Modeling and Optimization of Resource Allocation in Cloud
PhD Thesis Proposal

Atakan Aral

Thesis Advisor: Asst. Prof. Dr. Tolga Ovatman

Istanbul Technical University – Department of Computer Engineering

June 25, 2014
Outline

1. Introduction
2. Main Topics of the Thesis
3. Methods and Techniques
   - MapReduce Configuration
   - Resource Selection and Optimization
   - Work Distribution to Resources
4. Time Plan
5. Conclusion
Cloud Computing

Definition

Applications and services that run on a distributed network using *virtualized* resources and accessed by common Internet protocols and networking standards.

- **Broad network access**: Platform-independent, via standard methods
- **Measured service**: Pay-per-use, e.g. amount of storage/processing power, number of transactions, bandwidth etc.
- **On-demand self-service**: No need to contact provider to provision resources
- **Rapid elasticity**: Automatic scale up/out, illusion of infinite resources
- **Resource pooling**: Abstraction, virtualization, multi-tenancy
Cloud computing paradigm is revolutionary, however the technology it is built on is only evolutionary.
Cloud Computing Benefits

- Lower costs
- Ease of utilization
- Quality of Service
- Reliability

- Outsourced IT management
- Simplified maintenance and upgrade
- Lower barrier to entry
Cloud Computing Architecture
MapReduce

Definition

A programming model for processing large data sets with a parallel, distributed algorithm on a cluster.
MapReduce Entities

- **DataNode**: Stores blocks of data in distributed filesystem (HDFS).
- **NameNode**: Holds metadata (i.e. location information) of the files in HDFS.
- **Jobtracker**: Coordinates the MapReduce job.
- **Tasktracker**: Runs the tasks that the MapReduce job split into.
Life Cycle of Cloud Software

Development of Distributed Software
- MapReduce Configuration

Resource Allocation
- Resource Selection and Optimization

Load Balancing
- Work Distribution to Resources
Motivation

The cost of using 1000 machines for 1 hour, is the same as using 1 machine for 1000 hours in the cloud paradigm (Cost associativity).

- Optimum number of maps and reduces that maximize resource utilization are dependent on the resource consumption profile of the cloud software.
- Bottleneck resources should be identified.
- Optimization at Application level
In distributed computing environments, up to 85 percent of computing capacity remains idle mainly due to poor optimization of placement.

- Better assignment of virtual nodes to physical nodes may result in more efficient use of resources.
- There are several possible constraints to consider / optimize e.g. capacity limits, proximity to user, latency etc.
- Optimization at Infrastructure level
Motivation

Data flows between nodes only in the shuffle step of the MapReduce job, and tuning it can have a big impact on job execution time.

- Mapper and reducer nodes should be selected carefully to minimize network traffic.
- Dynamic load balancing should be ensured.
- Optimization at Platform level
Problem

Aim

Maximizing the utilization of all nodes for a Hadoop job by calculating the optimum parameters i.e. number of mappers and reducers

- Higher values mean higher parallelism but may cause resource contention and coordination problems.
- Optimum parameters depend on the resource consumption of the software.
Previous Solutions


- Calculates the optimum parameters (number of M/R) for the Hadoop job such that each resource set is fully utilized.
- Each application has a different bottleneck resource and utilization.
- Requires to run a small chunk of the application to create a signature
  - Split job into $n$ intervals of same duration.
  - Calculate the average consumption of all 3 resources (CPU, Disk, and Network) in each interval.
- Uses the configuration of the most similar signature in the database.
Previous Solutions

1. INPUT DATA
2. Generate Small Chunk
3. Run Application On Hadoop
4. Generate Signature for Small Data Run
5. Generate New Config. File
6. RUN App. On Entire Data With New Configuration

SIG. Database
App Sig
Optimum Config.
Suggested Solution

- Design model of the software will be statically analyzed in order to guess resource consumption pattern.
- Critical (bottleneck) resources will be identified.
- If required, source code or input data may also be included to the analysis.
Output

- An algorithm that calculates optimum Hadoop configuration for a given software
- An API that receives software model and outputs a Hadoop configuration suggestion
Problem

Aim

Optimally assigning interconnected virtual nodes to substrate network with constraints

- Constraints:
  - Datacenter capacities and bandwidth
  - Virtual topology requests and incompletely known cloud topology
  - Locality and jurisdiction
  - Application interaction and scalability rules

- Objectives (Minimization):
  - Inter-DC communication
  - Geographical proximity to user and latency
Previous Solutions


- Mapping user requests for virtual resources onto shared substrate resources
- Problem is solved in two coordinated phases: node and link mapping.
- In node mapping, the objective is to minimize the cost of mapping and the method is the random relaxation of MIP to LP.
- In link mapping, the objective is to minimize the number of hops and the method is shortest path or minimum cost flow algorithms.
- Suggested algorithm is compared with greedy heuristics in terms of acceptance ratio and number of hops.

- Which component of the software should be hosted at which datacenter?
- Minimize delay, cost, energy consumption, and CO$_2$ emission.
- Greedy initial solution is improved by moving one component at each step.
- A random subset of neighbours are analyzed for the best improvement.
- Suggested tabu search heuristic is compared with MIP formulation and greedy heuristic in terms of execution time and optimality.
- Tradeoff analysis between the multiple objectives
Previous Solutions


- Volley analyzes trace data and suggest migrations to improve DC capacity skew, inter-DC traffic and user latency.
- Considers client geographic diversity, client mobility and data dependency.
- Method contains of 3 sequential phases:
  1. Compute initial placement using weighted spherical mean,
  2. Iteratively move data to reduce latency,
  3. Iteratively collapse data to datacenters.
- Compared against 3 simple heuristics: oneDC, commonIP and hash.
Suggested Solution

- VN requests and DC network are represented as undirected weighted graphs.
  - Nodes store computational capacities (i.e. CPU, memory) and storage.
  - Edge weights represent bandwidth capacities.
- Graph similarity - subgraph matching algorithms will be employed.
- Suggested solution will be static.
An algorithm that optimizes the resource provisioning

An API that receives graph and constraint inputs and outputs a matching

Problem

Aim

Analysis of the shuffle and scheduling algorithms of Apache Hadoop framework.

- Initial selection of DataNodes, Mappers and Reducers is critical to reduce network traffic.
- Dynamic re-distribution of work during execution results in better load balancing and better performance.
Previous Solutions

- **FIFO scheduler**: Integrated scheduling algorithm, manual prioritization
- **Fair scheduler**: Each job receives an equal share of resources over time, on average. Organizes jobs into user pools that are resourced fairly.
- **Capacity scheduler**: Allows sharing a large cluster while giving each user a minimum capacity guarantee. Priority queues are used instead of pools where excess capacity is reused.
- **Hadoop On Demand**: Provisions virtual clusters from within larger physical clusters. Adapts and scales when the workload changes.
- **Learning scheduler**: Chooses the task that is expected to provide max utility.
- **Adaptive scheduler**: User specifies a deadline at job submission time, and the scheduler adjusts the resources to meet that deadline.
Suggested Solution

- Cost based analysis of the existing schedulers
- Which scheduler is more appropriate for given costs of map, reduce, shuffle phases?
- How can DataNode selection for blocks and clones be optimized to reduce network traffic?
Formal modeling and analysis of Hadoop schedulers

An API that receives costs of MapReduce phases and outputs a scheduler suggestion

A patch for open-source Apache Hadoop framework
## Time Plan

<table>
<thead>
<tr>
<th>2014</th>
<th>2015</th>
<th>2016</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td>7</td>
<td>8</td>
<td>9</td>
</tr>
<tr>
<td>10</td>
<td>11</td>
<td>12</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Resource Selection
- Research
- Modeling
- Development
- Documentation

### MapReduce Configuration
- Research
- Modeling
- Development
- Documentation

### Work Distribution
- Research
- Modeling
- Development
- Documentation

Today
Summary

- Optimization on 3 related phases of the cloud software life cycle
  1. Map Reduce Configuration
  2. Resource Selection and Assignment
  3. Work Distribution and Load Balancing

- Graph based modeling of the cloud environment and resource allocation problem

- Holistic approach that aims to increase the performance of cloud software (Map Reduce) by optimizing resource allocation
Unique Aspects

- Inclusion of cloud computing aspects to the RA problem
  - Virtual topology requests and incompletely known network topology
  - Locality and Jurisdiction
  - Scalability rules
- Inclusion of software characteristics (i.e. design model) to the RA problem
- Optimization with different objectives at each level
Our goal is to contribute to the long-held goal of utility computing.

Cloud providers will benefit since they will be able to use their resources more efficiently and will be able to serve more customers without violating the service level agreements.

Cloud users will also benefit in terms of shorter application execution time and less infrastructure requirement (lower cost).
Thank you for your time.
Appendix I: Cloud Computing Architecture
Appendix II: Resource Allocation Problem in Cloud

- Conceptual phase
  1. Resource modeling: Notations only exist for concrete resources. Different levels of granularity can be chosen. Vertical and horizontal heterogeneity cause interoperability problems.
  2. Resource offering and treatment: Independent from modeling. New requirements are present in addition to common network and computation ones.

- Operational phase
  3. Resource discovery and monitoring: Finding suitable candidate resources based on proximity or impact on network. Passive or active monitoring strategies exist.
  4. Resource selection and optimization: Finding a configuration that fulfills all requirements and optimizes infrastructure usage. Dynamicity, algorithm complexity, and large number of constraints are the main problems.