Towards Regression Testing for Database Applications

Gregory M. Kapfhammer†

Department of Computer Science
Allegheny College, Pennsylvania, USA
http://cs.allegheny.edu/~gkapfham/

ASTReNet and SOSoRNet, King’s College London, 2007

† In Conjunction with Mary Lou Soffa (UVa/CS), Panos Chrysanthis (Pitt/CS), Bruce Childers (Pitt/CS)
A regression testing framework for traditional database applications. Future research includes service-oriented applications that use Grid-enabled databases.
An Interesting Defect Report

Database Server Crashes

When you run a complex query against Microsoft SQL Server 2000, the SQL Server scheduler may stop responding. Additionally, you receive an error message that resembles the following: **Date Time server Error: 17883 Severity: 1, State: 0 Date Time server Process 52:0 (94c) ...**

An Input-Dependent Defect

This problem occurs when one or more of the following conditions are true: The query contains a `UNION` clause or a `UNION ALL` clause that affects many columns. The query contains several `JOIN` statements. The query has a large estimated cost. **BUG 473858 (SQL Server 8.0)**
An Interesting Defect Report

Database Server Crashes

When you run a complex query against Microsoft SQL Server 2000, the SQL Server scheduler may stop responding. Additionally, you receive an error message that resembles the following: **Date Time server Error: 17883 Severity: 1, State: 0 Date Time server Process 52:0 (94c) ...**

An Input-Dependent Defect

This problem occurs when one or more of the following conditions are true: The query contains a **UNION** clause or a **UNION ALL** clause that affects many columns. The query contains several **JOIN** statements. The query has a large estimated cost. **BUG 473858 (SQL Server 8.0)**
Jeppesen reports airspace boundary problems

About 350 airspace boundaries contained in Jeppesen NavData are incorrect, the FAA has warned. The error occurred at Jeppesen after a software upgrade when information was pulled from a database containing 20,000 airspace boundaries worldwide for the March NavData update, which takes effect March 20.
Real World Example

A Severe Defect

The Risks Digest, Volume 22, Issue 64, 2003

**Jeppesen reports airspace boundary problems**

*About 350 airspace boundaries contained in Jeppesen NavData are incorrect, the FAA has warned. The error occurred at Jeppesen after a software upgrade when information was pulled from a database containing 20,000 airspace boundaries worldwide for the March NavData update, which takes effect March 20.*

An Important Point

Practically all use of databases occurs from within application programs [Silberschatz et al., 2006, pg. 311]
Program and Database Interactions

Basic Operation
Program $P$ creates SQL statements in order to view and/or modify the state of the relational database.

SQL Construction
Static analysis does not reveal the exact SQL command since the program constructs the full SQL statement at run-time.
Database Interaction Granularity

Program $P$ interacts with two relational databases $D_k$ and $D_l$ at different levels of granularity (relation, record, attribute, ...)

Gregory M. Kapfhammer
Towards Regression Testing for Database Applications
Overview of the Coverage Monitoring Process

Calculating Coverage

Use instrumentation probes to capture and analyze a program’s interaction with the databases.

Regression Testing

The adequacy measurements can be used to support both test suite reduction and prioritization.
Database-Aware Coverage Trees

**Instrumentation Probes**

Use **static** and **dynamic** (load-time) instrumentation techniques to insert coverage monitoring probes.

**Coverage Trees**

Store the coverage results in a tree in order to support the calculation of many types of coverage (e.g., **data flow** or **call tree**).
## Comparing the Coverage Trees

### Tree Characteristics

<table>
<thead>
<tr>
<th>Tree</th>
<th>DB?</th>
<th>Context</th>
<th>Probe Time</th>
<th>Tree Space</th>
</tr>
</thead>
<tbody>
<tr>
<td>CCT</td>
<td>✗</td>
<td>Partial</td>
<td>Low - Moderate</td>
<td>Low</td>
</tr>
<tr>
<td>DCT</td>
<td>✗</td>
<td>Full</td>
<td>Low</td>
<td>Moderate - High</td>
</tr>
<tr>
<td>DI-CCT</td>
<td>✓</td>
<td>Partial</td>
<td>Moderate</td>
<td>Moderate</td>
</tr>
<tr>
<td>DI-DCT</td>
<td>✓</td>
<td>Full</td>
<td>Moderate</td>
<td>High</td>
</tr>
</tbody>
</table>

### Table Legend

- **Database?** ∈ {✗, ✓}
- **Context** ∈ {Partial, Full}
- **Probe Time Overhead** ∈ {Low, Moderate, High}
- **Tree Space Overhead** ∈ {Low, Moderate, High}
Regression Testing Overview

**Reduction** aims to find a smaller test suite that covers the same requirements as the original suite. **Prioritization** re-orders the tests so that they cover the requirements more effectively.
Finding the Overlap in Coverage

**Test Suite Reduction**

- $R_j \rightarrow T_i$ means that requirement $R_j$ is *covered by* test $T_i$
- $T = \langle T_2, T_3, T_6, T_9 \rangle$ cover all of the test requirements
Measuring Coverage Effectiveness

Prioritize to increase the CE of a test suite \( CE = \frac{Actual}{Ideal} \)
Regression tester uses several algorithms and test requirements.
### Characterizing the Case Study Applications

<table>
<thead>
<tr>
<th>Application</th>
<th># Tests</th>
<th>Test NCSS / Total NCSS</th>
</tr>
</thead>
<tbody>
<tr>
<td>RM</td>
<td>13</td>
<td>227/548 = 50.5%</td>
</tr>
<tr>
<td>FF</td>
<td>16</td>
<td>330/558 = 59.1%</td>
</tr>
<tr>
<td>PI</td>
<td>15</td>
<td>203/579 = 35.1%</td>
</tr>
<tr>
<td>ST</td>
<td>25</td>
<td>365/620 = 58.9%</td>
</tr>
<tr>
<td>TM</td>
<td>27</td>
<td>355/748 = 47.5%</td>
</tr>
<tr>
<td>GB</td>
<td>51</td>
<td>769/1455 = 52.8%</td>
</tr>
</tbody>
</table>
## Details About the Database Interactions

### Static Interaction Counts

<table>
<thead>
<tr>
<th>Application</th>
<th><code>executeUpdate</code></th>
<th><code>executeQuery</code></th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>RM</td>
<td>3</td>
<td>4</td>
<td>7</td>
</tr>
<tr>
<td>FF</td>
<td>3</td>
<td>4</td>
<td>7</td>
</tr>
<tr>
<td>PI</td>
<td>3</td>
<td>2</td>
<td>5</td>
</tr>
<tr>
<td>ST</td>
<td>4</td>
<td>3</td>
<td>7</td>
</tr>
<tr>
<td>TM</td>
<td>36</td>
<td>9</td>
<td>45</td>
</tr>
<tr>
<td>GB</td>
<td>11</td>
<td>23</td>
<td>34</td>
</tr>
</tbody>
</table>

### Dynamic Interaction Counts

Database interactions that occur in **iterative** or **recursive** computations are executed more frequently.
## Reducing the Size of the Test Suite

<table>
<thead>
<tr>
<th>App</th>
<th>Rel</th>
<th>Attr</th>
<th>Rec</th>
<th>Attr Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>RM</td>
<td>(13)</td>
<td>(7, .462)</td>
<td>(7, .462)</td>
<td>(9, .308)</td>
</tr>
<tr>
<td>FF</td>
<td>(16)</td>
<td>(7, .563)</td>
<td>(7, .563)</td>
<td>(11, .313)</td>
</tr>
<tr>
<td>PI</td>
<td>(15)</td>
<td>(6, .600)</td>
<td>(6, .600)</td>
<td>(8, .700)</td>
</tr>
<tr>
<td>ST</td>
<td>(25)</td>
<td>(5, .800)</td>
<td>(5, .760)</td>
<td>(11, .560)</td>
</tr>
<tr>
<td>TM</td>
<td>(27)</td>
<td>(14, .481)</td>
<td>(14, .481)</td>
<td>(15, .449)</td>
</tr>
<tr>
<td>GB</td>
<td>(51)</td>
<td>(33, .352)</td>
<td>(33, .352)</td>
<td>(33, .352)</td>
</tr>
<tr>
<td>All</td>
<td>(24.5)</td>
<td>(12, .510)</td>
<td>(12.17, .503)</td>
<td>(14.667, .401)</td>
</tr>
</tbody>
</table>

- Reduction factor for test suite size varies from .352 to .8
**Reducing the Testing Time**

- **GRO** reduces test execution time even though it removes few tests

---

**Gregory M. Kapfhammer**  
Towards Regression Testing for Database Applications
Preserving Requirement Coverage

- GRO guarantees coverage preservation while the others do not
Improving Coverage Effectiveness

GRO is the best choice and the original ordering is poor.
Use prioritization to reduce testing time by avoiding database restarts
Conclusions and Future Work

Concluding Remarks

- A new **perspective** on software testing and an **efficient** and **effective** method for database-aware regression testing

Future Work

- Challenges associated with grid-enabled databases
- Conduct experiments with larger database applications

Resources

- http://cs.allegheny.edu/~gkapfham/research/diatoms/