Performance Modelling Text Mining Applications

Daniel Schragl, Abelardo Pardo
School of Electrical & Information Engineering
The University of Sydney
Australia, NSW 2006
Email: daniel.schragl@sydney.edu.au, abelardo.pardo@sydney.edu.au

***Abstract*—Computationally expensive applications, like machine learning applications, can cause scalability issues when moving them to the cloud. Spreading the application across several nodes may help, but the required hardware can not be estimated in advance. Potentially resulting in high costs, as cloud computing resources have to be paid for. This paper identifies a gap in current research, and proposes deducting a methodology for developing performance models for text mining applications by developing such a performance model for a given text mining application. Further, the paper proposes validating the resulting performance model with data from load tests of several deployments, including a single server, a local cluster of servers with a load balancer, and several types of clouds (IaaS–cloud (Amazon EC2) and PaaS–cloud (Google App Engine)), allowing to assess the suitability of these deployments for text mining applications and the model's accuracy. Additionally, it shall be demonstrated how the derived model can be used to predict the performance and scalability limits of the modelled application.**

I. Introduction

Computationally expensive applications, like machine learning applications and certain automated feedback systems, can cause scalability issues when moving them to the cloud. Although, distributing an application across multiple nodes may help, this can cause high costs, as the required hardware can not be estimated in advance. Additionally, some applications can not be clustered easily or only to a certain extent, due to their architecture. Therefore, such applications may reach scalability limits faster than expected.

This paper identifies a gap in current research, and proposes deducting a methodology for developing performance models for text mining applications by developing such a performance model for a given automated feedback and text mining application, called Glosser [14]. Further, the paper proposes validating the resulting performance model with data from load tests of several deployment architectures, including a single server, a local cluster of servers with a load balancer, and several types of clouds (IaaS–cloud (Amazon EC2) and PaaS–cloud (Google App Engine)). This allows assessing the model's accuracy as well as the capabilities of these architectures and their suitablity for text mining applications.

The derived performance model shall be able to discover scalability and performance issues for a given text mining application, before that application is deployed to a cloud, allowing to estimate its future performance capabilities and resource demands and to address potential scalability issues in advance if needed.

II. Related Work

Over the years performance models have been created for many real–world applications, for instance for client/server systems [12], e–business systems [8], online auctioning sites [11][1][13], and large–scale key–value stores [2]. However, not for machine learning or automated feedback systems. Although, it could be argued that modelling techniques are general and apply to any application, we believe that creating specific performance models for text mining applications will lead to models that provide better insights into the modelled applications than standard performance models, resulting in models with higher accuracy. Such models are particularly in need nowadays, as the number and complexity of machine learning applications is increasing rapidly and cloud computing is spreading, adding a more visible cost to computing power than internal data centres ever did.

Therefore, this paper proposes addressing this gap in research through deriving a methodology for creating performance models for text mining systems, by developing such a performance model for a given text mining application, verifying that model for several deployment architectures, as well as showing how it can be used to predict the performance and scalability limits of the modelled application. Additionally, the paper proposes investigating the performance and capabilities of these deployment architectures as well as their suitability for text mining applications.

III. Performance Models

There are many types of models and they all have different purposes [9]. Before a model can be created the purpose of the model has to be decided: What aspects of the system should that model be able to describe or predict? What type of questions about a system should they answer? And how much work can be spent on creating the model?

While some models (e. g., Software Performance Engineering, UML Performance Simulator) are applied in the design phase of a system to evaluate architecture alternatives and rely on estimates, others, including Capacity Planning (CP), are applied to systems after they have been built and rely on measurements of the application under test [7]. Surveys of performance modelling techniques can be found in [7][3][6].

We chose to create a performance model based on the Capacity Planning (CP) methodology [9], which is typically applied to systems after they have been built and relies on measurements of the application under test [7], because they promise a better accuracy than models that are created in the design phase of a system, and a reasonable trade–off between work spent on creating them and achievable accuracy.

IV. Proposed Approach

A methodology for creating performance models for text mining systems based on CP shall be derived, by developing such a performance model for a given text mining application, called Glosser. The proposed steps are: assessing Glosser's capabilities through a series of load tests, establishing its performance profile (classifying its scalability problem space), performing a workload characterization, and deriving a performance model. Then verifying that model with data from load tests of Glosser in varies deployment architectures, including a single server, a local cluster of servers with a load balancer, and several types of clouds (IaaS–cloud (Amazon EC2) and PaaS–cloud (Google App Engine)), and comparing the results with the predictions. Then the model's accuracy as well as the performance and capabilities of these deployments and their suitability for text mining applications shall be analyzed.

V. Progress So Far

Glosser was set up on a single server, a local cluster with a load balancer, as well as on the Amazon EC2 cloud. A distributed load testing script has been developed, which simulates clients accessing Glosser through a browser creating load on the server under test, while spreading its testing load across several test clients. System monitoring tools have been deployed on the server nodes and all test clients. Glosser's logging has been modified to add additional performance related output (like duration needed for calls to machine learning libraries, as well as total CPU user time and total CPU system time required to complete the call). Additionally, several load test series have been performed and first results gathered. While we were able to extract some of the results, we are still working on improving the Python scripts we use to analyze the various log files generated during these test runs (e. g., Tomcat log files, test script log files, system monitoring logs). As soon as we have more detailed results we will be publishing them in a longer paper.

VI. About Glosser

Glosser [14][5] is an existing open source automated feedback system, which generates feedback for written assignments of university students. In some courses students are asked to write their assignments with Google Docs. Whenever a student or examiner wants they can use the Glosser website to analyze a given Google Doc document. Glosser then retrieves that document, indexes it with Lucene and MySQL, and processes it using several natural language processing and machine learning libraries, like WEKA, carrot2, Stanford Parser, and TML [4].

The workload Glosser produces on the underlying server is full with peaks, ranging from long spans with no load at all when idle to maximum CPU usage of all cores. A single request to one of the features of Glosser can create high CPU usage for several minutes. Multiple simultaneous requests cause contention and can greatly reduce the response time of the system. The RAM usage does not tend to change rapidly, except for when Java's garbage collector (GC) is executed. After a GC run the memory usage often drops steeply to a low from where it slowly recovers to the original level over time. Glosser's usage of the underlying database MySQL and the underlying search engine Lucene is infrequent, so that a lot of its computations are CPU bound. The underlying storages, MySQL and Lucene, are merely used to save and retrieve documents so they do not have to be fetched from the Google servers each time (thus, only new revisions and new documents are retrieved from Google). The proposed performance model for Glosser will allow more detailed conclusions about Glosser's scalability and performance limitations.

VII. Conclusion

This paper proposes deriving a methodology for creating a performance model for a given text mining application, by developing such a model for an existing automated feedback and text mining application. Such models are particularly in need nowadays, as the number and complexity of machine learning applications is increasing rapidly and cloud computing is spreading, adding a more visible cost to computing power than internal data centres ever did. Further, this paper proposes verifying the accuracy of the resulting model with data from load tests of several deployment architectures, including a server, a local cluster, and several cloud types (IaaS and PaaS). These results could be used to analyze the model's accuracy, as well as the capabilities and performance of these architectures and their suitability for text mining applications.

References

[1] Akula, V., and Menasce, D., Two-level workload characterization of online auctions, Electronic Commerce Research and Applications, Volume 6, Issue 2, Pages 192–208, 2007.

[2] Atikoglu, Berk, et al., Workload analysis of a large–scale key–value store, ACM SIGMETRICS/PERFORMANCE joint international conference on Measurement and Modeling of Computer Systems, 2012.

[3] Balsamo, S., Di Marco, A., Inverardi, P., Simeoni, M., Model–based performance prediction in software development: a survey IEEE Transactions on Software Engineering, Volume 30, Issue 5, Pages 295–310, 2004.

[4] Garcia Adev, J., Calvo, R., Mining Text with Pimiento, IEEE Journal of Internet Computing, Volume 10, Issue 4, Pages 27–35, 2006.

[5] Glosserproject, Glosser Project Website, http://glosserproject.org, 2013.

[6] Koziolek, H., Performance evaluation of component–based software systems: A survey, Performance Evaluation Journal, Special Issue on Software and Performance, Volume 67, Issue 8, Pages 634–658, 2010.

[7] Martens, A., Koziolek, H., Prechelt, L., Reussner, R., From monolithic to component-based performance evaluation of software architectures, Empirical Software Engineering Journal, Volume 16, Issue 5, Pages 587–622, 2011.

[8] Menasce, D., Almeida, V., Scaling for E–business: technologies, models, performance, and capacity planning, Prentice–Hall, New Jersey, 2000.

[9] Menasce, D., Performance by Design: Computer Capacity Planning by Example, Prentice–Hall, New Jersey, 2004.

[10] Menasce, D., Understanding Cloud Computing: Experimentation and Capacity Planning, Proc. 2009 Computer Measurement Group Conf., 2009.

[11] Menasce, D., and Akula, V., Improving the Performance of Online Auctions Through Server-side Activity–based Caching, World Wide Web, 2007, 2007.

[12] Menasce, D., Analytic performance models for single class and multiple class multithreaded software servers, CMG-CONFERENCE, 2006.

[13] Menasce, D., Akula, V., Evaluating caching policies for online auctions, ACM SIGMETRICS Performance Evaluation Review, 2006.

[14] Villalon, J., Kearney, P., Calvo, R., Reimann, P., Glosser: Enhanced Feedback for Student Writing Tasks, IEEE International Conference on Advanced Learning Technologies (ICALT'08), 2008.