Automatic Scaling for Cloud Web Applications in Practice

Corina Stratan, Guillaume Pierre, Hector Fernandez, Eliana Tirsa and Valentin Cristea

Vrije Universiteit Amsterdam and University Politehnica of Bucharest

Other versions for the title:

A Practical Approach for Automatic Scaling (Resource Provisioning?) of Web Applications in the Cloud

Implementing Automatic Scaling for Cloud Web Applications

Introduction

Why is automatic scaling in the cloud important/useful? (give a real-world example)

Challenges:

- web applications have multiple tiers with different performance models and requirements
- cloud resources are often heterogeneous
- the behaviour of the servers might change in time, either due to changes in the type of workload or to VM migration
- it is difficult to estimate what will be the performance effect of provisioning and de-provisioning the application with resources

The automatic scaling mechanisms currently available in clouds are based on simple rules and do not take all these issues into account. Our solution aims to address them as follows:

- monitoring tier-specific metrics like request rate and response time
- maintaing a performance profile for each VM instance, that is updated dynamically
- a provisioning mechanism that uses the performance profiles to estimate the potential effect of adding and removing resources
- load balancing with dynamically adjusted weights

Our solution was tested with a real-world application and access traces (Wikibench).

System overview

- performance profiling (offline and online)
- dynamic weighted load balancing
- resource provisioning mechanism

Implementation in ConPaaS

- ConPaaS overview
- Ganglia integration
- the provisioning / profiling manager

The Wikipedia workload

Describe the application and the workload traces. Explain the particularities of this application, like:

- the PHP pages take significantly more time to process than the web pages
- each PHP page needs a lot of DB queries

- the pages vary in complexity so it is difficult to make predictions (maybe compute the standard deviation of the response time from a trace to show this)

Experiments

1. Basic provisioning (similar to the currently available mechanisms in clouds): experiment on the DAS-3 nodes

2. Heterogeneity-aware provisioning: experiment on the UPB nodes with both the basic and heterogeneity-aware provisioning. Compare them in terms of:

- SLA compliance (for example the percentage of requests that met the SLA?)
- amount of used resources

Conclusion

?

Acknowledgments

This work is partially funded by the FP7 Programme of the European Commission in the context of the Contrail project under Grant Agreement FP7-ICT-257438.

Probably also mention ERRIC here.

References