An Empirical Study of Incorporating Cost into Test Suite Reduction and Prioritization

‡Adam M. Smith and †Gregory M. Kapfhammer

‡Department of Computer Science
University of Pittsburgh

†Department of Computer Science
Allegheny College

ACM Symposium on Applied Computing
March 8 - 12, 2009

Featuring an image from www.campusbicycle.com
Implement and empirically evaluate the efficiency and effectiveness of cost-aware greedy methods for regression test suite reduction and prioritization.
Important Contributions

Regression Testing Techniques

Implement and empirically evaluate the efficiency and effectiveness of cost-aware greedy methods for regression test suite reduction and prioritization.
Implement and empirically evaluate the efficiency and effectiveness of cost-aware greedy methods for regression test suite reduction and prioritization.
Implement and empirically evaluate the efficiency and effectiveness of cost-aware greedy methods for regression test suite reduction and prioritization.
Regression Testing and Bicycles

Efficiency: Low wind resistance and time to destination
Effectiveness: Transports all required materials and no break downs
Regression Testing and Bicycles

Cost: Frame material and components cause price to vary considerably
Regression Testing Techniques

Before | After
---|---
Reduction Prunes the Test Suite

Prioritization Reorders the Tests

It is **expensive** to run a test suite $T = \langle T_1, \ldots, T_n \rangle$. Reduction discards some of the $n$ tests in an attempt to **decrease** testing time while still preserving objectives like **coverage** or **fault detection**.
It is **expensive** to run a test suite $T = \langle T_1, \ldots, T_n \rangle$. **Reduction** discards some of the $n$ tests in an attempt to **decrease** testing time while still **preserving** objectives like **coverage** or **fault detection**.
Regression Testing Techniques

Before

After

Reduction Prunes the Test Suite

Prioritization Reorders the Tests

It is **expensive** to run a test suite $T = \langle T_1, \ldots, T_n \rangle$. Reduction discards some of the $n$ tests in an attempt to **decrease** testing time while still **preserving** objectives like **coverage** or **fault detection**.
It is **expensive** to run a test suite $T = \langle T_1, \ldots, T_n \rangle$. **Reduction** discards some of the $n$ tests in an attempt to **decrease** testing time while still **preserving** objectives like **coverage** or **fault detection**.
Regression Testing Techniques

Before | After
---|---
Reduction Prunes the Test Suite
Prioritization Reorders the Tests

It is **expensive** to run a test suite $T = \langle T_1, \ldots, T_n \rangle$. **Prioritization** searches through the $n! = n \times (n-1) \times \ldots \times 1$ orderings for those that **maximize** an objective function like **coverage** or **fault detection**.
Finding the Overlap in Coverage

- $R_j \rightarrow T_i$ means that requirement $R_j$ is covered by test $T_i$
- Test suite reduction discards the test cases that redundantly cover the test requirements
- $T = \langle T_2, T_3, T_6, T_9 \rangle$ covers all of the test requirements
Finding the Overlap in Coverage

- \( R_j \rightarrow T_i \) means that requirement \( R_j \) is covered by test \( T_i \)
- Test suite reduction discards the test cases that redundantly cover the test requirements
- \( T = \langle T_2, T_3, T_6, T_9 \rangle \) covers all of the test requirements
Finding the Overlap in Coverage

- $R_j \rightarrow T_i$ means that requirement $R_j$ is covered by test $T_i$
- Test suite reduction discards the test cases that redundantly cover the test requirements
- $T = \langle T_2, T_3, T_6, T_9 \rangle$ covers all of the test requirements
Greedy Approaches to Regression Testing

Original Test Suite

\[ T_1 \quad T_2 \quad T_3 \quad T_4 \]

Reduction Technique

First Output

\[ T_1 \quad T_4 \]

First Residual

\[ T_2 \quad T_3 \]

Second Output

\[ T_2 \quad T_3 \]

Prioritized Test Suite

\[ T_1 \quad T_4 \quad T_2 \quad T_3 \]

- Harrold, Gupta, Soffa (HGS)
- Delayed Greedy (DGR)
- Traditional Greedy (GRD)
- 2-Optimal Greedy (2OPT)

Hypothesis: Using the execution time of a test case can improve the reduced and prioritized test suites

Compare (i) greedy choices (cost, coverage, and ratio) and (ii) algorithms
Greedy Approaches to Regression Testing

Original Test Suite

- $T_1$
- $T_2$
- $T_3$
- $T_4$

Reduction Technique

First Output

- $T_1$
- $T_4$

First Residual

- $T_2$
- $T_3$
- $T_2$

Second Output

- $T_2$
- $T_3$

Prioritized Test Suite

- $T_1$
- $T_4$
- $T_2$
- $T_3$

- Harrold, Gupta, Soffa (HGS)
- Delayed Greedy (DGR)
- Traditional Greedy (GRD)
- 2-Optimal Greedy (2OPT)

Hypothesis: Using the execution time of a test case can improve the reduced and prioritized test suites

Compare (i) greedy choices (cost, coverage, and ratio) and (ii) algorithms
Greedy Approaches to Regression Testing

Original Test Suite

\[ T_1 \quad T_2 \quad T_3 \quad T_4 \]

Reduction Technique

First Output

\[ T_1 \quad T_4 \]

First Residual

\[ T_2 \quad T_3 \]

Second Output

\[ T_2 \quad T_3 \]

Prioritized Test Suite

\[ T_1 \quad T_4 \quad T_2 \quad T_3 \]

- Harrold, Gupta, Soffa (HGS)
- Delayed Greedy (DGR)
- Traditional Greedy (GRD)
- 2-Optimal Greedy (2OPT)

Hypothesis: Using the execution time of a test case can improve the reduced and prioritized test suites

Compare (i) greedy choices (cost, coverage, and ratio) and (ii) algorithms

An Empirical Study of Incorporating Cost into Test Suite Reduction and Prioritization
Greedy Approaches to Regression Testing

Original Test Suite

\[ T_1 \quad T_2 \quad T_3 \quad T_4 \]

Reduction Technique

First Output

\[ T_1 \quad T_4 \]

First Residual

\[ T_2 \quad T_3 \]

Second Output

\[ T_1 \quad T_4 \quad T_2 \quad T_3 \]

Prioritized Test Suite

Hypothesis: Using the execution time of a test case can improve the reduced and prioritized test suites

- Harrold, Gupta, Soffa (HGS)
- Delayed Greedy (DGR)
- Traditional Greedy (GRD)
- 2-Optimal Greedy (2OPT)

Compare (i) greedy choices (cost, coverage, and ratio) and (ii) algorithms
Evaluating Test Suite Prioritizers

Prioritize to increase the CE of a test suite $CE = \frac{Actual}{Ideal} \in [0, 1]$
Evaluating Test Suite Prioritizers

 Covered Test Reqs $C(T, t)$

 Testing Time $(t)$

 $T_1$ Done  $T_{n-1}$ Done

 $\text{Cover } R(T_1)$

 $\bigcup_{i=1}^{n-1} R(T_i)$

 Cover $R(T)$

 $T_n$ Done

 Area $\int_0^{t(n)} C(T, t)$

 Prioritize to increase the CE of a test suite $CE = \frac{\text{Actual}}{\text{Ideal}} \in [0, 1]$
Evaluating Test Suite Reducers

Reduction Factor for Size (RFFS): How small is the reduced test suite?
Evaluating Test Suite Reducers

Reduction Factor for Time (RFFT): How fast is the reduced test suite?
Greedy Choices Impact Effectiveness

<table>
<thead>
<tr>
<th></th>
<th>$R_1$</th>
<th>$R_2$</th>
<th>$R_3$</th>
<th>$R_4$</th>
<th>$R_5$</th>
<th>Execution Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>$T_1$</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>4</td>
</tr>
<tr>
<td>$T_2$</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>$T_3$</td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>$T_4$</td>
<td>✓</td>
<td></td>
<td></td>
<td>✓</td>
<td>✓</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Greedy-by</th>
<th>$T_r$</th>
<th>$time(T_r)$</th>
<th>$T_p$</th>
<th>CE</th>
</tr>
</thead>
<tbody>
<tr>
<td>coverage</td>
<td>$\langle T_1, T_4 \rangle$</td>
<td>5</td>
<td>$\langle T_1, T_4, T_2, T_3 \rangle$</td>
<td>0.400</td>
</tr>
<tr>
<td>time</td>
<td>$\langle T_2, T_3, T_4 \rangle$</td>
<td>3</td>
<td>$\langle T_2, T_3, T_4, T_1 \rangle$</td>
<td>0.714</td>
</tr>
<tr>
<td>ratio</td>
<td>$\langle T_2, T_4, T_3 \rangle$</td>
<td>3</td>
<td>$\langle T_2, T_4, T_3, T_1 \rangle$</td>
<td>0.743</td>
</tr>
</tbody>
</table>
Greedy Choices Impact Effectiveness

<table>
<thead>
<tr>
<th></th>
<th>$R_1$</th>
<th>$R_2$</th>
<th>$R_3$</th>
<th>$R_4$</th>
<th>$R_5$</th>
<th>Execution Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>$T_1$</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>4</td>
</tr>
<tr>
<td>$T_2$</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>$T_3$</td>
<td>✓</td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>$T_4$</td>
<td>✓</td>
<td></td>
<td>✓</td>
<td></td>
<td>✓</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Greedy-by</th>
<th>$T_r$</th>
<th>$time(T_r)$</th>
<th>$T_p$</th>
<th>CE</th>
</tr>
</thead>
<tbody>
<tr>
<td>coverage</td>
<td>$\langle T_1, T_4 \rangle$</td>
<td>5</td>
<td>$\langle T_1, T_4, T_2, T_3 \rangle$</td>
<td>0.400</td>
</tr>
<tr>
<td>time</td>
<td>$\langle T_2, T_3, T_4 \rangle$</td>
<td>3</td>
<td>$\langle T_2, T_3, T_4, T_1 \rangle$</td>
<td>0.714</td>
</tr>
<tr>
<td>ratio</td>
<td>$\langle T_2, T_4, T_3 \rangle$</td>
<td>3</td>
<td>$\langle T_2, T_4, T_3, T_1 \rangle$</td>
<td>0.743</td>
</tr>
</tbody>
</table>
## Greedy Choices Impact Effectiveness

<table>
<thead>
<tr>
<th></th>
<th>$R_1$</th>
<th>$R_2$</th>
<th>$R_3$</th>
<th>$R_4$</th>
<th>$R_5$</th>
<th>Execution Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>$T_1$</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>4</td>
</tr>
<tr>
<td>$T_2$</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>$T_3$</td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>$T_4$</td>
<td>✓</td>
<td></td>
<td></td>
<td>✓</td>
<td>✓</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Greedy-by</th>
<th>$T_r$</th>
<th>$time(T_r)$</th>
<th>$T_p$</th>
<th>CE</th>
</tr>
</thead>
<tbody>
<tr>
<td>coverage</td>
<td>$\langle T_1, T_4 \rangle$</td>
<td>5</td>
<td>$\langle T_1, T_4, T_2, T_3 \rangle$</td>
<td>0.400</td>
</tr>
<tr>
<td>time</td>
<td>$\langle T_2, T_3, T_4 \rangle$</td>
<td>3</td>
<td>$\langle T_2, T_3, T_4, T_1 \rangle$</td>
<td>0.714</td>
</tr>
<tr>
<td>ratio</td>
<td>$\langle T_2, T_4, T_3 \rangle$</td>
<td>3</td>
<td>$\langle T_2, T_4, T_3, T_1 \rangle$</td>
<td>0.743</td>
</tr>
</tbody>
</table>
Greedy Choices Impact Effectiveness

<table>
<thead>
<tr>
<th></th>
<th>$R_1$</th>
<th>$R_2$</th>
<th>$R_3$</th>
<th>$R_4$</th>
<th>$R_5$</th>
<th>Execution Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>$T_1$</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>4</td>
</tr>
<tr>
<td>$T_2$</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>$T_3$</td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>$T_4$</td>
<td>✓</td>
<td></td>
<td></td>
<td>✓</td>
<td>✓</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Greedy-by</th>
<th>$T_r$</th>
<th>$time(T_r)$</th>
<th>$T_p$</th>
<th>CE</th>
</tr>
</thead>
<tbody>
<tr>
<td>coverage</td>
<td>$\langle T_1, T_4 \rangle$</td>
<td>5</td>
<td>$\langle T_1, T_4, T_2, T_3 \rangle$</td>
<td>0.400</td>
</tr>
<tr>
<td>time</td>
<td>$\langle T_2, T_3, T_4 \rangle$</td>
<td>3</td>
<td>$\langle T_2, T_3, T_4, T_1 \rangle$</td>
<td>0.714</td>
</tr>
<tr>
<td>ratio</td>
<td>$\langle T_2, T_4, T_3 \rangle$</td>
<td>3</td>
<td>$\langle T_2, T_4, T_3, T_1 \rangle$</td>
<td>0.743</td>
</tr>
</tbody>
</table>
## Case Study Applications

An Empirical Study of Incorporating Cost into Test Suite Reduction and Prioritization

| Name | $|T|$ | $|R(T)|$ | CCN | NCSS  |
|------|-----|--------|-----|-------|
| DS   | 110 | 40     | 1.35| 1243.00|
| GB   | 51  | 88     | 2.60| 1455.00|
| JD   | 54  | 783    | 1.64| 2716.00|
| LF   | 13  | 6      | 1.40| 215.00 |
| RM   | 13  | 19     | 2.13| 569.00 |
| SK   | 27  | 117    | 2.00| 628.00 |
| TM   | 27  | 46     | 2.21| 748.00 |
| RP   | 76  | 221    | 2.65| 6822.00|
### Case Study Applications

| Name | | $|T|$ | $|R(T)|$ | CCN | NCSS |
|------|---|---|---|---|---|
| DS   | 110 | 40 | 1.35 | 1243.00 |
| GB   | 51 | 88 | 2.60 | 1455.00 |
| JD   | 54 | 783 | 1.64 | 2716.00 |
| LF   | 13 | 6 | 1.40 | 215.00 |
| RM   | 13 | 19 | 2.13 | 569.00 |
| SK   | 27 | 117 | 2.00 | 628.00 |
| TM   | 27 | 46 | 2.21 | 748.00 |
| RP   | 76 | 221 | 2.65 | 6822.00 |

An Empirical Study of Incorporating Cost into Test Suite Reduction and Prioritization
### Case Study Applications

| Name | $|T|$ | $|\mathcal{R}(T)|$ | CCN  | NCSS  |
|------|-----|----------------|------|-------|
| DS   | 110 | 40            | 1.35 | 1243.00|
| GB   | 51  | 88            | 2.60 | 1455.00|
| JD   | 54  | 783           | 1.64 | 2716.00|
| LF   | 13  | 6             | 1.40 | 215.00 |
| RM   | 13  | 19            | 2.13 | 569.00 |
| SK   | 27  | 117           | 2.00 | 628.00 |
| TM   | 27  | 46            | 2.21 | 748.00 |
| RP   | 76  | 221           | 2.65 | 6822.00|

An Empirical Study of Incorporating Cost into Test Suite Reduction and Prioritization
### Case Study Applications

| Name | $|T|$ | $|R(T)|$ | CCN | NCSS   |
|------|-----|--------|-----|--------|
| DS   | 110 | 40     | 1.35| 1243.00|
| GB   | 51  | 88     | 2.60| 1455.00|
| JD   | 54  | 783    | 1.64| 2716.00|
| LF   | 13  | 6      | 1.40| 215.00 |
| RM   | 13  | 19     | 2.13| 569.00 |
| SK   | 27  | 117    | 2.00| 628.00 |
| TM   | 27  | 46     | 2.21| 748.00 |
| RP   | 76  | 221    | 2.65| 6822.00|

**Questions:** Do the **greedy** reducers and prioritizers efficiently identify test suites that **improve** effectiveness? What are the fundamental **trade-offs**?
The myopic focus on cost leads to low RFFT values for 2OPT and GRD.
Overview of RFFT Trends

The myopic focus on **cost** leads to **low** RFFT values for 2OPT and GRD
Overview of RFFS Trends

Reduction Factor for Size (RFFS)

DGR and HGS are the best at creating test suites that improve RFFS.
Overview of RFFS Trends

DGR and HGS are the best at creating test suites that improve RFFS.
Overview of CE Trends

Coverage Effectiveness (CE)

alg: HGS

metric: coverage

0.7520

0.8231

alg: DGR

0.8344

0.9388

Using ratio and cost improves the CE of the prioritized test suite.
Overview of CE Trends

Using **ratio** and **cost** improves the CE of the prioritized test suite
An Empirical Study of Incorporating Cost into Test Suite Reduction and Prioritization

For 2OPT and GRD, ratio and coverage create the best test suites.
An Empirical Study of Incorporating Cost into Test Suite Reduction and Prioritization

For 2OPT and GRD, ratio and coverage create the best test suites
It is often easy to construct test suites with high RFFS values.
Reduction Factor for Size - SK

It is often easy to construct test suites with high RFFS values.
DGR and HGS exhibit lackluster performance when reordering.
DGR and HGS exhibit lackluster performance when reordering
For the chosen case study applications, the techniques are efficient.
Efficiency Measurements

For the chosen case study applications, the techniques are efficient.
Use *mutation* and *real* faults to support the calculation of fault detection effectiveness (FDE) and average percentage of faults detected (APFD). Consider *search-based* testing methods.
Software developers use testing to raise confidence in the correctness of a software system. Automated reduction and prioritization techniques attempt to decrease the time required to detect faults during test suite execution. This package uses the Harald Gupta Softa, delayed greedy, traditional greedy, and 2-optimal greedy algorithms for both test suite reduction and prioritization. Even though reducing and reordering a test suite is primarily done to ensure that testing is cost-effective, these algorithms are normally configured to make greedy choices with coverage information alone. This paper extends these algorithms to greedily reduce and prioritize the tests by using both test cost (e.g., execution time) and the ratio of code coverage to test cost. An empirical study with eight:

http://raise.googlecode.com/ provides tools, data sets, and resources
Concluding Remarks

Regression Testing Techniques

- **Implementation** and empirical **evaluation** of methods for test suite reduction and prioritization
- Freely available **data sets** and free/open source **tools**

http://www.cs.allegheny.edu/~gkapfham/research/kanonizo/