Managing Tiny Tasks for Data-Parallel, Subsampling Workloads

Sundeep Kambhampati, Jaimie Kelley, Christopher Stewart, William C. L. Stewart, Rajiv Ramnath
The Ohio State University, Nationwide Children's Hospital
Accuracy versus Speed

- Data is growing faster than processor clock rates.
  - Discrete objects from the real world: DNA samples, event clicks, user reviews, communications, etc.

- **Subsampling workloads** process only a portion of a data set (i.e., random sample).

- Subsampling speeds up data processing by doing less work but decreases accuracy.
Motivating Example: EAGLET

DNA Samples
1 strand = 1 family

Base Pair Subsamples

Efficient Analysis of Genetic Linkage: Estimation and Testing

Gene to Inspect

Gene relevance to medical condition?

T-test

Reprinted from the U.S. National Library of Medicine, National Institutes of Health, Bethesda, MD.
Subsampling on Data-Parallel Platforms

- Discrete data objects partition across nodes.
- For mean and other stats, objects are processed independently.
- Subsampling on data-parallel platforms:
  - 1. Partition samples across nodes.
  - 2. Map tasks randomly subsample partitions at runtime to compute statistics.
Task Sizing

- Map task size is configurable
  - Affects cache miss rate
  - Affects startup costs
- How to configure the size of a map task to reduce latency.
Task Sizing Example: EAGLET

Efficient Analysis of Genetic Linkage: Estimation and Testing

Gene to Inspect

Subsampling (At Runtime)

Gene relevance to medical condition?

T-test
Problem: Size tasks to minimize latency?

Job latency is a function of task size.

<table>
<thead>
<tr>
<th>Task Size</th>
<th>Task startup overhead</th>
<th>Cache miss rate</th>
<th>Job Latency</th>
</tr>
</thead>
<tbody>
<tr>
<td>All Data</td>
<td></td>
<td></td>
<td>All Data</td>
</tr>
<tr>
<td>One Unit</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Our Approach: Tiny Tasks

• What is a tiny task?
  - ↑ task size → cache miss rate increases sharply

• Finding sharp increases in cache miss rate:
  - Workload: EAGLET, 230 MB, 400 individuals
  - Hardware: Intel Sandy Bridge
    - 6 dual cores, 1.5 MB L2, 15 MB L3
  - Monitoring: Oprofile
Cache Miss Increases

![Graph showing cache miss rate per instruction vs. task size (MB). The x-axis represents task size in MB, ranging from 0.04 to 40, while the y-axis represents cache miss rate per instruction, ranging from 0.0001 to 1.0. The graph includes a line with two starred points, indicating a significant increase in cache miss rate as task size increases.]
Overheads

- **Startup costs**
  - Workload: Hello World, tasks == map slots

- **Hardware: 72 core cluster**
  - 6 dual-core Intel Sandy Bridge processors

- **Subsampling data-parallel tasks**
- **Task sizing for subsampling workloads**
- **Pressures of scheduling tiny tasks**
- **Our platform for task sizing**
Overheads: Platforms

Hadoop is widely used in practice for map reduce workloads.

<table>
<thead>
<tr>
<th>Codename</th>
<th>Core</th>
<th>Task-level Failures</th>
<th>Full Distributed File System</th>
<th>Java</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vanilla Hadoop</td>
<td>Hadoop</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Job-level Hadoop</td>
<td>Hadoop</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Lite Hadoop</td>
<td>Hadoop</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>BashReduce</td>
<td>Unix Utilities</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>

BashReduce is a very lightweight implementation of map reduce.
Overheads: Startup Costs

Startup Costs: **BashReduce** < Hadoop
Task Sizing Algorithm

1. Pick random samples and run the tiniest task
2. Collect misses
3. Loop, increasing task size and using new random samples
4. Keep comparing miss rates
5. Use the task size right before a large increase in cache miss rate.

• Subsampling data-parallel tasks
• Task sizing for subsampling workloads
• Pressures of scheduling tiny tasks
• Our platform for task sizing
Cache Miss Increases

![Graph showing cache miss rate per instruction against task size (MB). The x-axis represents task size in MB, ranging from 0.04 to 40, and the y-axis represents cache miss rate per instruction, ranging from 0.0001 to 1.0000. The graph shows an increasing trend in cache miss rate as task size increases, with a significant increase at around 4 MB. A yellow star indicates a specific task size where the cache miss rate is unusually high.](image-url)
# Task Sizing: Workloads

<table>
<thead>
<tr>
<th>Workload</th>
<th>EAGLET</th>
<th>Netflix (High confidence)</th>
<th>Netflix (Low Confidence)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>Genetic study on Bi-Polar Disorder</td>
<td>User movie ratings</td>
<td>User movie ratings</td>
</tr>
<tr>
<td>BashReduce Task Sizing</td>
<td>2.5 MB</td>
<td>1 MB</td>
<td>1 MB</td>
</tr>
<tr>
<td>Tiniest Task Sizing</td>
<td>588.8 KB</td>
<td>118 KB</td>
<td>118 KB</td>
</tr>
<tr>
<td>Largest Task Sizing</td>
<td>1 TB</td>
<td>2 GB</td>
<td>2 GB</td>
</tr>
</tbody>
</table>

1: Xeon 12 cores/node, 2.0 GHz, 15 MB L2, 32 GB Memory  
2: Xeon 12 cores/node, 2.3 GHz, 15 MB L2, 32 GB Memory  
3: Opteron 32 cores/node, 2.3 GHz, 32 MB L2, 64 GB Memory  
Largest Job Run: **Eaglet, with 1 TB**
Our Task Sizing has **consistent higher throughput**. Type 1 hardware
Task Sizing: Evaluation

Type 2 Hardware

![Graph showing throughput (MB per sec) vs job size on BTS. Each line represents different number of cores: 72 Cores (red), 60 Cores (blue), 36 Cores (dashed), 12 Cores (black).](image)

Throughput (MB per sec)

Job Size on BTS

Type 2 Hardware
Task Sizing: Evaluation

% of Peak Throughput

% of Running Time

Service Level Objective Bound

Type 2 Hardware
Conclusion

1. Subsampling workloads benefit from task sizing to reduce cache miss rates and runtime costs.

2. We measure startup costs on tiny tasks for existing data-parallel platforms.

3. We implemented an algorithm to size tasks at sharp increases in cache miss rate within the BashReduce scheduler to reduce runtime overheads.

4. We validate our improved BashReduce against existing data-parallel platforms across multiple workloads.