Copyright is owned by the Author of the thesis. Permission is given for a copy to be downloaded by an individual for the purpose of research and private study only. The thesis may not be reproduced elsewhere without the permission of the Author.
A NOVEL TOOL FOR RESOURCE UTILISATION REGRESSION TESTING OF JVM-BASED APPLICATIONS

A thesis presented in partial fulfilment of the requirements for the degree of

Master of Science
in
Computer Science

at Massey University, Manawatu, New Zealand

FERGUS HEWSON
2013
Abstract

In recent years, automated regression testing with tools, like JUnit, has become a cornerstone of modern software engineering practice. Automated testing focuses on the functional aspects of software. Automated performance testing is less popular. Where used, it is based on the principle of comparing measurements against static thresholds representing parameters such as system runtime or memory usage.

This thesis presents an alternative approach to automatically generate test oracles from system calibration runs. This approach is particularly useful to safeguard the investment made when software is manually fine-tuned. A proof-of-concept tool was developed that works for all applications that can be deployed on the Java Virtual Machine (JVM), and is capable of testing all properties that can be accessed through Java Management Extensions (JMX) technology.
Acknowledgement

It would not have been possible to write this thesis without the help and support of the people around me, to only some of whom it is possible to give particular mention here. In particular I would like to thanks my supervisors Jens Dietrich (Massey University) and Stephen Marsland (Massey University). Without your guidance and support this thesis would not have been possible.

Jens, your supervision over the past 3 years has been invaluable, your attitude towards Software Engineering is inspiring and I am thankful you have had you as a supervisor. Stephen, thanks for the encouragement over the past few months and your insights into machine learning and performing academic research.

A big thanks goes to my parents, Jayne and Brian, for supporting me during this thesis, without their help I would not be where I am today. Thanks also goes out to my fellow students in the computer science department at Massey University.
Contents

List of Figures ........................................... 11

List of Tables ........................................... 13

1 Introduction ............................................ 17
  1.1 Software testing ..................................... 18
  1.2 Overview ............................................ 19
  1.3 Motivation Example .................................. 19
  1.4 Research Goals ..................................... 24
  1.5 Road Map for Thesis ............................... 25

2 Existing work .......................................... 27
  2.1 Model-based approaches ............................ 29
  2.2 Measurement-based approaches ................. 30
    2.2.1 Benchmarking ................................. 31
    2.2.2 Profiling ..................................... 32
    2.2.3 Instrumentation ............................... 34
    2.2.4 Performance test drivers .................. 35
  2.3 Summary ............................................ 35

3 Methodology ........................................... 37
  3.1 Test Programs ...................................... 38
    3.1.1 Java reference types ......................... 39
    3.1.2 Object leaker .................................. 40
    3.1.3 Classloading ................................... 41
    3.1.4 Consumer Producer .......................... 43
  3.2 Monitored Data .................................... 45
    3.2.1 Dynamic Properties .......................... 46
    3.2.2 Meta-data Properties ....................... 46
    3.2.3 Notifications ................................. 46
    3.2.4 Custom Properties ......................... 49
  3.3 Analysis ............................................ 49
  3.4 Experimental Design ............................. 50

4 Design ................................................ 53
  4.1 Instrumentation .................................... 53
  4.2 Data Collection ................................... 55
    4.2.1 Data Storage ................................. 56
# Test program code

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>B.1 Java references</td>
<td>119</td>
</tr>
<tr>
<td>B.2 Object Leaker</td>
<td>120</td>
</tr>
<tr>
<td>B.3 Classloading</td>
<td>121</td>
</tr>
<tr>
<td>B.4 Consumer Producer</td>
<td>123</td>
</tr>
<tr>
<td>B.4.1 JMX</td>
<td>124</td>
</tr>
<tr>
<td>B.4.2 domain</td>
<td>126</td>
</tr>
<tr>
<td>B.5 Shut-down thread</td>
<td>128</td>
</tr>
</tbody>
</table>

# Analysis code

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>C.1 Kalman Filter</td>
<td>129</td>
</tr>
<tr>
<td>C.2 Particle filter</td>
<td>130</td>
</tr>
<tr>
<td>C.3 Bounds matcher</td>
<td>132</td>
</tr>
<tr>
<td>C.4 Meta-data classifier</td>
<td>134</td>
</tr>
<tr>
<td>C.5 Notification classifier</td>
<td>135</td>
</tr>
<tr>
<td>C.6 Test-run classifier</td>
<td>137</td>
</tr>
</tbody>
</table>

# Acronyms

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Acronyms</td>
<td>139</td>
</tr>
</tbody>
</table>
List of Figures

1.1 Proposed performance test life-cycle ................................ 20
1.2 Motivation heap-memory-used graphs ............................... 24
3.1 Classloading dependency diagram. ................................. 42
4.1 Design overview. .................................................. 54
4.2 Training sequence diagram. ....................................... 57
4.3 Connection sequence diagram. .................................... 59
4.4 Dynamic-property data flow diagram. .............................. 60
4.5 Affects of smoothing .............................................. 61
4.6 Time-point generation example. .................................. 61
4.7 Notification classification algorithm diagram ..................... 70
5.1 Effects of training-set size ........................................ 82
5.2 Effect of smoothing size on data and generated bounds. ...... 83
5.3 Eden-space memory used graphs for Java reference types .... 86
5.4 Heap-memory committed graphs for Java reference types ..... 86
5.5 Scavenge GC collection count graphs for object leaker test-program. 88
5.6 Markswep GC collection count graphs for object leaker test-program. 90
5.7 Compilation time graphs for class-loading test-program. ...... 91
5.8 Permanent generation used graphs for class loading test program. 93
5.9 Processed job count graphs for consumer producer test-program. 95
5.10 Buffer-size graphs for consumer producer test-program. ...... 96
5.11 Notification classifier results processed job count different example 97
5.12 Notification classifier results processed job count similar example 98
# List of Tables

3.1 Java reference type experiments. ........................................... 39  
3.2 Object leaker experiments .................................................. 40  
3.3 Class loading experiments. ................................................... 43  
3.4 Consumer producer variation points ....................................... 44  
3.5 Consumer producer test program experiments. .......................... 45  
3.6 Default dynamic-properties ................................................... 47  
3.7 Standard meta-data properties .............................................. 48  
4.1 Smoothing tuning test plan .................................................... 61  
4.2 Time-point generation tuning test .......................................... 62  
4.3 Simple classifier test plan. ................................................... 63  
4.4 Kalman filter test plan. ....................................................... 64  
4.5 Particle filter parameters. .................................................... 66  
4.6 Bounds matcher test plan. ..................................................... 67  
5.1 Training tuning test results ................................................... 74  
5.2 Smoothing tuning test results. .............................................. 75  
5.3 Time-point generator tuning test results. ................................ 76  
5.4 Bounds matching tuning test results. .................................... 76  
5.5 Simple classifier average method tuning test results. .................. 77  
5.6 Kalman filter average type tuning test results. ........................ 77  
5.7 Kalman classifier process noise tuning test results. ................... 78  
5.8 Default parameter values for particle filter tuning tests. ............. 78  
5.9 Particle filter iterations tuning test results. ........................... 78  
5.10 Particle classifier re-sample percentage tuning test results .......... 79  
5.11 Particle classifier propagate method tuning test results. .............. 79  
5.12 Particle classifier weighting method tuning test results. ............ 79  
5.13 Particle classifier particle count tuning test results. ................. 80  
5.14 Overall classification results .............................................. 81  
5.15 Heap-memory-used rankings aggregated across all test programs. ... 82  
5.16 Java references test-program overall rankings. ........................ 84  
5.17 Eden space memory used rankings for Java references test program. 85  
5.18 Java reference types Eden space memory used example result. ...... 85  
5.19 Heap-memory committed rankings for Java reference test program. 85  
5.20 Java reference types heap-memory-used example result. ............. 87  
5.21 Object leaker test-program overall rankings. .......................... 87  
5.22 Scavenge GC collection count rankings for object leaker test pro-  

gram. ................................................................. 88  
5.23 Object leaker scavenge GC collection count experiment result. ... 89
5.24 Marksweep GC collection count rankings for object leaker test-program. ............................... 89
5.25 Object leaker Marksweep GC collection count experiment result. ................................. 90
5.26 Class-loading overall rankings. .......................................................... 90
5.27 Compilation time rankings for class-loading test program. ............................................ 91
5.28 Class-loading compilation time experiment result. ......................................................... 92
5.29 Permanent generation memory used rankings for class-loading test-program. ......................... 92
5.30 Class-loading permanent generation used experiment result. ........................................... 93
5.31 Consumer producer test-program overall rankings. ....................................................... 94
5.32 Processed job count rankings for consumer producer test-program. ................................. 94
5.33 Consumer producer processed job count example result. .............................................. 95
5.34 Buffer size rankings for consumer producer test-program. ............................................ 96
5.35 Consumer producer buffer size experiment result. ......................................................... 96
5.36 Notification classifier positive data classification from different example ............................ 98
5.37 Notification classifier negative data results from different example ............................... 98
5.38 Notification classifier results for different notifications rate. ........................................ 99
5.39 Positive data results from notification classifier similar example. ..................................... 99
5.40 Negative data results from notification classifier similar example. ................................... 99
5.41 Notification classifier results for similar notifications rate. .............................................. 99
Listings

1.1 Motivation example ................................................. 22
1.2 Object Manager Interface ........................................... 22
1.3 Strong object manager implementation ............................. 23
1.4 Soft object manager implementation ............................... 23
4.1 Simple classification algorithm .................................... 63
B.1 Weak object manager implementation ............................... 119
B.2 Object leaker test program ......................................... 120
B.3 Class loading test program runner ................................ 121
B.4 Class loading test domain object interface ......................... 121
B.5 Class loading test program domain object ......................... 121
B.6 Class loading test program domain object factory ............... 122
B.7 Class loading configuration object ................................ 122
B.8 Consumer Producer Algorithm ..................................... 123
B.9 Consumer object definition ....................................... 123
B.10 Producer object definition ....................................... 124
B.11 Interface for ProductionMXBean .................................. 124
B.12 Interface for CounterMXBean .................................... 124
B.13 ProductionMBeanInfo object definition ......................... 125
B.14 Counter object definition ...................................... 125
B.15 Student object definition ....................................... 126
B.16 StudentFactory object definition ................................ 127
B.17 WebSiteBuilder object definition ................................ 127
B.18 Shutdown Thread object definition ............................... 128
C.1 The Kalman Filter classification algorithm ....................... 129
C.2 Method used to calculate the process noise for the Kalman filter. 130
C.3 The Particle Filter classification algorithm ...................... 130
C.4 Effective sample size evaluation method .......................... 131
C.5 Method used to re-sample particles with respect to their weightings. 131
C.6 Method used to update the particles weight after each iteration. 132
C.7 Method used to update particle weights .......................... 132
C.8 Simple Bounds Matcher Algorithm, matches time-stamps to the nearest bounds. ........................................ 132
C.9 Interpolation Bounds Matcher Algorithm, uses interpolation to generate bounds for a given time-stamp based on the bounds the time-stamp falls between. ........................................ 133
C.10 Method to interpolate a Y value between two points given an X value. ........................................ 133
C.11 Meta-data training algorithm .................................... 134
C.12 Meta-data classification algorithm ................................ 135
C.13 Notification Training Algorithm .................................. 135
C.14 Notification Classification Algorithm . . . . . . . . . . . . . . . . 136
C.15 Test-Run Classifier Training Algorithm . . . . . . . . . . . . . . . 137
C.16 Test-Run Classification Algorithm . . . . . . . . . . . . . . . . . 137