Ciel: A Universal Execution Engine For Distributed Data-Flow Computing

Presented by: Tejas Kannan

Date: 17/10/2018

Outline

1. Introduction
2. Implementation and Contributions
3. Critique and Further Reading
4. Conclusion

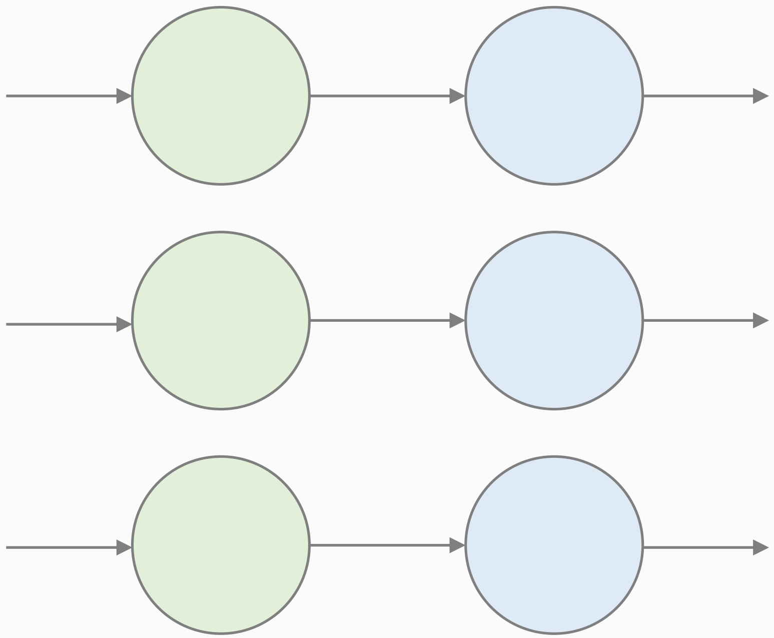
Background Details

- Authors: Derek Murray, Malte Schwarzkopf, Christopher Smowton, Steven Smith, Anil Madhavapeddy and Steven Hand

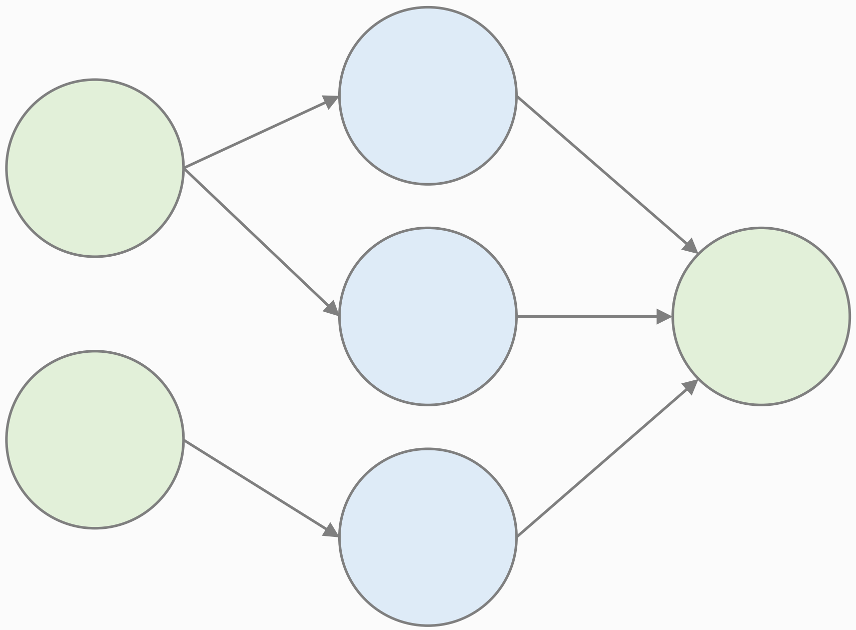
- Product of the University of Cambridge Computer Laboratory

- Published in 2011 at the NSDI Conference

Limitations of Existing Platforms



MapReduce [1,4,9]



Dryad [5,6]

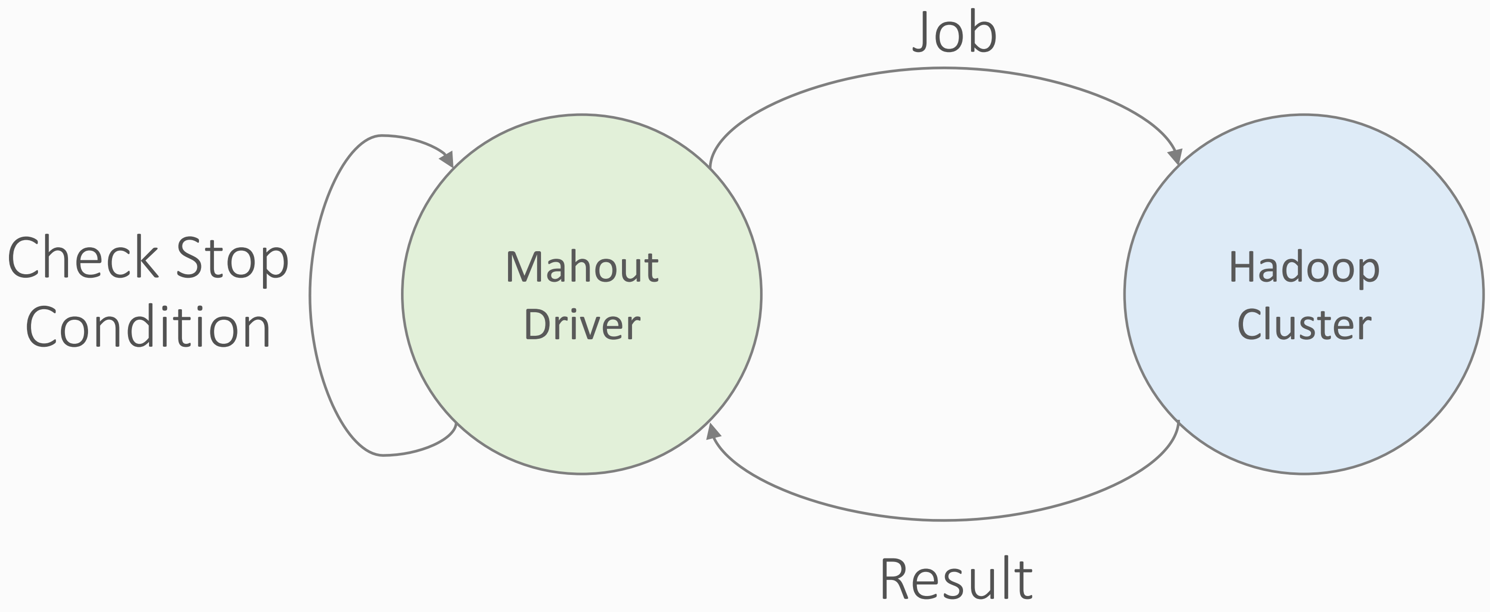
Limitations of Existing Platforms

MapReduce

Dryad

Issue: task graph is fixed, so iteration is difficult

Adding Iteration to Hadoop with Mahout [2]



Problems:
1. Job overhead every iteration
2. No fault-tolerance between iterations

Ciel's Dynamic Task Graph

Ciel enables a dynamic graph by allowing tasks to create follow-up tasks

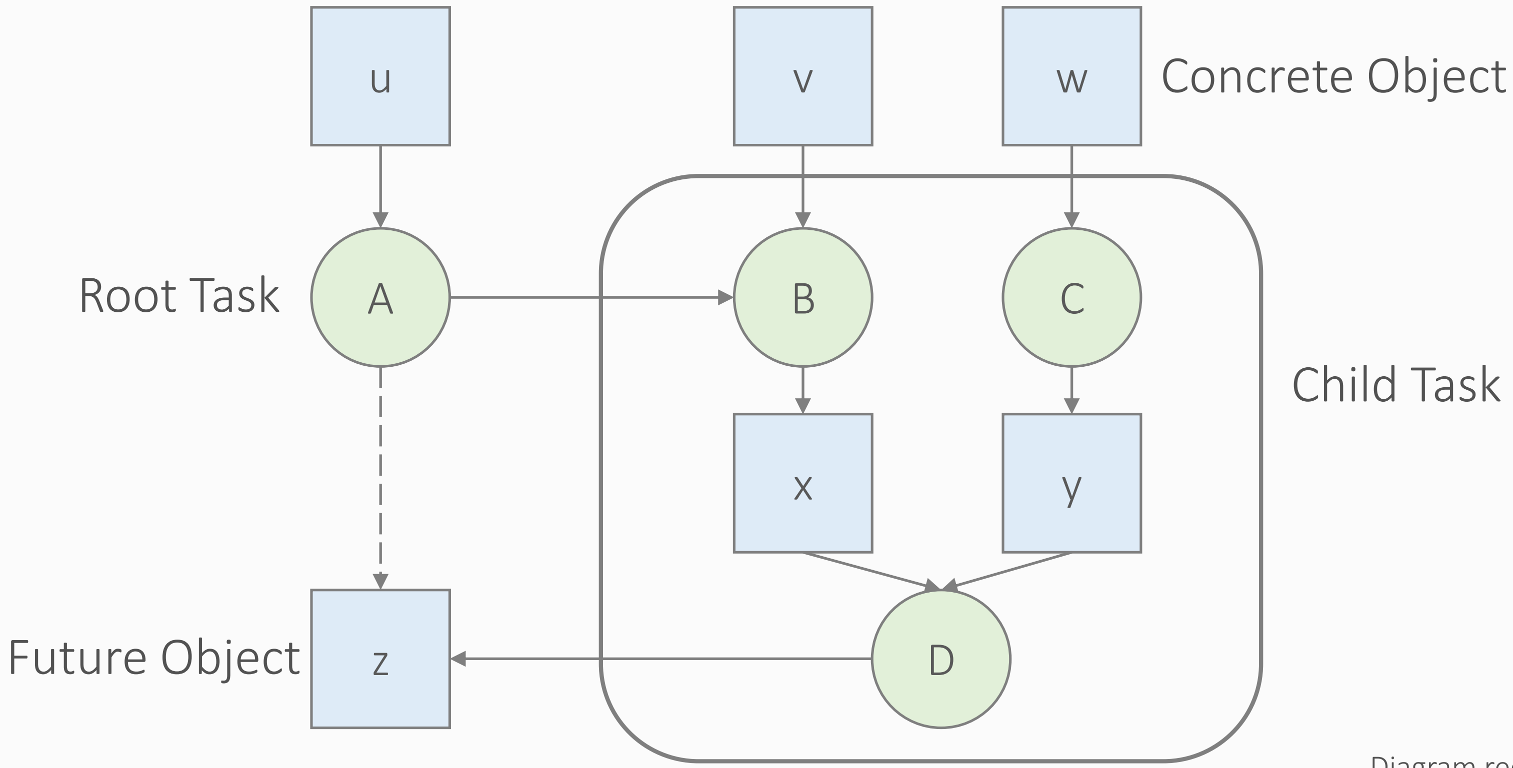


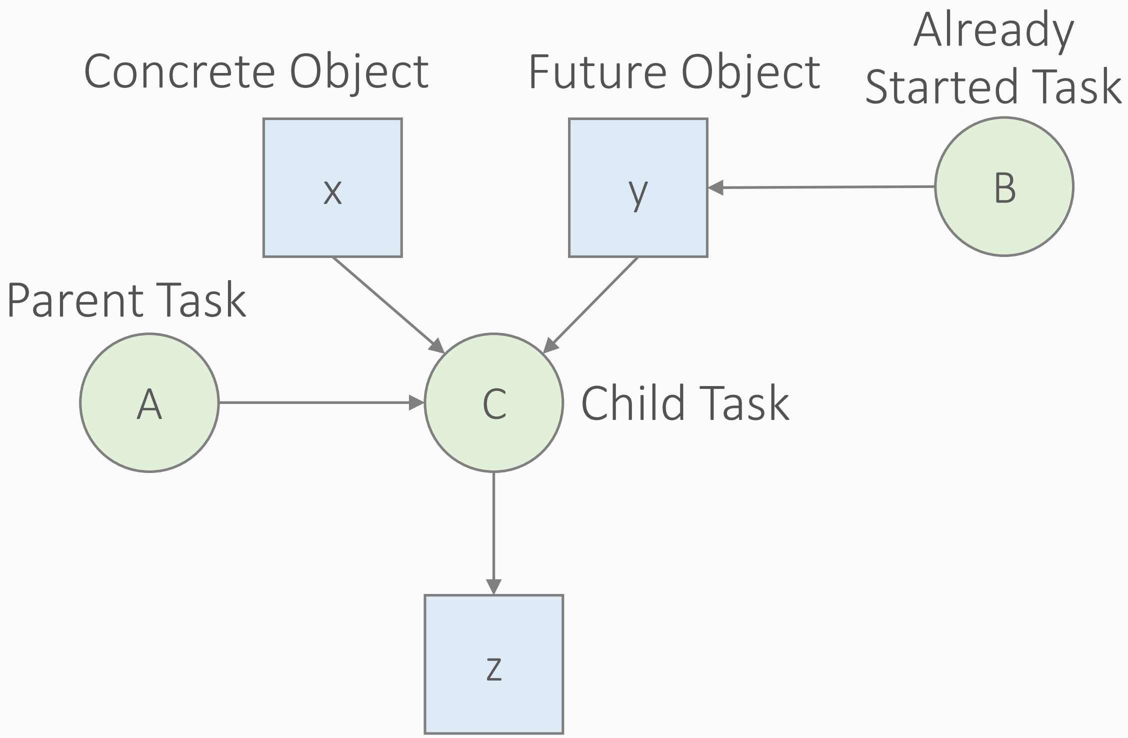
Diagram recreated from [8]

Preventing Cycles

A child task can depend only on:

1. Concrete references
2. Future references from already running tasks

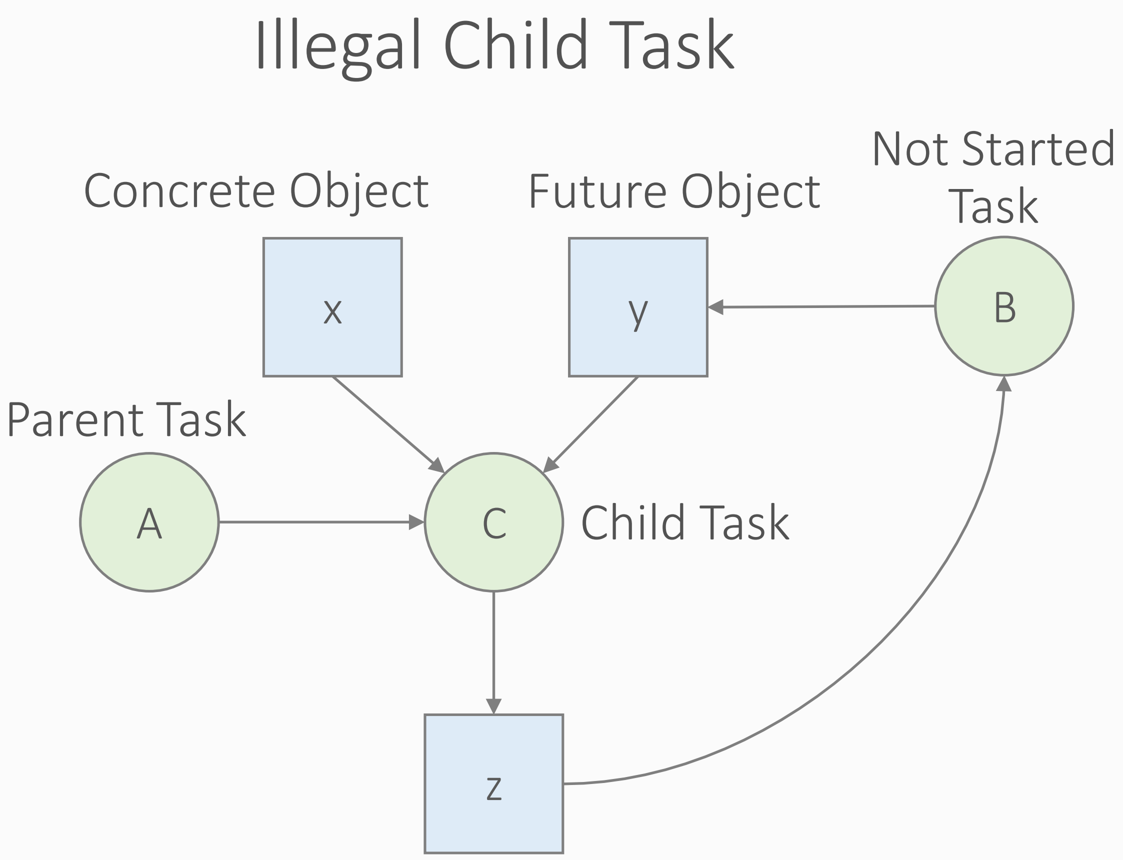
Legal Child Task



Preventing Cycles

A child task can depend only on:

1. Concrete references
2. Future references from already running tasks



System Architecture

Object Table: Maintains references to objects stored on workers

Worker Table: Holds worker nodes and used to track their health

Task Table: Contains references to spawned tasks, as well as their dependencies

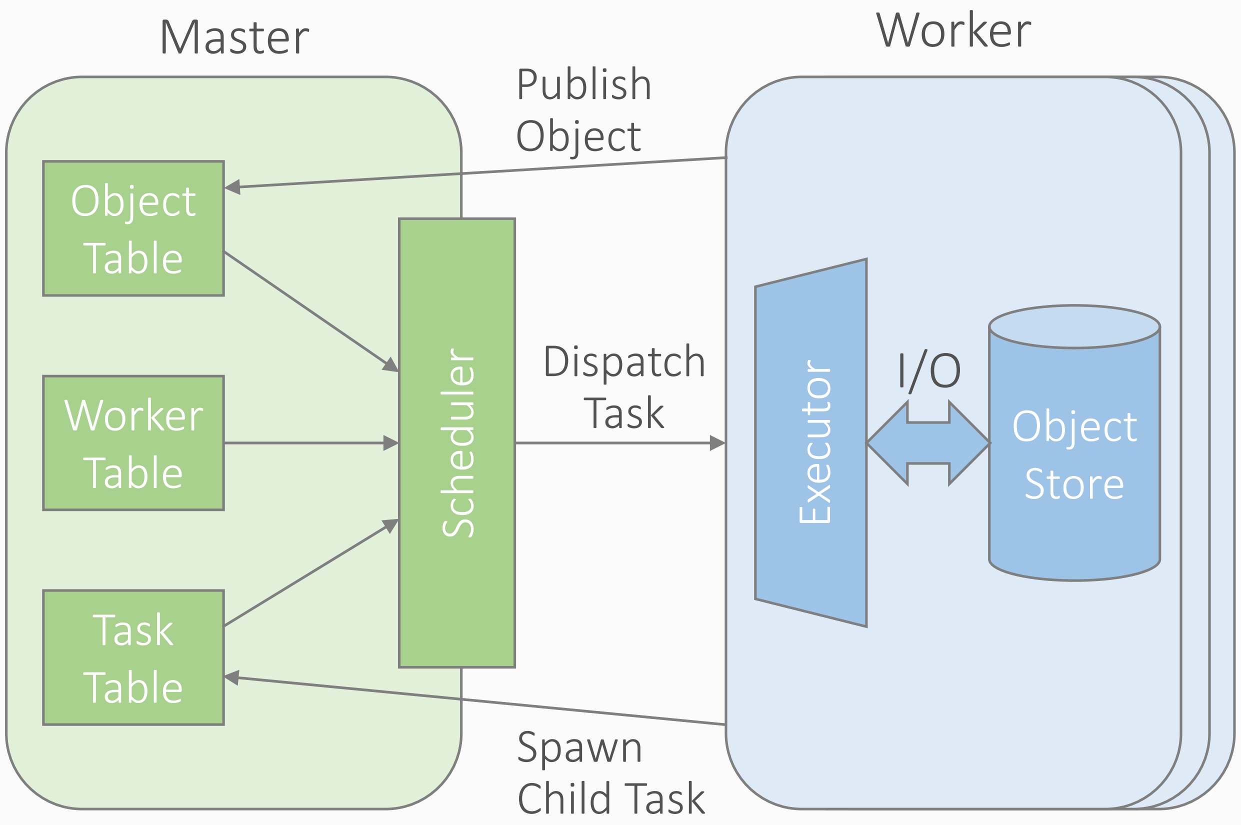


Diagram recreated from [8]

Scheduling Tasks

Scheduling is done using lazy evaluation:

1. Evaluate starting from the root task

2. For each subsequent task:

   a. If the task has concrete dependencies, evaluate it

   b. Otherwise, recursively evaluate tasks needed to resolve dependencies and unblock this task

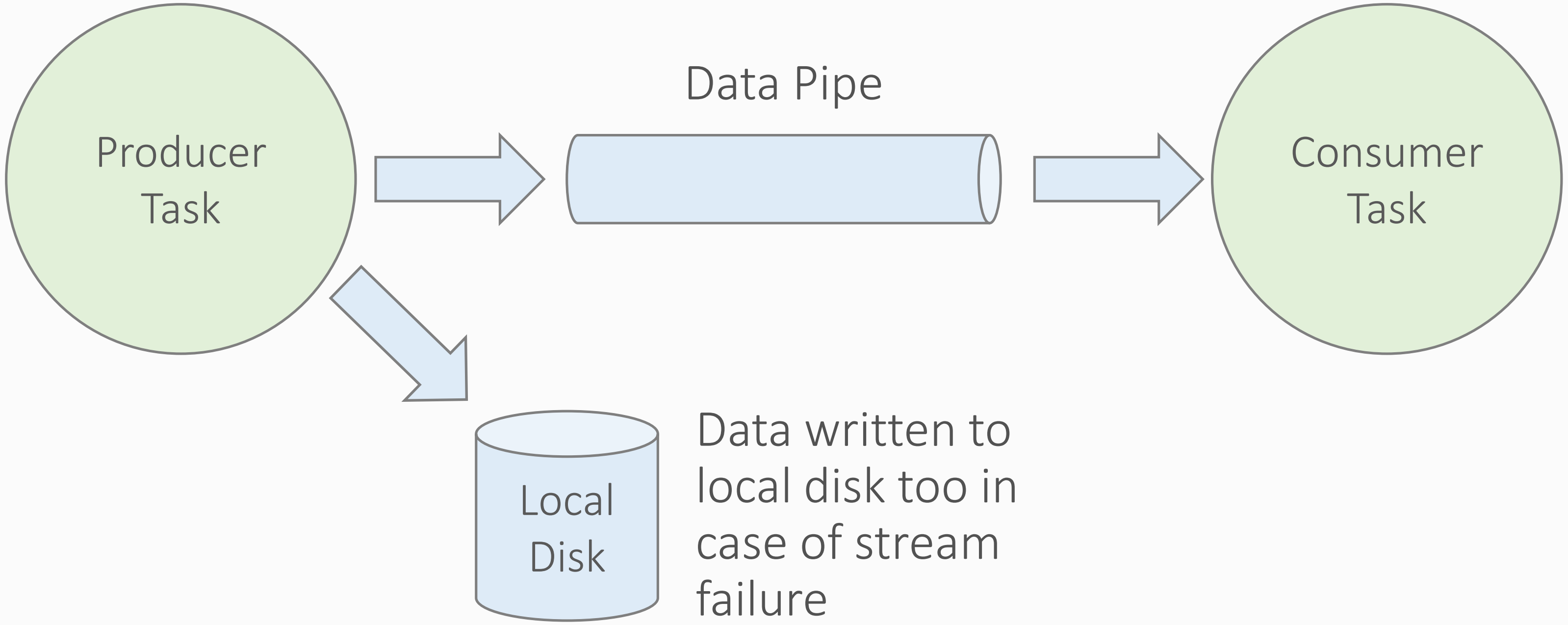
Tasks are dispatched to workers who are nearest to the data

Performance Optimization: Memoization

- Tasks are deterministic
- Objects are given unique names using properties of the parent task
- Object name and reference stored in master's object table
- If an object already exists, it is reused instead of recomputed
- Reduces runtime during computations which involve repetitive tasks

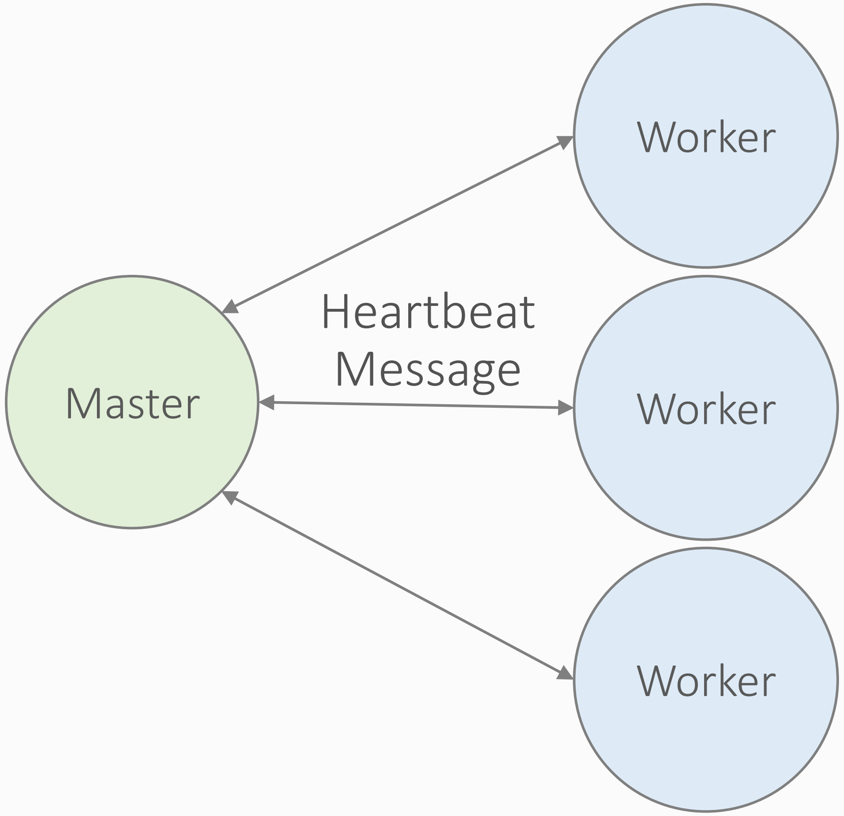
Performance Optimization: Streaming

Some tasks do not need the entire input object to start making progress



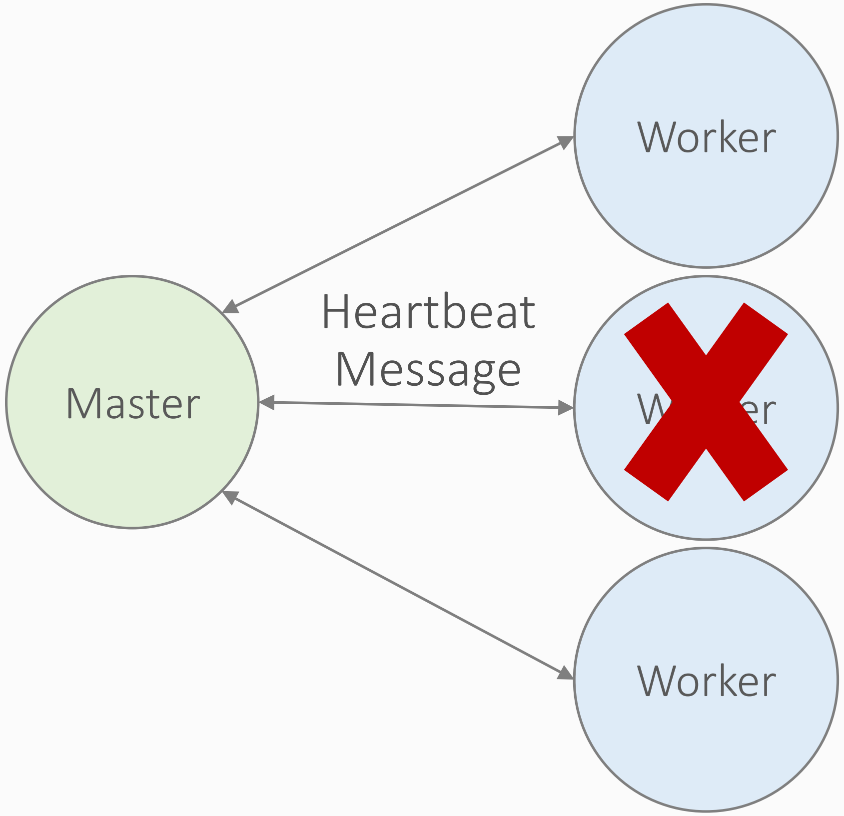
Recovering From Failures

Worker failures are detected using periodic heartbeat messages



Recovering From Failures

Worker failures are detected using periodic heartbeat messages

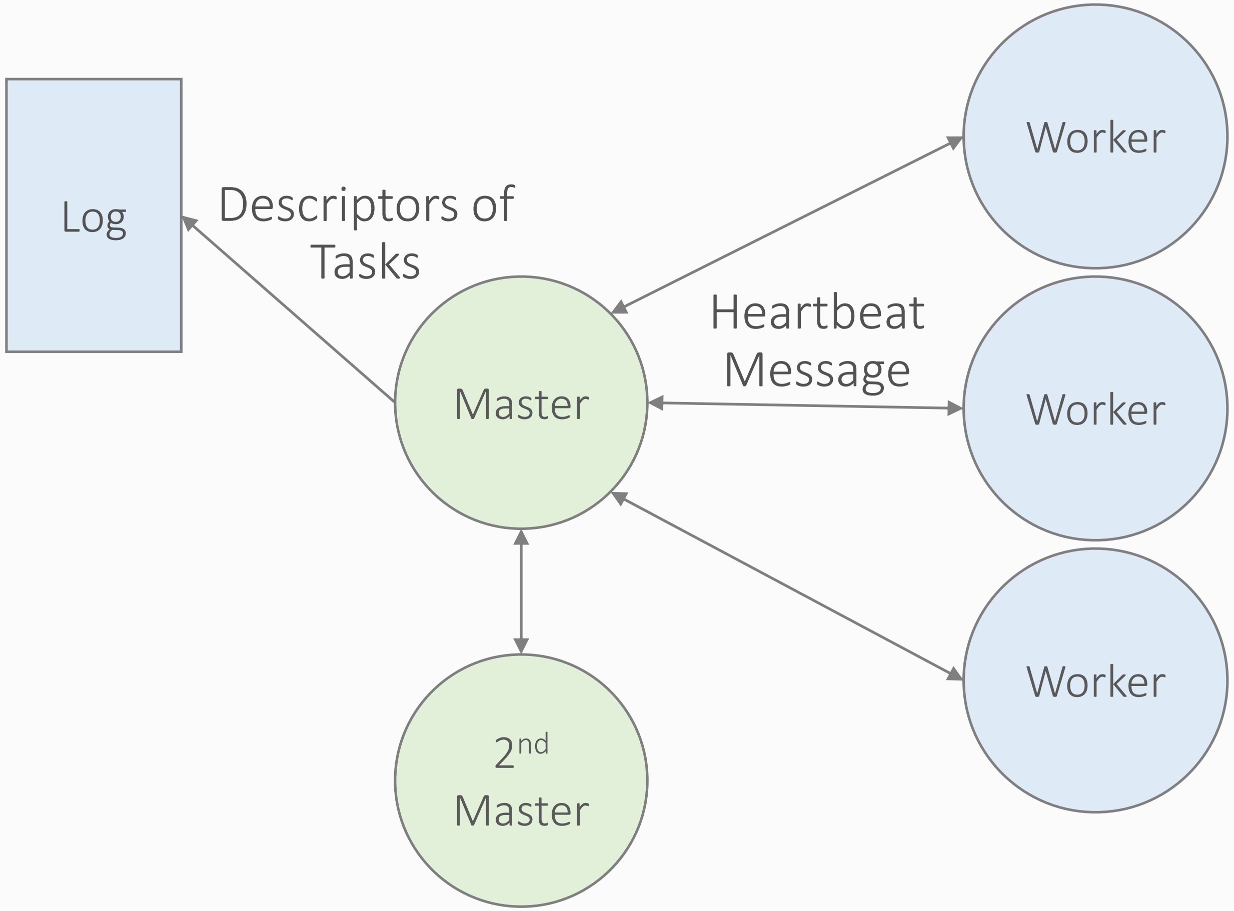


1. Master invalidates object references at the failed worker

2. Master schedules the re-computation of any lost object according to the lazy policy

Recovering From Failures

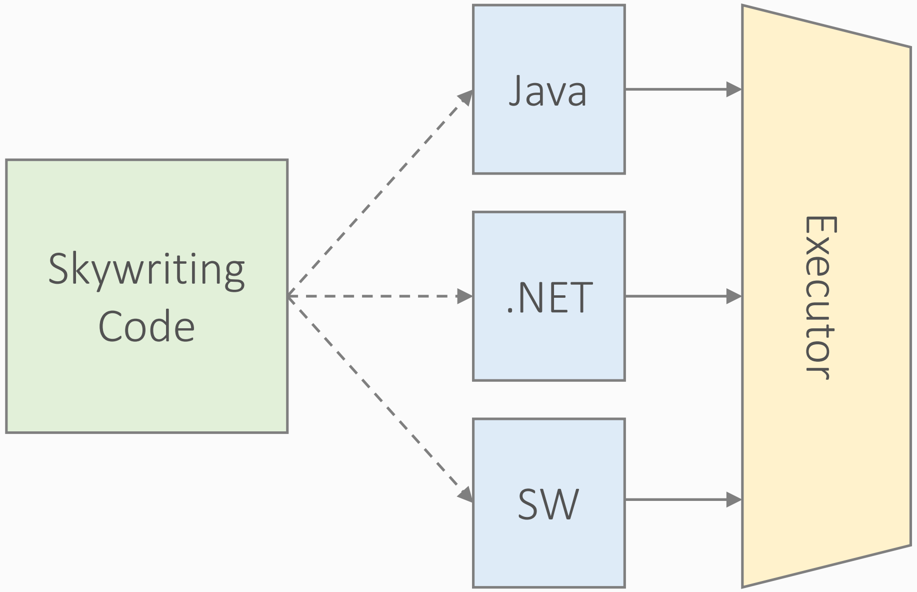
Master failures are also detected using periodic heartbeat messages



On recovery, a master node can rebuild its object table using the workers' object stores

Creating Ciel Jobs

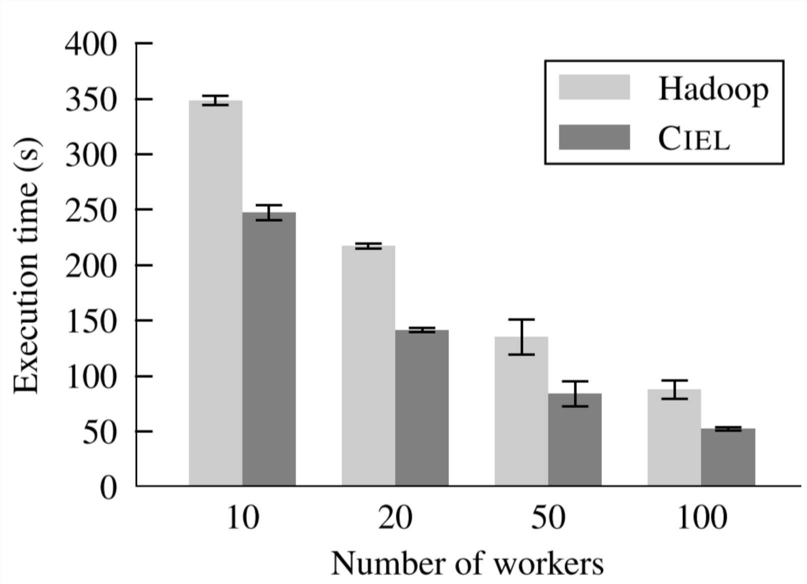
Skywriting is an interpreted language created to run Ciel jobs



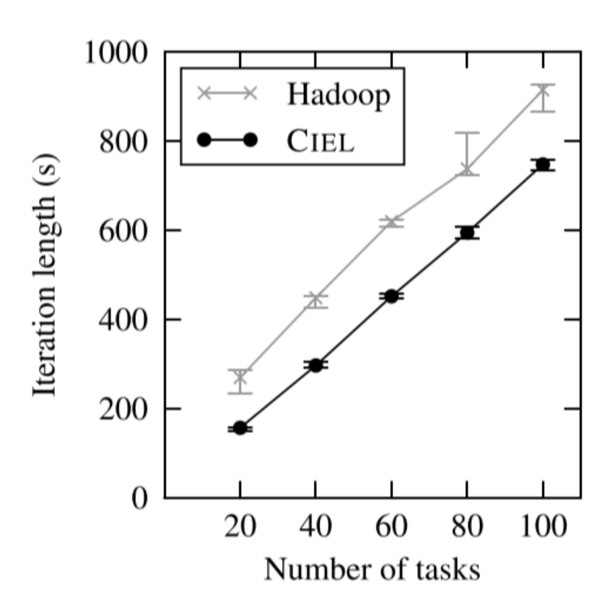
To boost performance, Skywriting tasks can make calls to procedures written in other languages

Performance Evaluation

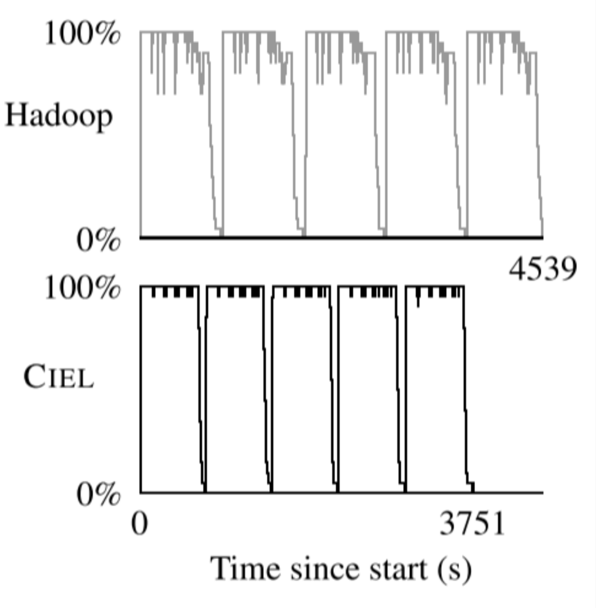
Ciel outperforms Hadoop when running both Grep and K-Means



Execution Time on Grep



Iteration Length on K-Means

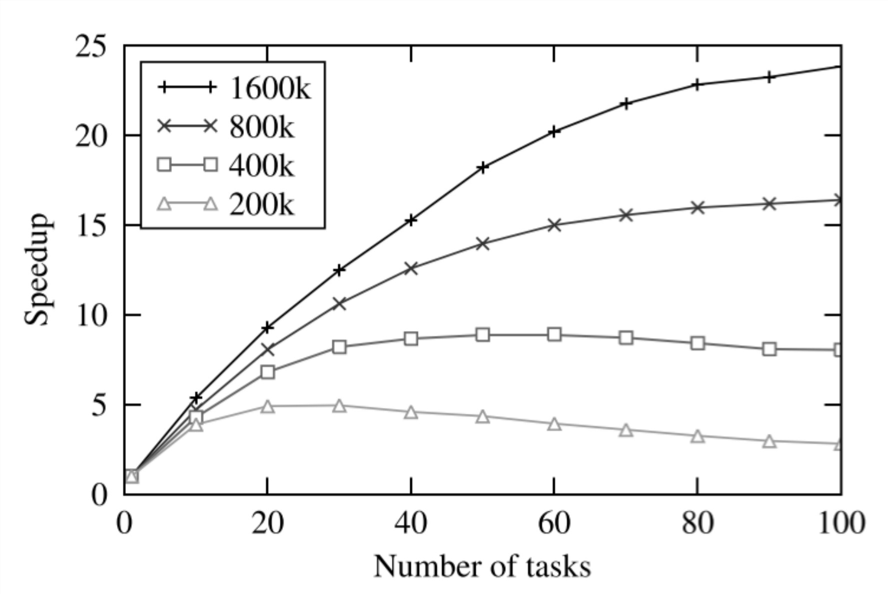


Cluster Utilization on K-Means

Graphs taken from [8]

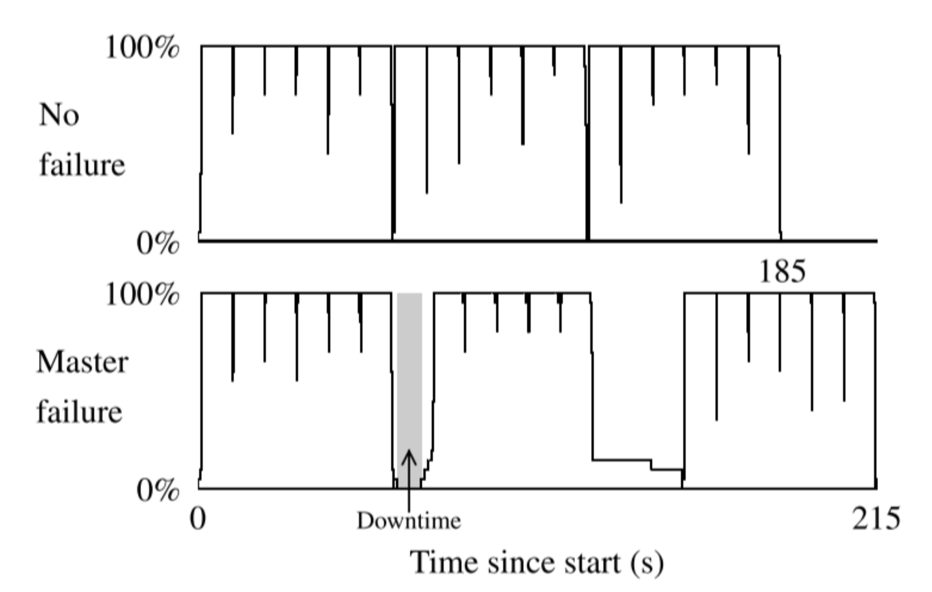
Performance Evaluation

Binomial Operations Pricing



Speedup Using Streaming

Failed Master During Iteration



Cluster Utilization

Graphs taken from [8]

Criticism

1. Ciel's execution is never compared to a more optimized iterative platform such as HaLoop [3]

2. Number of trials during testing never specified

3. Streaming optimization demonstrated but never compared to another system

4. Ciel does not use multiple cores on worker nodes while scheduling

Selection of Related Work

1. Hive enables SQL-like queries to be executed on large datasets using Hadoop [10]

2. Spark allows for iterative tasks and derives its efficiency from in-memory computation [11, 12]

3. Naiad uses cycles in its execution graph to enable low latency processing of streams, as well as iterative and incremental tasks [7]

Conclusion

1. Distributed data processing engine meant for general purpose tasks
2. Dynamic task allocation enables efficient iterative computations
3. Fault-tolerant design with automatic recovery
4. Scripting language Skywriting used to construct Ciel jobs
5. Empirically outperforms Hadoop on iterative tasks

References

[1] Apache hadoop. https://hadoop.apache.org/.

[2] Apache mahout. https://mahout.apache.org/.

[3] Yingyi Bu, Bill Howe, Magdalena Balazinska, and Michael D Ernst. Haloop: efficient iterative data processing on large clusters. Proceedings of the VLDB Endowment, 3(1-2):285–296, 2010.

[4] Jeffrey Dean and Sanjay Ghemawat. Mapreduce: simplified data processing on large clusters. Communications of the ACM, 51(1):107–113, 2008.

[5] Yuan Yu Michael Isard Dennis Fetterly, Mihai Budiu, Ulfar Erlingsson, and Pradeep Kumar Gunda Jon Currey. Dryadlinq: A system for general-purpose distributed data-parallel computing using a high-level language. Proc. LSDS-IR, page 8, 2009.

[6] Michael Isard, Mihai Budiu, Yuan Yu, Andrew Birrell, and Dennis Fetterly. Dryad: distributed data-parallel programs from sequential building blocks. In ACM SIGOPS operating systems review, volume 41, pages 59–72. ACM, 2007.

[7] Derek G Murray, Frank McSherry, Rebecca Isaacs, Michael Isard, Paul Barham, and Martín Abadi. Naiad: a timely dataflow system. In Proceedings of the Twenty-Fourth ACM Symposium on Operating Systems Principles, pages 439–455. ACM, 2013.

[8] Derek G Murray, Malte Schwarzkopf, Christopher Smowton, Steven Smith, Anil Madhavapeddy, and Steven Hand. Ciel: a universal execution engine for distributed data-flow computing. In Proc. 8thACM/USENIX Symposium on Networked Systems Design and Implementation, pages 113–126, 2011.

[9] Konstantin Shvachko, Hairong Kuang, Sanjay Radia, and Robert Chansler. The hadoop distributed filesystem. In Mass storage systems and technologies (MSST), 2010 IEEE 26th symposium on, pages 1–10.Ieee, 2010.

[10] Ashish Thusoo, Joydeep Sen Sarma, Namit Jain, Zheng Shao, Prasad Chakka, Suresh Anthony, Hao Liu, Pete Wyckoff, and Raghotham Murthy. Hive: a warehousing solution over a map-reduce framework. Proceedings of the VLDB Endowment, 2(2):1626–1629, 2009.

[11] Matei Zaharia, Mosharaf Chowdhury, Tathagata Das, Ankur Dave, Justin Ma, Murphy McCauley,Michael J Franklin, Scott Shenker, and Ion Stoica. Resilient distributed datasets: A fault-tolerant abstraction for in-memory cluster computing. In Proceedings of the 9th USENIX conference on Networked Systems Design and Implementation. USENIX Association, 2012.

[12] Matei Zaharia, Mosharaf Chowdhury, Michael J Franklin, Scott Shenker, and Ion Stoica. Spark: Cluster computing with working sets. HotCloud, 10(10-10):95, 2010.

Questions?