CIEL: Just in time local reference availability

Bogdan-Alexandru Matican

University of Cambridge

March 12, 2013

Table of contents

1 Introduction

2 Improvements

3 Approach

4 Conclusion

CIEL

Some relevant highlights and focused details from the system:

- task execution engine
- dynamic task graph
- data comes from references (internal or external)
- tasks depend on input references and generate output ones
- new tasks spawned on existing workers
- data must be pulled in locally before execution starts

Status quo

Workers cannot start tasks without data being locally available.
Currently:

- tasks execute through the system, generating references
- master runs topological sorting on DAG for next tasks
- master decides new task can/should start
- worker is appointed to start a certain task
- needed references are pulled in from other locations
- worker finally begins task execution

Idea

Waiting for data seems wasteful. Why not fast-forward?

- analyze DAG, predict next tasks
- get references migrated before the tasks begin

Approach

- understand tasks dependencies centrally
- if task finishes and outputs might be needed, migrate
- on output, write locally and remotely for efficiency

Problems

- external reference dependencies
- dynamic reference requirements
- predictive powers
- reference size relative to network IO
- test in multi-machine environment

Conclusion

- CIEL looks fun
- nicely written python codebase
- descriptive research paper
- good room for optimizations