OptiRun
A Platform for Optimized Test Execution in Distributed Environments

MASTER'S THESIS

JANICKE FALCH

JUNE 2016

INTRODUCTION

Our society is becoming increasingly dependent on computers and digital media. The importance of, and demand for, high-quality software has become substantial. Pressure on software vendors to deliver frequent releases of quality software requires efficiency in every stage of the development process. Software testing is an important part of this process, as it serves to provide quality assurance and defect detection as well as ensuring that the test object meets its requirements. With test automation, software testing can be performed rapidly, precisely and repeatedly.

The telecommunications company *Altibox AS* has long wished to incorporate user-level test automation in the testing process of their web application *TV Overalt*, but has failed to make it a priority until now.

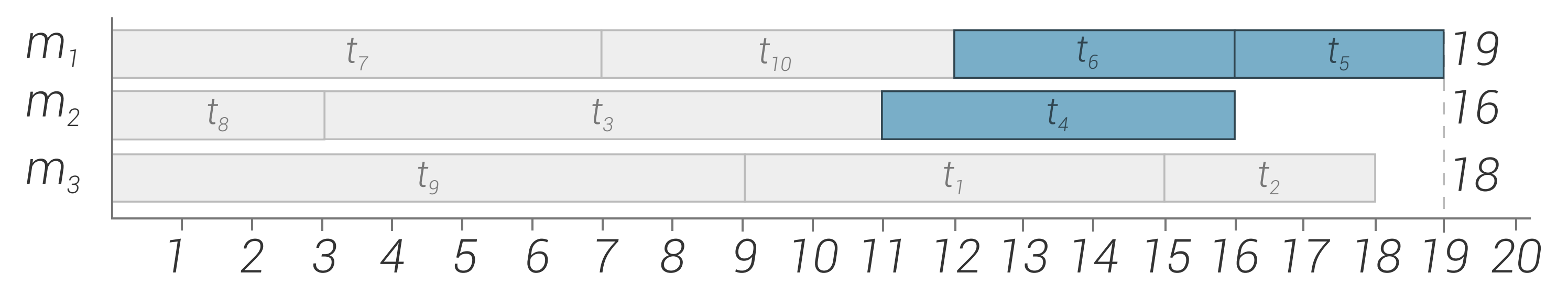
This thesis presents *OptiRun*; a platform where Altibox can run parallel tests in a distributed system. OptiRun is operated from a web-based interface that was developed as part of the thesis. A major objective has been to design and implement *OptiX*; a mechanism for allocating tests to machines in such a way that the overall execution time of a test suite is attempted minimized.

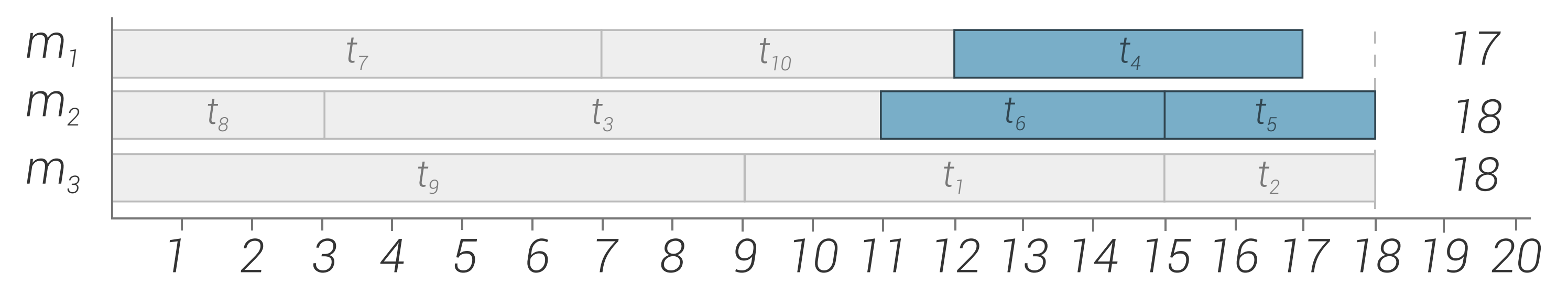
METHODOLOGY

OptiRun consists of two main elements; a controller, which takes care of test allocation, execution and result reporting, and a web application where users can upload and manage test scripts, request test executions, view execution results and report failed test executions to the issue tracking system *JIRA*.

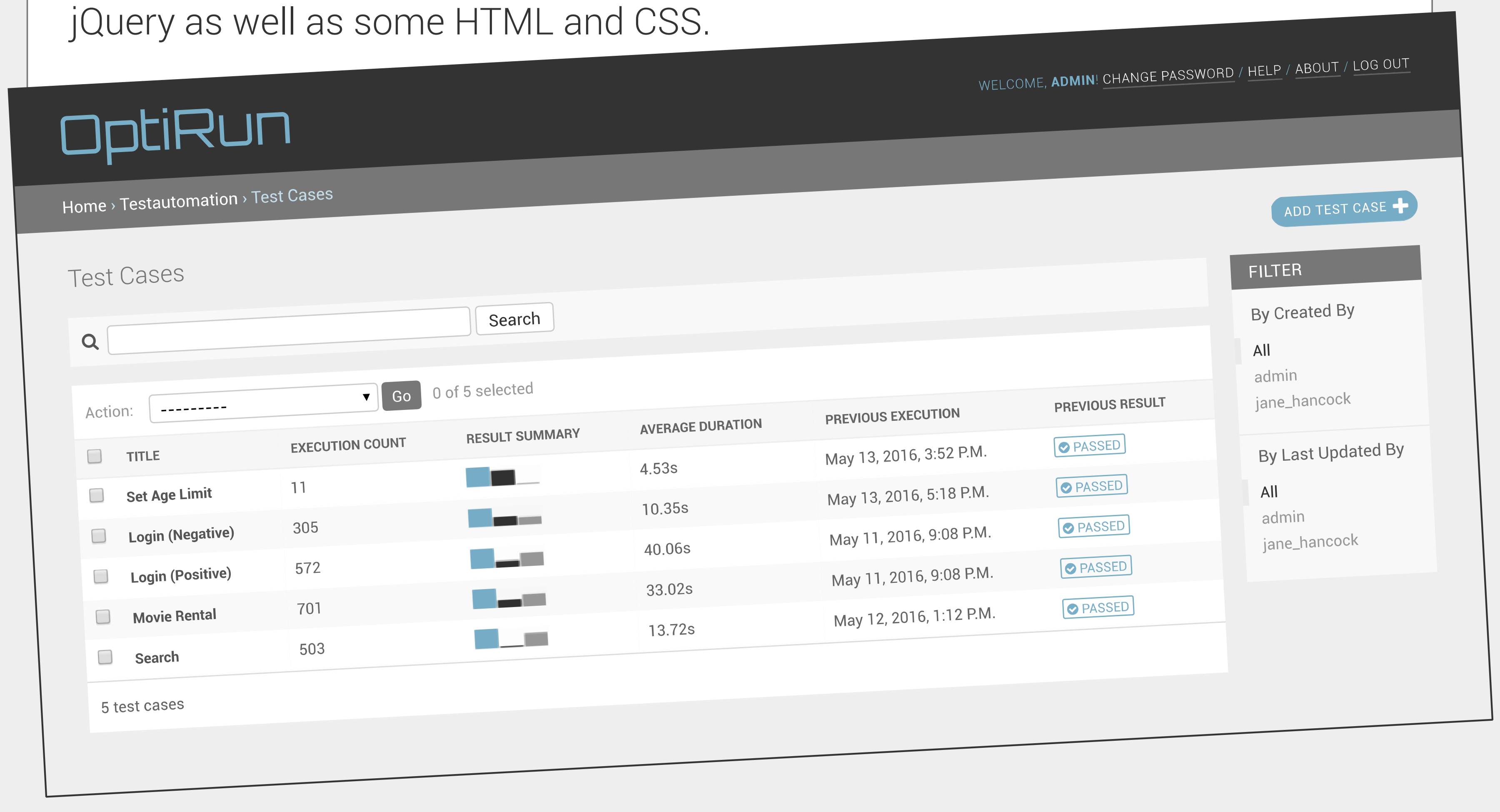
This project was written in the simple, but powerful high-level programming language Python. *Selenium*, a software testing framework for web applications, was used for test execution, and the accompanying *Selenium Grid* was incorporated to enable remote execution in distributed environments.

In order to minimize the overall execution time of a test suite where the tests have resource constraints determining which machines they can be executed on, the tests must be carefully allocated to machines in the distributed system. An allocation mechanism which has been named OptiX takes care of this. The tests are first strategically sorted before being allocated using a greedy algorithm. After the initial allocation, OptiX attempts to improve the result by identifying two subsets of tests currently allocated to two different machines, that when swapped will reduce the overall execution time. This improvement step is repeated until OptiX can no longer find an improvement, or it times out. The time used to allocate the tests is also taken into consideration. The two figures below show the allocation state before and after the conduction of such an improvement in a scenario with three machines and ten tests. Note that the durations of the tests used in this example are artificially shortened compared to what they would normally be in a realistic situation.





OptiRun's web-based user interface was built on the Python Web framework *Django*, which allows for rapid development and seamless interaction with the rest of the system. It was mainly written in Python, but also includes elements of jQuery as well as some HTML and CSS.



Test Cases

0 of 5 selected

TITLE	EXECUTION COUNT	RESULT SUMMARY	AVERAGE DURATION	PREVIOUS EXECUTION	PREVIOUS RESULT
Set Age Limit	11		4.53s	May 13, 2016, 3:52 P.M.	PASSED
Login (Negative)	305		10.35s	May 13, 2016, 5:18 P.M.	PASSED
Login (Positive)	572		40.06s	May 11, 2016, 9:08 P.M.	PASSED
Movie Rental	701		33.02s	May 11, 2016, 9:08 P.M.	PASSED
Search	503		13.72s	May 12, 2016, 1:12 P.M.	PASSED

5 test cases

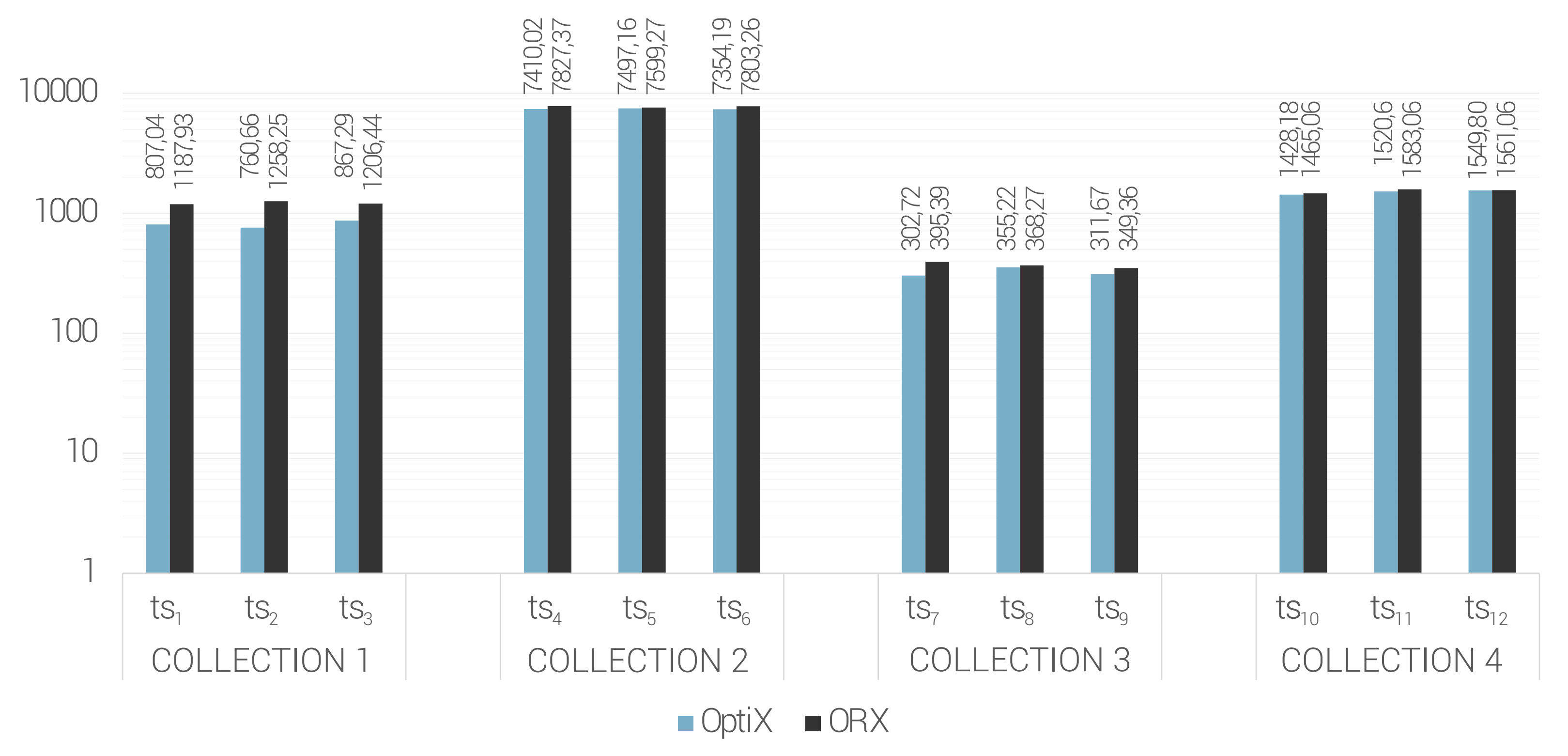
RESULTS

In order to measure and evaluate the performance of the test allocation mechanism OptiX, which represented a major objective in this project, an alternative allocation mechanism was also implemented. This was done using *OR-tools*, Google's library for combinatorial programming and constraint optimization. This alternative version was named *ORX*.

The test data used in the experimental evaluation is divided into four collections, each consisting of three test suites. The test suites in each collection all represent scenarios with a given number of tests and available machines. What separates the test suites in the same collection is the number of machines each test in the test suite can be executed on. This means that there is a varying number of combinatorial solutions to the optimization problem. The test durations and the specific machines each test could be executed on were determined at random. Details about the test data is displayed in the table below:

	# OF TESTS	# OF MACHINES	# OF MACHINES TESTS ARE EXECUTABLE ON
COLLECTION 1	1000	100	ts_1: 100, ts_2: 10, ts_3: *Random*
COLLECTION 2	1000	10	ts_4: 10, ts_5: 5, ts_6: *Random*
COLLECTION 3	200	50	ts_7: 50, ts_8: 10, ts_9: *Random*
COLLECTION 4	200	10	ts_{10}:10, ts_{11}:5, ts_{12}:*Random*

All of the test suites were run with ORX to establish benchmark values, and with OptiX for comparison and evaluation. The results are visualized in the graph below, which shows the combined time used for both allocation and execution. Note that the graph uses a logarithmic scale due to major variation in numbers. As the graph shows, OptiX obtained better results than the benchmark values provided by ORX for all of the 12 test suites.



CONCLUSION

This thesis presents OptiRun; a platform for optimized test execution in distributed environments. OptiRun consists of a controller and a web-based user interface from which the tool can be operated.

OptiX, a mechanism intended for strategically allocating tests with resource constraints to machines in the distributed system, was designed and implemented as part of the thesis. The aim of OptiX is to minimize the overall execution time of test suites. ORX was created as an alternative allocation mechanism. It was built on OR-tools, and was implemented for benchmarking purposes in the evaluation process of OptiX. During the experimental evaluation, OptiX provided better results for all of the test suites.

OptiRun was created to support Altibox in the procedure of incorporating test automation as a practice in the testing process of their online web service TV Overalt.

altibox

University of Stavanger

IPTV Department
Altibox AS

Department of Electrical Engineering and Computer Science
Faculty of Science and Technology
University of Stavanger