Flexible Lightweight Scalable Software Analysis

Rob Hackman
Prof. Jo Atlee
Prof. Mike Godfrey
Davood Anbarnam
model-based feature interaction analysis (past work)
GOAL: system-wide interaction analysis

- No system-wide model
- Heterogeneous components
  - legacy, generated, third-party
  - distributed ECUs
  - bus-based communications
- 100 million lines of code (roughly)
- High variability (SPL)

http://www.flexautomotive.net/EMCFLEXBLOG/post/2015/09/08/can-bus-for-controller-area-network)
flexible, lightweight, scalable software analysis

software artifacts  extractors  lightweight model

Source Code  Fact Extractor  Fact Base  queries  Linked Fact Base (Neo4j)  graph database
Object Code  Fact Extractor  Fact Base
Models  Fact Extractor  Fact Base
Build scripts / Config files  Fact Extractor  Fact Base
Facts  Fact Extractor  Fact Base

Flexible Lightweight Scalable Software Analysis
### facts (entities, relations, attributes)

<table>
<thead>
<tr>
<th><strong>entities</strong></th>
<th><strong>relationships</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>components</td>
<td></td>
</tr>
<tr>
<td>classes</td>
<td>ROS node <strong>receives</strong> message from ROS node</td>
</tr>
<tr>
<td>functions</td>
<td>variable on <strong>RHS</strong> of variable assignment</td>
</tr>
<tr>
<td>variables</td>
<td>function <strong>calls</strong> function</td>
</tr>
<tr>
<td>ROS nodes</td>
<td>function <strong>reads</strong> variable</td>
</tr>
<tr>
<td></td>
<td>function <strong>writes</strong> variable</td>
</tr>
<tr>
<td></td>
<td>class <strong>contains</strong> function</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>attributes</strong></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>function</strong> is a callback</td>
<td></td>
</tr>
<tr>
<td><strong>variable</strong> used in control-flow decision</td>
<td></td>
</tr>
</tbody>
</table>
facts (entities, relations, attributes)

C++ Code

```cpp
class Square {
    public:
        int getArea() {
            int area = size * size;
            return area;
        }
    int size;
};
```

Factbase

- **FACT TUPLE:**
  - $INSTANCE$ Square (class)
  - $INSTANCE$ size (variable)
  - $INSTANCE$ getArea (function)
  - $INSTANCE$ area (variable)

  - contain Square getArea
  - contain Square size
  - contain getArea area

- **FACT ATTRIBUTE:**
  - size { inDecisionCond = false }
  - area { inDecisionCond = false }
facts (graph database)

- **Square**
  - contain
  - contain
  - varAssign

- **getArea**
  - contain
  - write

- **size**
  - \( \text{inDecisionCond} = \text{false} \)

- **area**
  - \( \text{inDecisionCond} = \text{false} \)

**Factbase**

- **FACT TUPLE**:
  - $\text{INSTANCE}$ Square class
  - $\text{INSTANCE}$ size variable
  - $\text{INSTANCE}$ getArea function
  - $\text{INSTANCE}$ area variable

- contain
  - Square getArea
  - Square size
  - getArea area

- varAssign
  - size area

- write
  - getArea area

**FACT ATTRIBUTE**:
- size \( \{ \text{inDecisionCond} = \text{false} \} \)
- area \( \{ \text{inDecisionCond} = \text{false} \} \)
query language (over a graph database)

**General Path Queries**
- Call flows
- Information flows (assignments, parameter passing, message passing)

**Interaction Queries**
- Communication loops (possible delays, nontermination)
- Race conditions (possible nondeterminism)
- Multiple inputs (possible livelock)
- Control-flow alteration (variable assignment flows to control-flow decision)
Demo
scalability (large case study)

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Source Code Files (#)</strong></td>
<td>9,289</td>
</tr>
<tr>
<td><strong>Source code (LoC)</strong></td>
<td>1,475,643*</td>
</tr>
<tr>
<td><strong>Header Files (#)</strong></td>
<td>10,131</td>
</tr>
<tr>
<td><strong>Header Files (LoC)</strong></td>
<td>623,068*</td>
</tr>
<tr>
<td><strong>Generated Nodes (#)</strong></td>
<td>206,531</td>
</tr>
<tr>
<td><strong>Generated Relationships (#)</strong></td>
<td>590,463</td>
</tr>
</tbody>
</table>

*generated using David A. Wheeler's 'SLOCCount'

**Nodes** = “module” = group of files with same prefix
**Edges** = inter-module communication
scalability (large case study)

Example: Control-Flow Alteration Query
Detect whether or not a function call to some function `bar` depends on some global variable `globVar`, which could have been written to by another function `foo`

```c
int globVar;
void foo() {
    . . .
    globVar = . . .;
    . . .
}

. . .
if (globVar) bar();
```
scalability (large case study)

<table>
<thead>
<tr>
<th>Interaction/Behavior Alteration</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interactions among Modules (pairs of components (not paths))</td>
<td>587,998</td>
</tr>
<tr>
<td>Behaviour Alterations among Functions (paths)</td>
<td>6,231,572</td>
</tr>
<tr>
<td>Behaviour Alterations among Functions (paths (without loops))</td>
<td>32,287</td>
</tr>
<tr>
<td>Behaviour Alterations among Functions (paths (without loops &amp; involving more than one module))</td>
<td>27,479</td>
</tr>
</tbody>
</table>
scalability (large case study)

behaviour alterations among functions
paths (without loops & involving more than one module)

• Of the 27,479 detected paths, most are small

• **Triage:** Higher path rate may indicate higher chance of error
current work (precision)

1) Improving the **precision** of the fact extraction and static analysis
current work (scalability)

2) Improving the **scalability** of path queries
current work (user experience)

3) Reporting and **visualizing** query results

4) Improving the **user experience** in making follow-up queries
current work (variability)

5) Investigating **variability**-aware interaction analysis

```java
class BaseFeature { ... };
class FeatureA : public BaseFeature { ... };
class FeatureB : public BaseFeature { ... };

bool config_A; // Given a value somehow

int main () {
    BaseFeature *p;
    if ( config_A ) {
        FeatureA *a = new FeatureA;
        p = a;
    } else {
        FeatureB *b = new FeatureB;
        p = b;
    }
    p->execute ( );
}
```

- extracting the conditions under which facts hold
- querying over configurations of factbases
- exploring configurations of factbases

GOAL: system-wide interaction analysis

- No system-wide model
- Heterogeneous components
  - legacy, generated, third-party
  - distributed ECUs
  - bus-based communications
- 100 million lines of code (roughly)
- High variability (SPL)

flexible, lightweight, scalable software analysis

current work

Improving precision of the extraction and analysis

Improving scalability of path queries

Reporting and visualizing query results

Improving the user experience in making follow-up queries

Supporting variability-aware extraction, analyses, and exploration