</tr>
</tbody>
</table>

Job 2

<table>
<thead>
<tr>
<th>Stage</th>
<th>#Tasks</th>
<th>&lt;Res. req&gt;</th>
<th>@Dur</th>
</tr>
</thead>
<tbody>
<tr>
<td>S0</td>
<td>2</td>
<td>&lt;.29&gt;</td>
<td>1</td>
</tr>
</tbody>
</table>

Time

Capacity

Move into future

Fair Allocation

JCT Job 1: 2.1
JCT Job 2: 2.0
How much leftover to contribute?

Traditional scheduling to **compute expected completion time**

**Scheduling in the future** from finish to current time

**Stage ID**
- Job 1: 2 x <.08> @.5
- Job 1: 3 x <.21> @1
- Job 2: 1 x <.1> @.1

**Capacity**
- 0
- 0.5
- 1.0

**Time**
- 0
- 1
- 2

**Move into future**

**Fair Allocation**

**JCT Job 1: 2.1**
**JCT Job 2: 2.0**
How much leftover to contribute?

Traditional scheduling to compute expected completion time

Scheduling in the future from finish to current time
How much leftover to contribute?

Traditional scheduling to **compute expected completion time**

**Scheduling** in the future from finish to current time

Donate leftover resources through altruism

---

[Diagram showing scheduling and resource allocation]
How much leftover to contribute?

Traditional scheduling to compute expected completion time

Scheduling in the future from finish to current time

Donate leftover resources through altruism
How to redistribute the leftover resources?

<table>
<thead>
<tr>
<th>Stage ID</th>
<th>#Tasks x &lt;Res. req&gt; @Dur</th>
</tr>
</thead>
<tbody>
<tr>
<td>S0</td>
<td>2 x &lt;.08&gt; @.5</td>
</tr>
<tr>
<td>S1</td>
<td>3 x &lt;.21&gt; @1</td>
</tr>
<tr>
<td>S2</td>
<td>1 x &lt;.1&gt; @.1</td>
</tr>
</tbody>
</table>

**Job 1**
- Time
- Capacity
- Stage ID
- S0: 2 tasks @0.5
- S1: 3 tasks @1
- S2: 1 task @0.1

**Job 2**
- Time
- Capacity
- Stage ID
- S0: 2 tasks @0.5
- S1: 3 tasks @1
- S2: 1 task @0.1

**Fair Allocation**
- Total: 0.50

**Leftover**
- JCT Job 1: 2.1
- JCT Job 2: 2.0
How to redistribute the leftover resources?

Goal 1: Improve average JCT
How to redistribute the leftover resources?

Goal 1: Improve average JCT

Goal 2: Maximize efficiency

Goals 1 and 2 can be interchanged
How to redistribute the leftover resources?

Goal 1: Improve average JCT

- Schedule jobs closest to completion time first

Goal 2: Maximize efficiency

Goals 1 and 2 can be interchanged

<table>
<thead>
<tr>
<th>Stage ID</th>
<th>#Tasks x &lt;Res. req&gt; @Dur</th>
</tr>
</thead>
<tbody>
<tr>
<td>S0</td>
<td>2 x &lt;.08&gt; @.5</td>
</tr>
<tr>
<td>S1</td>
<td>3 x &lt;.21&gt; @1</td>
</tr>
<tr>
<td>S2</td>
<td>1 x &lt;.1&gt; @.1</td>
</tr>
</tbody>
</table>

Job 1

<table>
<thead>
<tr>
<th>Stage ID</th>
<th>#Tasks x &lt;Res. req&gt; @Dur</th>
</tr>
</thead>
<tbody>
<tr>
<td>S0</td>
<td>2 x &lt;.29&gt; @1</td>
</tr>
<tr>
<td>S1</td>
<td>S0</td>
</tr>
<tr>
<td>S1</td>
<td>S1</td>
</tr>
<tr>
<td>S1</td>
<td>S2</td>
</tr>
</tbody>
</table>

Job 2

<table>
<thead>
<tr>
<th>Stage ID</th>
<th>#Tasks x &lt;Res. req&gt; @Dur</th>
</tr>
</thead>
<tbody>
<tr>
<td>S0</td>
<td>2 x &lt;.8&gt; @1</td>
</tr>
<tr>
<td>S1</td>
<td>3 x &lt;.21&gt; @1</td>
</tr>
<tr>
<td>S2</td>
<td>1 x &lt;.1&gt; @.1</td>
</tr>
</tbody>
</table>

Fair Allocation

Leftover

Total: 0.50

JCT Job 1: 2.1
JCT Job 2: 2.0
How to redistribute the leftover resources?

Goal 1: Improve average JCT
- *Schedule jobs closest to completion time first*

Goal 2: Maximize efficiency

Goals 1 and 2 can be interchanged
How to redistribute the leftover resources?

Goal 1: Improve average JCT
- Schedule jobs closest to completion time first

Goal 2: Maximize efficiency
- Pack as many unscheduled tasks are possible

Goals 1 and 2 can be interchanged

---

Goal 1: Improve average JCT

Stage ID

<table>
<thead>
<tr>
<th>Stage ID</th>
<th>Tasks</th>
<th>Resource req</th>
<th>Duration</th>
</tr>
</thead>
<tbody>
<tr>
<td>S0</td>
<td>2</td>
<td>.08</td>
<td>.5</td>
</tr>
<tr>
<td>S1</td>
<td>3</td>
<td>.21</td>
<td>1</td>
</tr>
<tr>
<td>S2</td>
<td>1</td>
<td>.1</td>
<td>.1</td>
</tr>
</tbody>
</table>

**Total: 0.21**

JCT Job 1: 2.1
JCT Job 2: 1.0
How to redistribute the leftover resources?

Goal 1: Improve average JCT
- Schedule jobs closest to completion time first

Goal 2: Maximize efficiency
- Pack as many unscheduled tasks are possible

Goals 1 and 2 can be interchanged

Goal 1: Improve average JCT

Goal 2: Maximize efficiency

Goals 1 and 2 can be interchanged
Putting it all together

We saw

Increase leftover via Inter-Job Scheduling
  ▪ Adopting best fair schedulers
Putting it all together

We saw

*Increase leftover via Inter-Job Scheduling*
  - Adopting best fair schedulers

*Compute leftover via Intra-Job Scheduling*
Putting it all together

We saw

**Increase leftover via Inter-Job Scheduling**
- Adopting best fair schedulers

**Compute leftover via Intra-Job Scheduling**

**Leftover redistribution**
- Improve JCT and cluster efficiency
Putting it all together

We saw

Increase leftover via Inter-Job Scheduling
- Adopting best fair schedulers

Compute leftover via Intra-Job Scheduling

Leftover redistribution
- Improve JCT and cluster efficiency

Other things in the paper

Bounding altruism with $P(Altruism)$
Putting it all together

We saw

Increase leftover via Inter-Job Scheduling
- Adopting best fair schedulers

Compute leftover via Intra-Job Scheduling

Leftover redistribution
- Improve JCT and cluster efficiency

Other things in the paper

Bounding altruism with $P(Altruism)$

Resource estimation
Putting it all together

**We saw**

- **Increase leftover via Inter-Job Scheduling**
  - Adopting best fair schedulers

- **Compute leftover via Intra-Job Scheduling**

- **Leftover redistribution**
  - Improve JCT and cluster efficiency

**Other things in the paper**

- Bounding altruism with $P(Altruism)$
- Resource estimation
- Data locality
- Straggler mitigation
- Task failures
Evaluation

- Implemented in Yarn and Tez
- 100 machine cluster deployment
- Replay Bing / Facebook traces and TPC-DS / TPC-H workloads
Fairness vs. Performance vs. Efficiency

Inter-job fairness

Job Performance

Cluster efficiency

Jain's Fairness Index

Avg. JCT (seconds)

Makespan (seconds)

Cluster efficiency
Fairness vs. Performance vs. Efficiency

Comparable performance with best approaches in each metric
Fairness vs. Performance vs. Efficiency

- Jain's Fairness Index
  - Tetris: 0.74
  - DRF: 0.86
  - SJF: 0.64
  - Carbyne: 0.81

- Avg. JCT (seconds)
  - Tetris: 1123
  - DRF: 1224
  - SJF: 769
  - Carbyne: 814

- Makespan (seconds)
  - Tetris: 4356
  - DRF: 5478
  - SJF: 6210
  - Carbyne: 4492

Inter-job fairness: Comparable performance with best approaches in each metric

Job Performance

Cluster efficiency

Gains from Altruism helps in long run
Job Performance

Snapshot of the execution of a TPC-DS query

Gains from
Altruism helps in long run
Job Performance

Snapshot of the execution of a TPC-DS query

- **DRF**
  - Tasks from a DRF allocation

- **Carbyne w/o leftover**
  - Only running tasks from being altruistic

- **Carbyne**
  - Tasks from being altruistic and from leftover redistribution

**Gains from**

- Altruism helps in long run!
Job Performance

Snapshot of the execution of a TPC-DS query

- **Carbyne w/o leftover**
  - Only running tasks from being altruistic

- **Carbyne**
  - Tasks from being altruistic and from leftover redistribution

- **DRF**
  - Tasks from a DRF allocation

**Low resource contention**
- DRF takes greedy decisions

**Gains from**
- Altruism helps in long run
Job Performance

![Graph showing job performance over time with different strategies: Carbyne w/o leftover, Carbyne, and DRF.](image)

**Snapshot of the execution of a TPC-DS query**

### Low resource contention
- DRF takes greedy decisions
- Approximates DRF allocation due to leftover

### Gains from
- Altruism helps in long run

**DRF**
- Tasks from a DRF allocation

**Carbyne w/o leftover**
- Only running tasks from being altruistic

**Carbyne**
- Tasks from being altruistic and from leftover redistribution
**Job Performance**

*Snapshot of the execution of a TPC-DS query*

- **Carbyne w/o Leftover**
- **Carbyne**
- **DRF**

**DRF**
- Tasks from a DRF allocation

**Carbyne w/o leftover**
- Only running tasks from being altruistic

**Carbyne**
- Tasks from being altruistic and from leftover redistribution

**High resource contention**
- DRF progress is slowed down

**Gains from**
- Altruism helps in long run
Job Performance

**High resource contention**
- DRF progress is slowed down
- Carbyne progress faster due to receiving leftover

**Gains from**
Altruism helps in long run

**DRF**
- Tasks from a DRF allocation

**Carbyne w/o leftover**
- Only running tasks from being altruistic

**Carbyne**
- Tasks from being altruistic and from leftover redistribution
Job Performance

Snapshot of the execution of a TPC-DS query

- Carbyne w/o Leftover
- Carbyne
- DRF

# Running Tasks vs. Time (Seconds)

Fraction of Jobs vs. Job Completion Time (Seconds)
Job Performance

Performance

- Better jobs completion time
Job Performance

Performance
- Better jobs completion time
- 16% jobs slowed down
  - 4% only by more than 0.8x
**Job Performance**

- Better jobs completion time
- 16% jobs slowed down
- 4% only by more than 0.8x
Job Performance

**Performance**
- Better jobs completion time
- 16% jobs slowed down
  - 4% only by more than 0.8x

**Which jobs slow down?**
- Longer jobs
**Job Performance**

Performance
- Better jobs completion time
- 16% jobs slowed down
  - 4% only by more than 0.8x

Which jobs slow down?
- Longer jobs
- No bias towards shorter jobs

*Snapshot of the execution of a TPC-DS query*
**Performance**

- Better jobs completion time
- 16% jobs slowed down
  - 4% only by more than 0.8x

**Which jobs slow down?**

- Longer jobs
- No bias towards shorter jobs
- Most jobs benefit from leftover
Impact of a Better Intra-Job Scheduler

Default intra-job scheduler
- Tetris
Impact of a Better Intra-Job Scheduler

Default intra-job scheduler
- Tetris

![Graph showing the impact of Carbyne+Tetris on the fraction of jobs versus factor of improvement (w.r.t. DRF).]
Impact of a Better Intra-Job Scheduler

Default intra-job scheduler
- Tetris
- Limited view of the job’s DAG

Factor of Improvement (w.r.t. DRF)

Fraction of Jobs

Carbyne+Tetris

0.5 3 5.5 8 10.5 13

0 0.2 0.4 0.6 0.8 1

Factor of Improvement (w.r.t. DRF)
Impact of a Better Intra-Job Scheduler

Default intra-job scheduler
- Tetris
- Limited view of the job’s DAG

Better intra-job scheduler
- Graphene – DAG-wide view

![Graph showing Fraction of Jobs vs. Factor of Improvement (w.r.t. DRF)]

- Carbyne + Tetris
Impact of a Better Intra-Job Scheduler

**Default intra-job scheduler**
- Tetris
- Limited view of the job’s DAG

**Better intra-job scheduler**
- Graphene – DAG-wide view

![Graph showing fraction of jobs vs factor of improvement](image)
Impact of a Better Intra-Job Scheduler

Default intra-job scheduler
- Tetris
- Limited view of the job’s DAG

Better intra-job scheduler
- Graphene – DAG-wide view
- Extracts more leftover
- Further increase performance
Carbyne

- Increase Leftover via Inter-Job Scheduling
- Maximize Leftover for Individual Jobs
- Redistribution via Leftover Scheduling

Fairness

Performance Efficiency
- **Long-term altruistic view** of Carbyne *outperforms* existing cluster schedulers which focus on instantaneous fairness.
- **Long-term altruistic view** of Carbyne outperforms existing cluster schedulers which focus on instantaneous fairness

- Implemented inside YARN and Tez
Carbyne

- **Long-term altruistic view** of Carbyne outperforms existing cluster schedulers which focus on instantaneous fairness.

- Implemented inside YARN and Tez.

- Performance comparable with best approaches in terms of fairness and job completion time and cluster efficiency.
Backup slides
Increasing levels of altruism increase performance.

Comparable when $P(\text{Altruism}) = 0$
Impact of Misestimations

- Consistently better performance
- Comparable when resources are underestimated
More contention increases the need for carefully rearranging tasks

Too much contention saturates the cluster; not much room for leftover allocations
Even in online case, Carbyne comes closely to the best metric in each metric
Data Locality vs. Straggler Mitigation vs. Task Failures

**Data Locality**
- Altruistically giving up resources for data-local task may have adverse effects
- An altruistically delayed data-local task is likely to find data locality when it is eventually scheduled

**Straggler Mitigation**
- Likely to prioritize speculative tasks during leftover scheduling because it selects jobs in the SRTF order

**Handling Task Failures**
- Does not distinguish between new and restarted tasks
- In case of task failures, it has to recalculate the expected completion time for the job