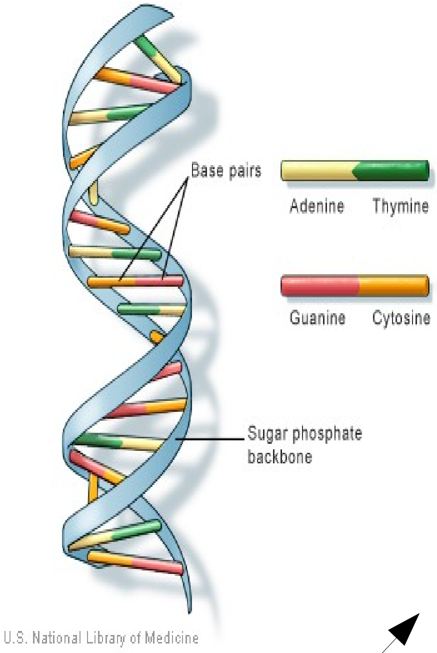
Managing Tiny Tasks for Data-Parallel, Subsampling Workloads

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

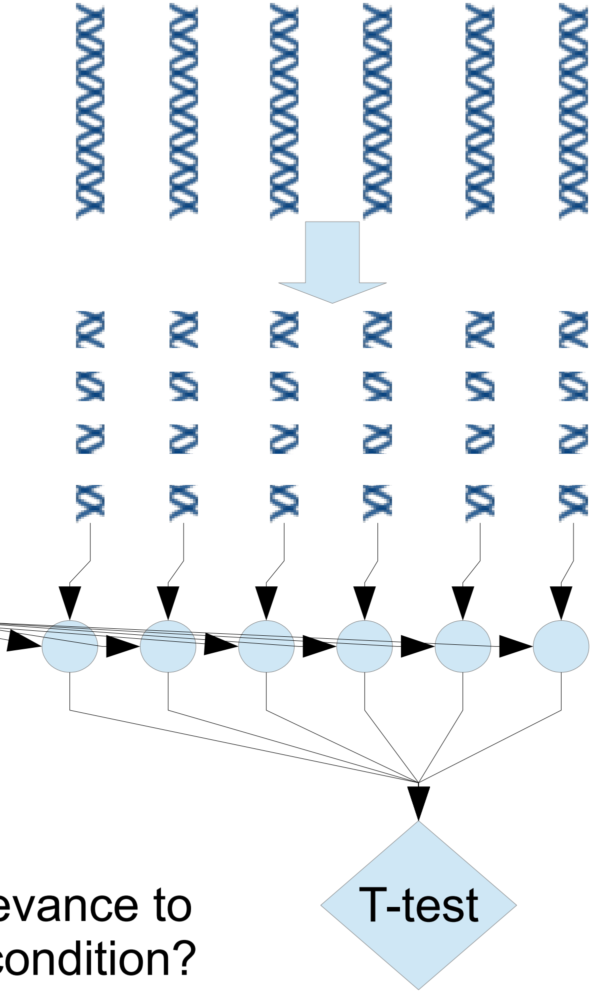


DNA Samples
1 strand = 1 family

Base Pair
Subsamples

Efficient **A**nalysis of
Genetic **L**inkage:
Estimation and **T**esting

Gene to
Inspect



Gene relevance to
medical condition?

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.
  - 3. Reduce tasks compute statistics using map results.

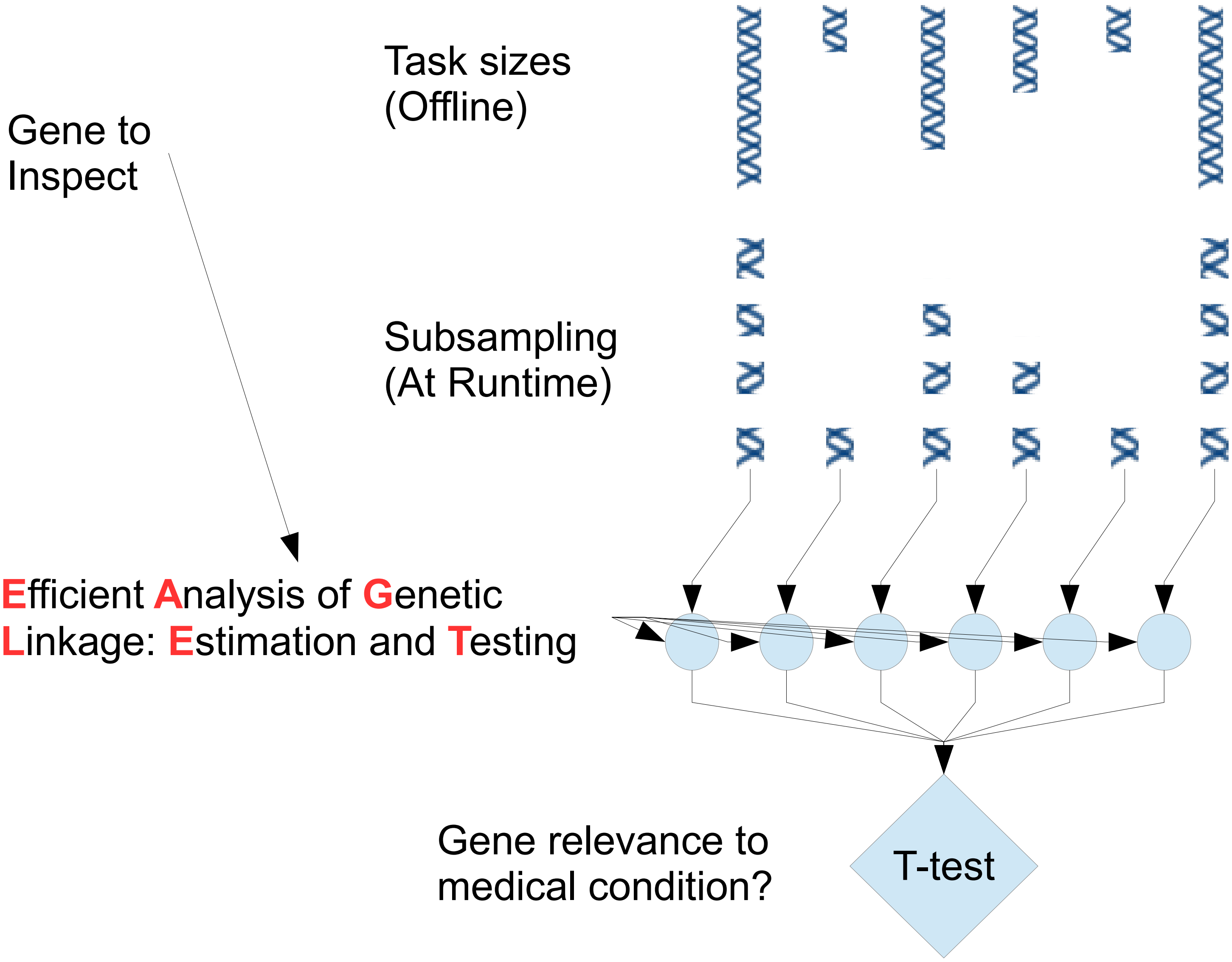
Task Sizing

- Map task size is configurable
  - Affects cache miss rate
  - Affects startup costs
- **How to configure the size of a map task to reduce latency.**

- Subsampling data-parallel tasks

- Task sizing for subsampling workloads
- Pressures of scheduling tiny tasks
- Our platform for task sizing

Task Sizing Example: EAGLET



Task sizes (Offline)

Gene to Inspect

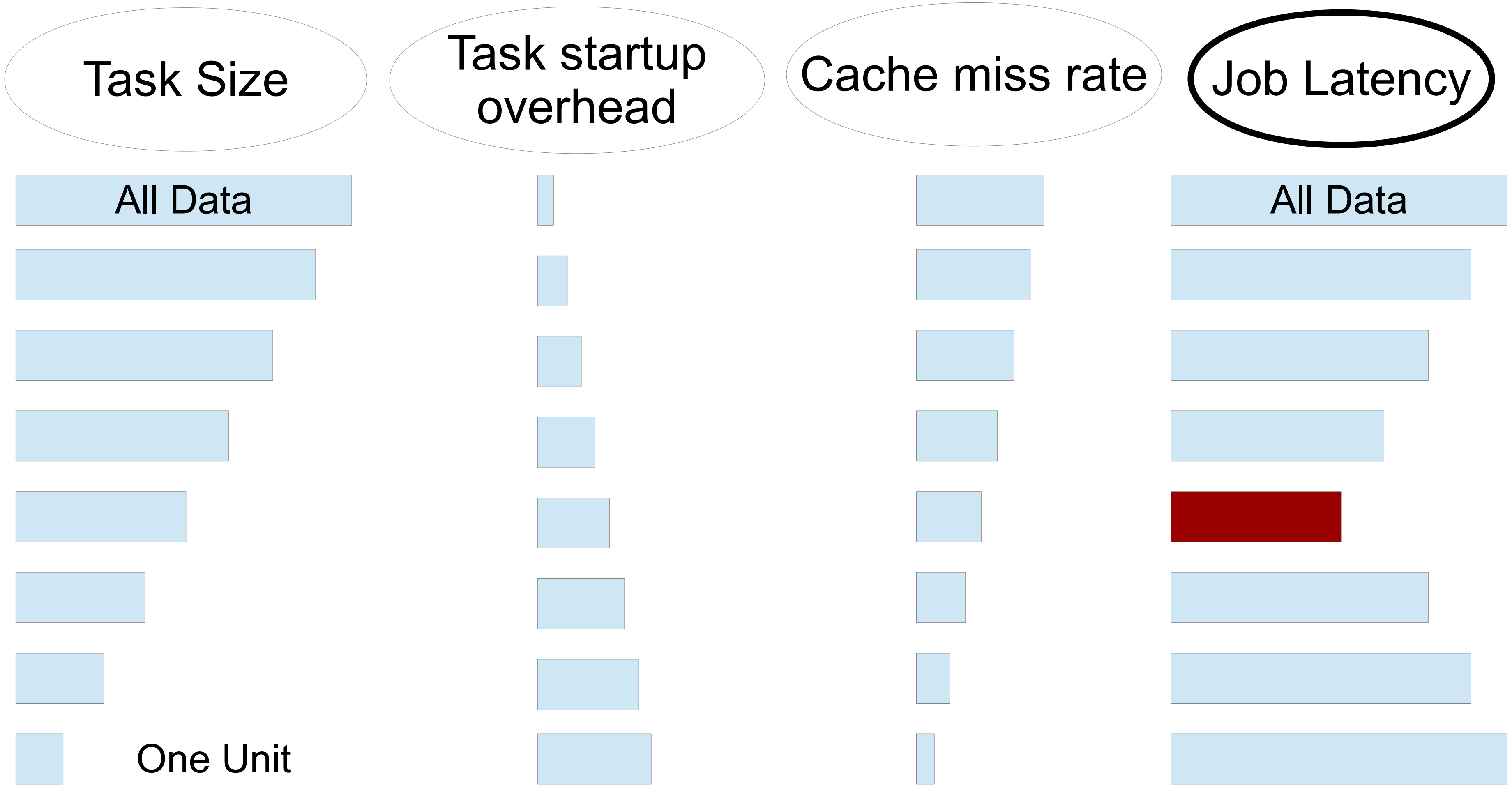
Subsampling (At Runtime)

Efficient Analysis of Genetic Linkage: Estimation and Testing

Gene relevance to medical condition?

Problem: Size tasks to minimize latency?

Job latency is a function of task size.

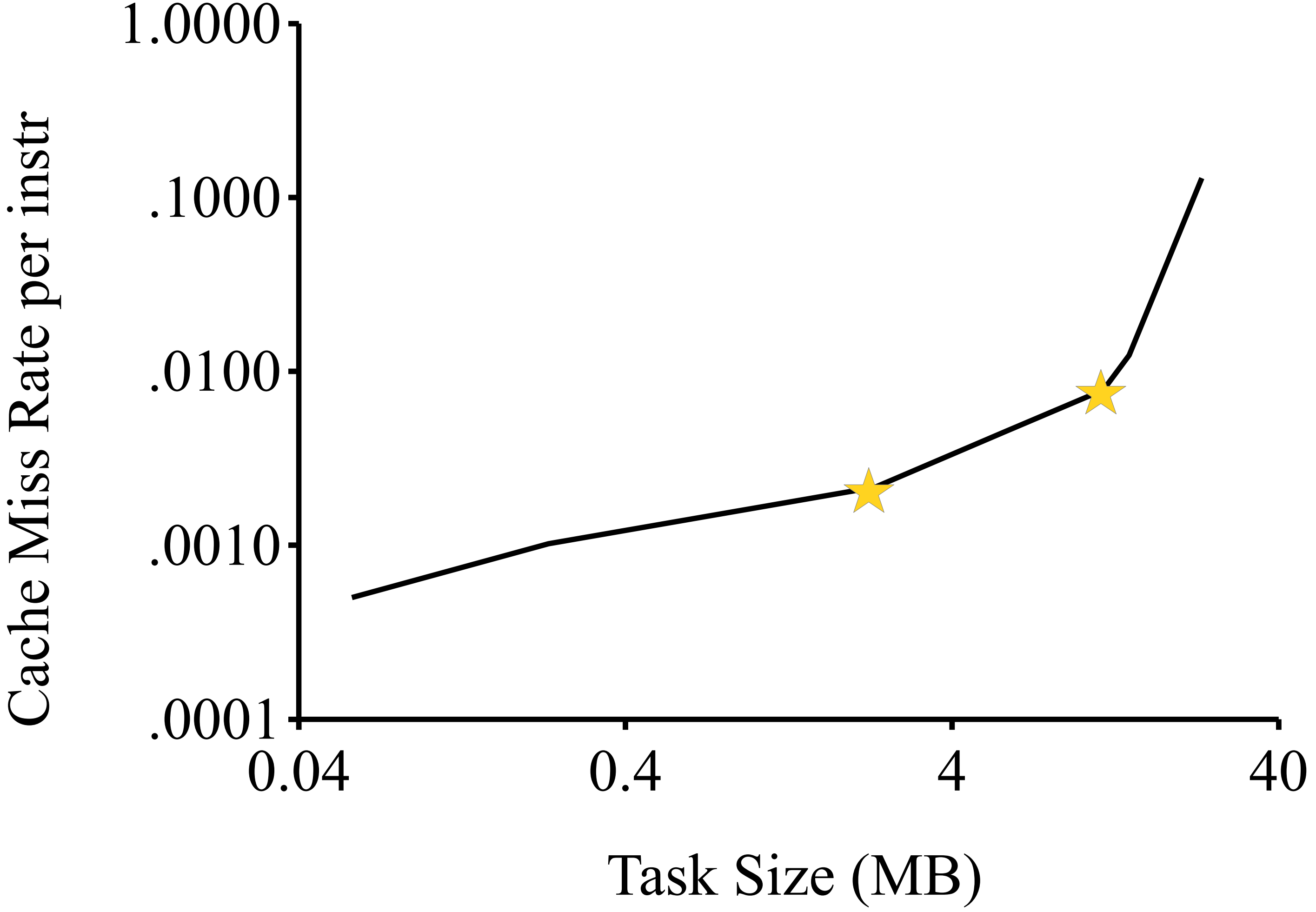


Our Approach: Tiny Tasks

- 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



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

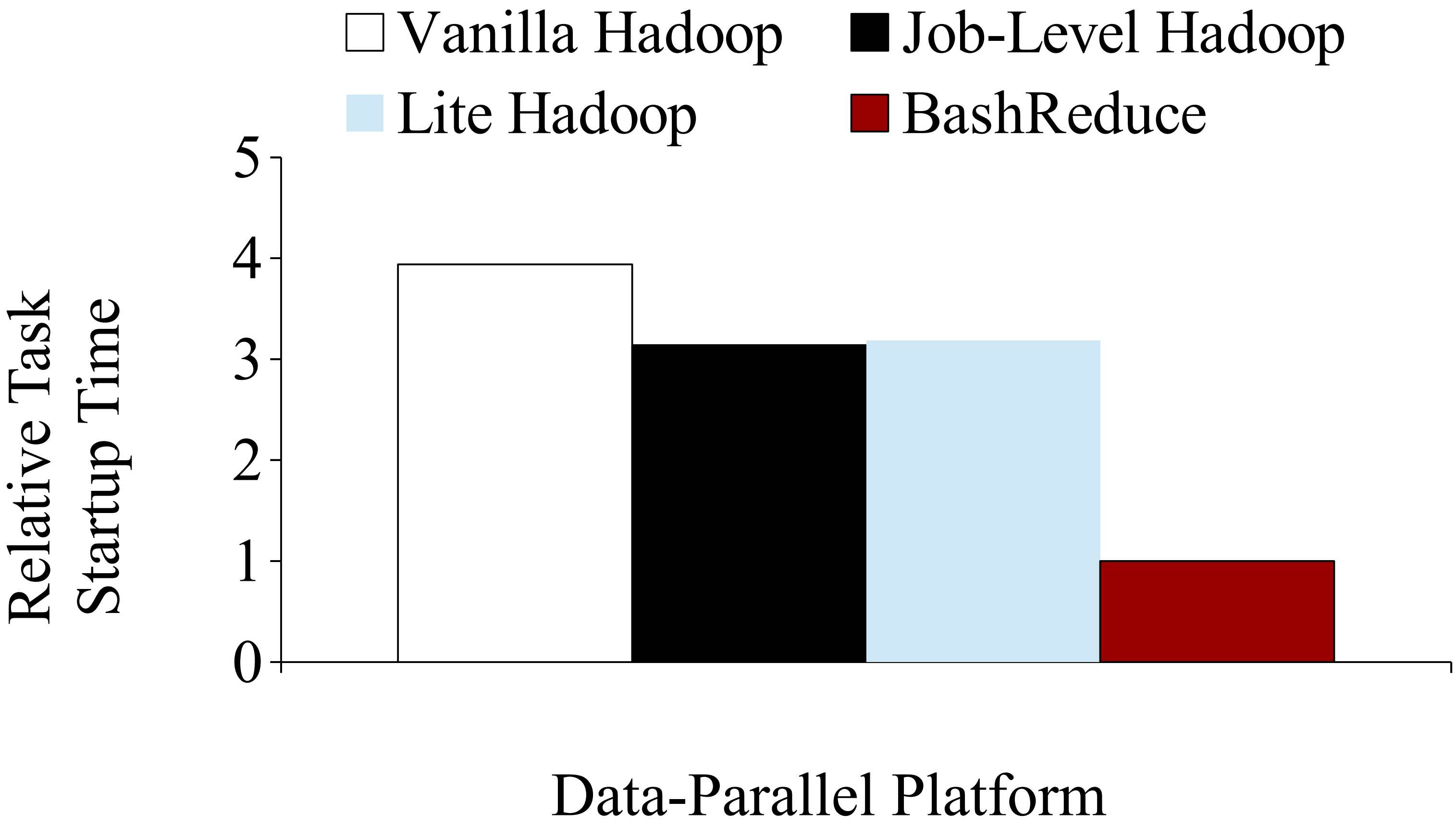
Overheads: Platforms

Hadoop is widely used in practice for map reduce workloads.

Codename	Core	Task-level Failures	Full Distributed File System	Java
Vanilla Hadoop	Hadoop	Yes	Yes	Yes
Job-level Hadoop	Hadoop	No	Yes	Yes
Lite Hadoop	Hadoop	No	No	Yes
BashReduce	Unix Utilities	No	No	No

BashReduce is a very lightweight implementation of map reduce.

Overheads: Startup Costs



Startup Costs: **BashReduce** < Hadoop

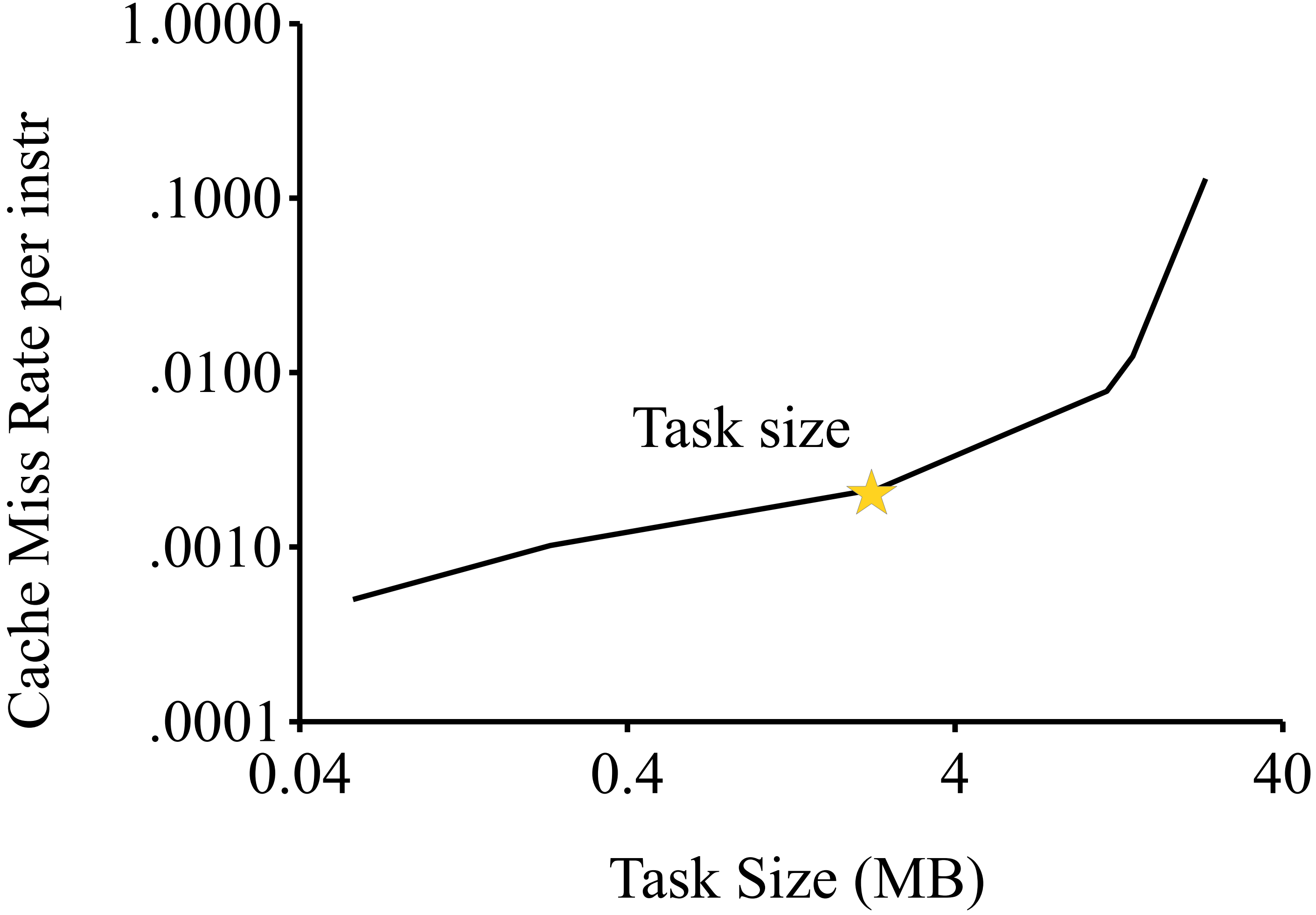
Task Sizing Algorithm

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



Task Sizing: Workloads

Workload	EAGLET	Netflix (High confidence)	Netflix (Low Confidence)
Description	Genetic study on Bi-Polar Disorder	User movie ratings	User movie ratings
BashReduce Task Sizing	2.5 MB	1 MB	1 MB
Tiniest Task Sizing	588.8 KB	118 KB	118 KB
Largest Task Sizing	1 TB	2 GB	2 GB

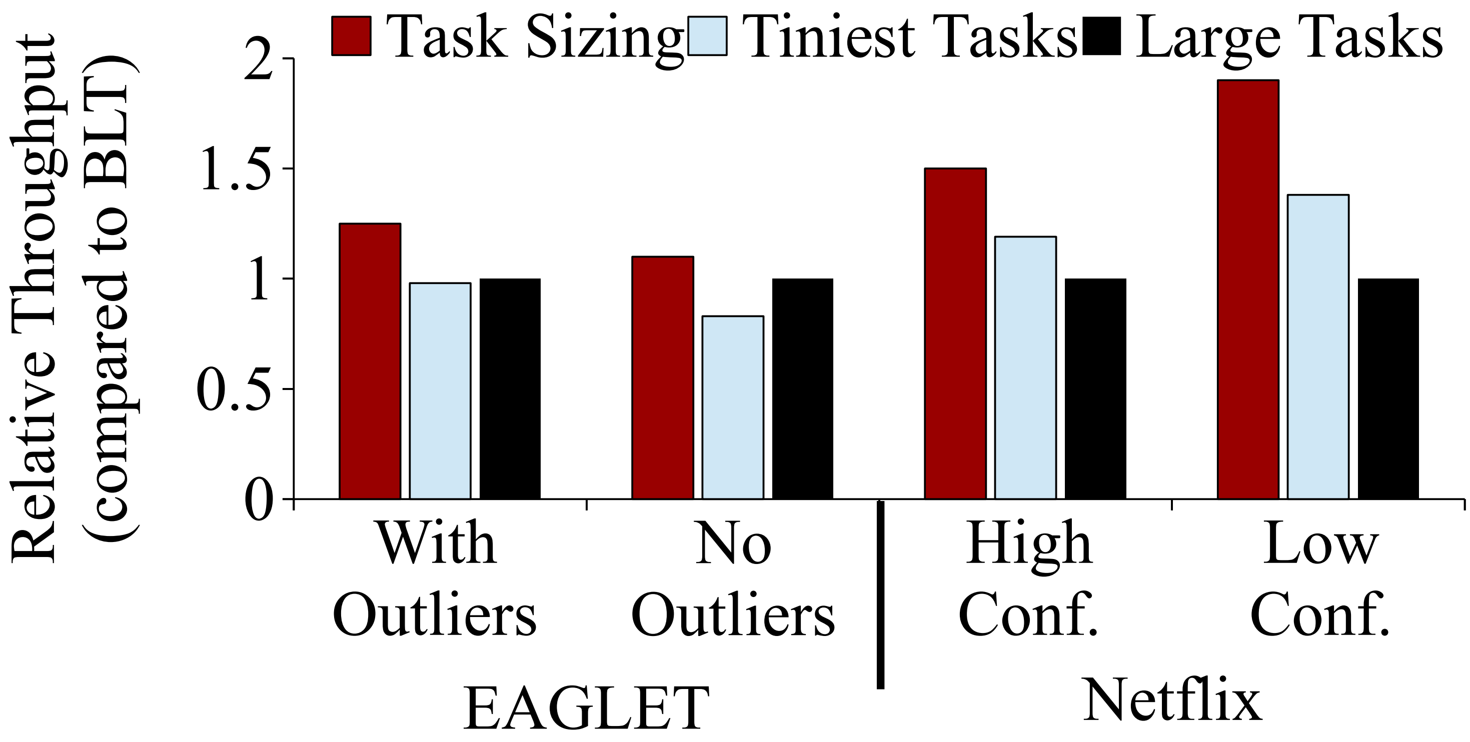
1: Xeon 12 cores/node, 2.0 GHz, 15 MB L2, 32 GB Memory

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**

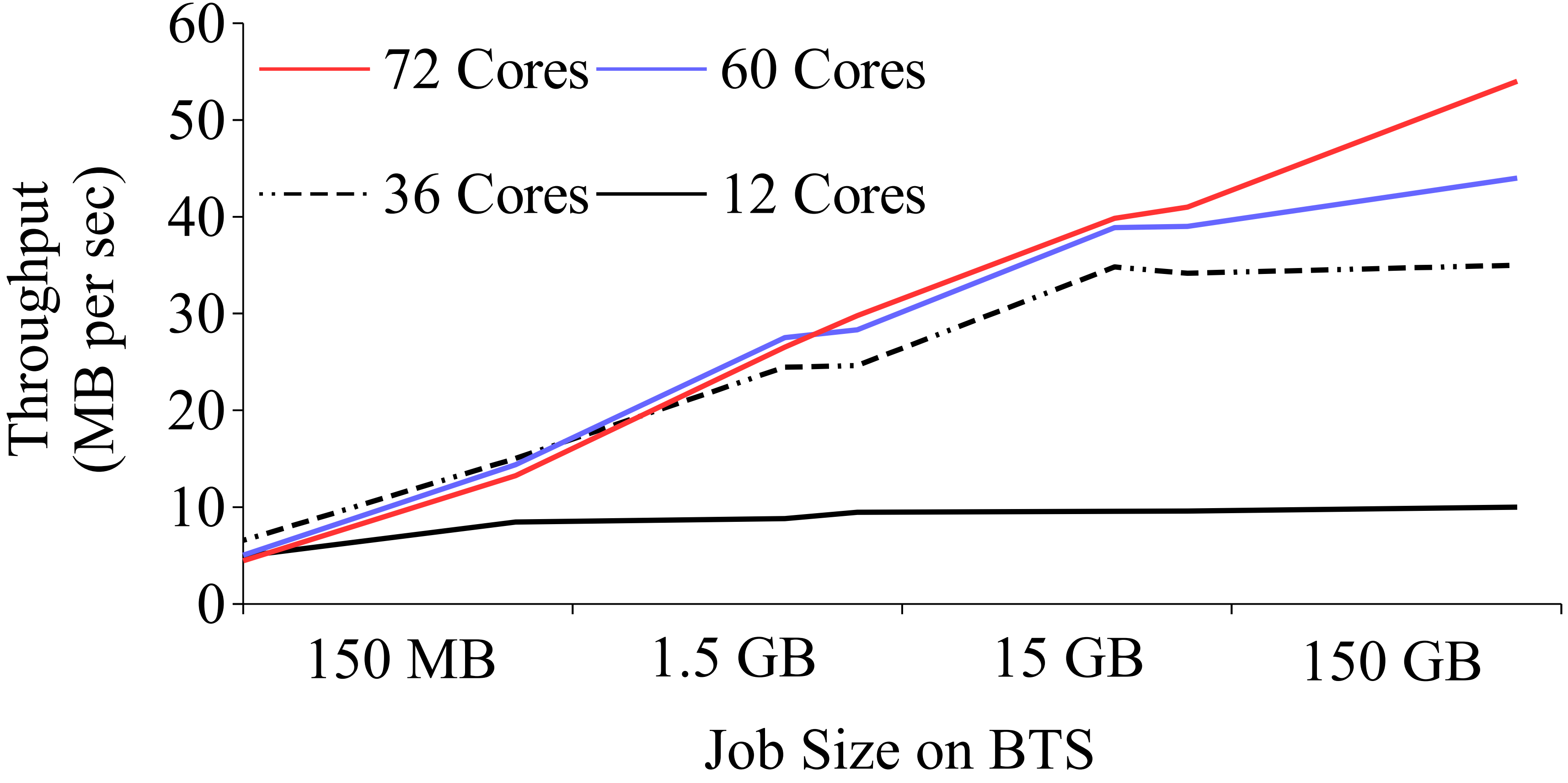
Task Sizing: Evaluation



Our Task Sizing has **consistent higher throughput**.

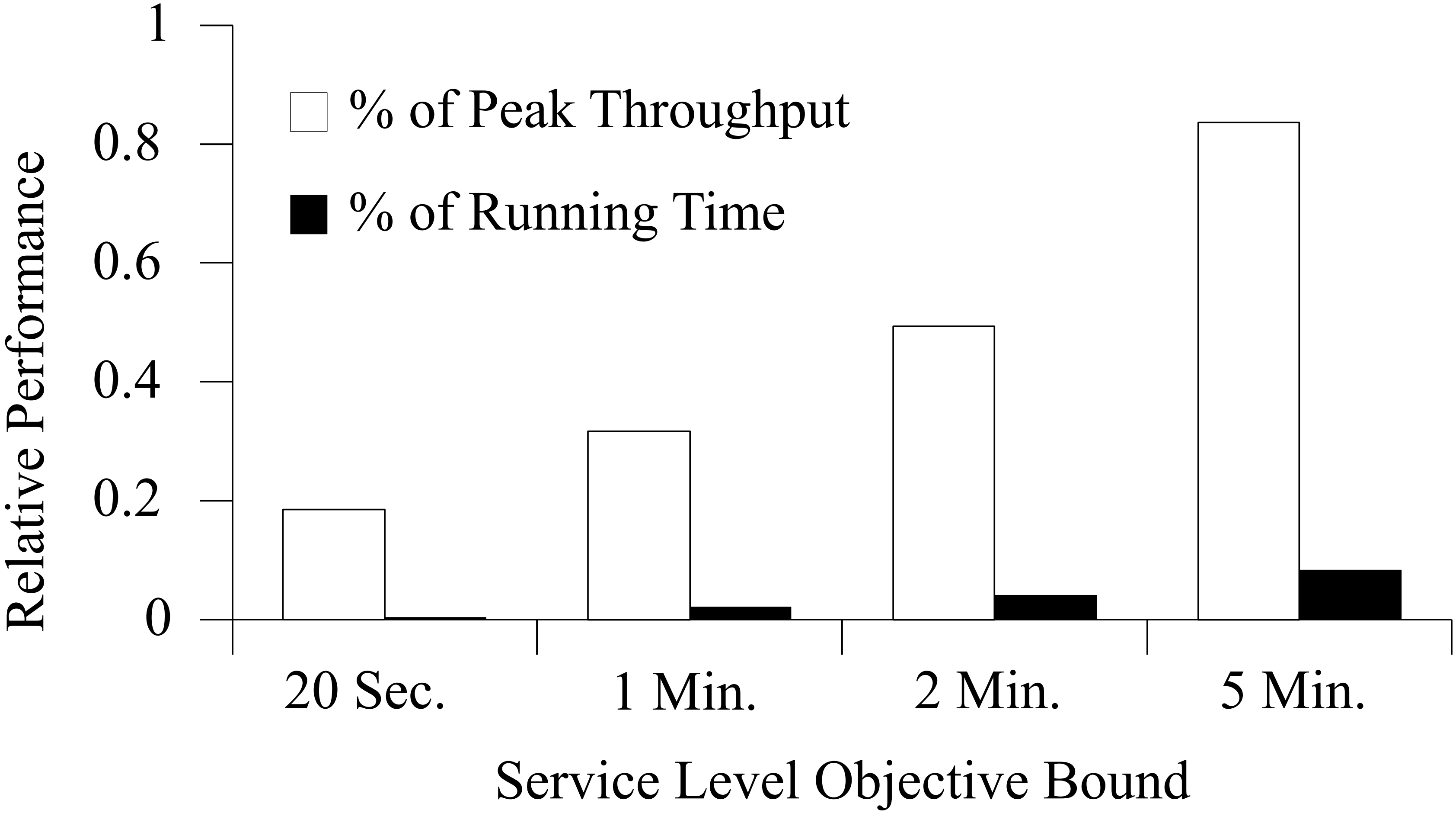
Type 1 hardware

Task Sizing: Evaluation



Type 2 Hardware

Task Sizing: Evaluation



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.