Framework for profiling Critical Path related Algorithms

Henri Trenquier

Supervisors:
Dr. Zhiming Zhao & Dr. Arie Taal
MSc Systems and Network Engineering

UNIVERSITY OF AMSTERDAM

February 6, 2018
Context

LIGO Gravitational wave detection analysis

- Data to be processed
  - Deadline
  - Order
- Time critical problem
Context
LIGO Gravitational wave detection analysis

- Defines a task graph
Context
Critical Path related Algorithms

• "Deadline-constrained workflow scheduling algorithms for Infrastructure as a Service Clouds" (2013)
  Saeid Abrishami and Mahmoud Naghibzadeh and Dick H.J. Epema
  Cited 311 times.

• Critical path related algorithms i.e. IC-PCP

Figure: Critical Path on a Directed Acyclic Graph
Complexity of the problem

Multi-parameter problem:
Complexity of the problem

Multi-parameter problem:
- Time constraints
Complexity of the problem

Multi-parameter problem:

- Time constraints
- Performance model
Complexity of the problem

Multi-parameter problem:
- Time constraints
- Performance model
- Cost constraints
Complexity of the problem

Multi-parameter problem:

- Time constraints
- Performance model
- Cost constraints
- Unlimited task graphs
Can we profile critical path related algorithms by the means of a framework?

- How relevant is a framework for profiling critical path related algorithms?
- What are the challenges for such a framework?
Motivation

Why a framework?
- Compare algorithms performances
- Compare implementations of the same algorithm
- Manipulate and plot output data

What is the best implementation for the "Inspiral" (LIGO) task graph?
Prototype Architecture

Figure: Prototype Architecture
Prototype Inner Working

**Figure:** Prototype Architecture

1. **Step 1: main.py**
   - Browse input task-graph dataset
   - Run algorithm on dataset with specific parameters
   - Write the results in an output file

2. **Step 2: plot.py**
   - Browse output file lines
   - Structure the data
   - Plot the structured data
File Formats

- Input task graph format
  - Topology

- Performance Model

- VM prices

- Output file format
Input Data sets

Other topologies: Cybershake, Montage, Sipht, Epigenomics
Assumption: “If the algorithm succeeds for a shorter deadline, then it will succeed for a longer one”

Input datasets: Montage & Inspiral

KPI: shortest deadline
Experiment #2

- Input datasets: *Inspiral*
- KPI: Time and Success rate

![Graphs showing Success Rate and Average Time](image)
Experiment #2

- Input datasets: **Montage**
- Experiment #1 assumption: False
Discussion

- Performance Model: not changeable
- Algorithms’ codes have to be adapted
- Solution check is done by the algorithm (not the framework)
- Framework needs a lot of resources
- Output format: Cost?
Conclusion

Can we profile critical path related algorithms by the means of a framework?

- Meaningful results
- Assumption proved wrong
- Empirical solution
More motivations

But also...

- Store algorithms and results
- Authoritative source

Figure: https://www.nature.com/news/1-500-scientists-lift-the-lid-on-reproducibility-1.19970
Future Work

- Universal standards (challenge)
- Web-application
- Community databases
- New features:

<table>
<thead>
<tr>
<th>Feature</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>DAG Generator</td>
<td>Create and visualize new topologies</td>
</tr>
<tr>
<td>DAG Parsing</td>
<td>Manage different input topologies format</td>
</tr>
<tr>
<td>input checker</td>
<td>Check input algorithm parameters</td>
</tr>
<tr>
<td>Solution checker</td>
<td>Make sure the final solution is valid</td>
</tr>
</tbody>
</table>

Table: Framework features