Dynamic QoS Optimization Architecture for Cloud-based DDDAS

Tao Chen, Rami Bahsoon
txc919, r.bahsoon@cs.bham.ac.uk
School of Computer Science
University of Birmingham, UK

Georgios Theodoropoulos
georgios.theodoropoulos@durham.ac.uk
School of Engineering and Computer Sciences
Durham University, UK

ICCS, DDDAS workshop 2013
Outline

• Background and Motivation
• Objectives
• Challenges
  1. Quality of Service (QoS) Sensitivity
  2. Conflicting Objectives
• Architectural Solution
• Conclusion
Quality of Service

• What is Quality of Service (QoS)?

“QoS is the service providers' ability to achieve the service users' requirements, such as response time, throughput, availability, security and so forth”
Cloud-based DDDAS

• Dynamic data-driven application
• Computation and data intensive
• Depends on the Cloud

Examples: weather prediction, traffic management, bio-sensing etc.
Motivation

• Large volume and fluctuated data to be executed at runtime.
• Can not estimate demand of DDDAS applications at design time
• Over-/under-provisioning issues in the cloud
• QoS violations
Motivation

• Require new ways for improving the elastic adaptation, so we can cope with dynamic at runtime.

• Require new approach to provide more accurate adaptation with reasonable efficiency.

Can be achieved by leveraging on DDDAS engineering principles to assist runtime adaptation.
Cloud Architecture

Environmental Primitives (e.g., workload)

Service 1

QoS

Service 2

Software Control Primitives (e.g., threads, session lifetime, security policy)

Platform as a Service

Community Service

Business Service

IT Service

Infrastructure as a Service

Hardware Control Primitives (e.g., CPU, memory)

BODAS Application
Objectives

Dynamically optimizing QoS of cloud-based DDDAS by provisioning the best combination of control primitives, while minimizing the cost of these primitives and considering various constrains.

The theoretical optimal QoS is:
reach the best possible value of the QoS using the allocated control primitives, while not leaving any of these primitives idle.

The info-symbiotic loop of DDDAS principle is a promising approach to achieve this objective.
QoS sensitivity

QoS is sensitive to both control primitives and environmental primitives.

• Questions to solve:

  • Which primitives could correlate with a QoS?
  • When these primitives correlate with the QoS?
  • How the uncertainty of the QoS can be apportioned and sensitive to these primitives?

QoS modeling – a model that takes correlated primitives as inputs and predict the expected QoS as output.
Conflicting Objectives

Conflict could be *intra* and *inter* services. e.g. Performance versus Costs, Security versus Performance, Consistency versus Performance, Availability versus Costs etc.

Questions to solve:

- Which objectives to trade?
- How much proportions to trade?
Dynamic Multi-objectives Optimization Problem (DMOP)

Dynamic QoS optimization for the cloud-based DDDAS can be formulated as DMOP.

The process incorporates dynamic tradeoffs decision making, where the dynamics are attributed to continuous changes in the objective function, their degree of conflict and constraints.
Architectural Solution

A replica of service

$S_{11}$

$S_{12}$

$S_{ij}$

1. new data

2. historical data

3. primitives

4.1. fetch new data to update model

4.2. if optimization is needed

5. temporally decided provision

6. QoS Optimizer

7. Feedback

Simulator$_{11}$

Simulator$_{12}$

Simulator$_{ij}$

Data Sensor

Primitives Selector

QoS Objective Function Trainer

12
Architectural Solution

Simulator_{11}

Simulator_{12}

\ldots

Simulator_{ij}

\ldots

Data Sensor

Primitives Selector

QoS Objective Function Trainer

QoS Optimizer

1. new data

2. historical data

3. primitives

4.1. fetch new data to update model

4.2. if optimization is needed

5. temporally decided provision

7. Feedback

new data

historical data

primitives

fetch new data to update model

if optimization is needed

Feedback
Data Sensor

• Collect QoS data, demand of hardware and software primitives etc.

• Store historical data for simulating the physical system.
Architectural Solution

1. new data
2. historical data
3. primitives
4.1. fetch new data to update model
4.2. if optimization is needed
5. temporally decided provision

15
Primitive Selector

• Apply general techniques from information theory (e.g., Mutual Information)

• Determine *which* and *when* a primitive correlate with a QoS
Architectural Solution

Simulator$_{II}$

Simulator$_{I2}$

Simulator$_{ij}$

Data Sensor

Primitives Selector

QoS Objective Function Trainer

QoS Optimizer

1. new data

2. historical data

3. primitives

4.1. fetch new data to update model

4.2. if optimization is needed

5. temporally decided provision

7. Feedback

S$_{II}$

S$_{I2}$

S$_{ij}$
QoS Objective Function Trainer

• Apply machine learning techniques (e.g., Neural Network)

• Determine how a primitive correlate with a QoS
Architectural Solution

1. new data
2. historical data
3. primitives
4.1. fetch new data to update model
4.2. if optimization is needed
5. temporally decided provision

7. Feedback

$S_{11}$

$S_{12}$

$S_{ij}$

Data Sensor

Primitives Selector

QoS Objective Function Trainer

QoS Optimizer
QoS Optimizer

• Apply evolutionary techniques (e.g., Metaheuristics)

• Search for the near optimal trade-off decision of adaptation, based on the previously defined objective functions
**Architectural Solution**

1. **New data**
   - Data Sensor

2. **Historical data**
   - Primitives Selector

3. **Primitives**
   - QoS Objective Function Trainer

4. **Fetch new data** (to update model)
   - QoS Optimizer

5. **Temporally decided provision**

6. **Feedback**
Conclusion

- We motivate the needs for adaptively optimizing QoS of cloud-based DDDAS.
- We outline our objective and the challenges.
- We describe our DDDAS based architectural solution and the potential techniques that can be applied within the architecture.
Thank You!

Questions?

Tao Chen, Rami Bahsoon
txc919, r.bahsoon@cs.bham.ac.uk
School of Computer Science
University of Birmingham, UK

Georgios Theodoropoulos
georgios.theodoropoulos@durham.ac.uk
School of Engineering and Computer Sciences
Durham University, UK