Fixing Software Configurations based on Self-Adaptive Priorities

Bo Wang¹, Leonardo Passos², Yingfei Xiong¹, Krzysztof Czarnecki², Haiyan Zhao¹
Wei Zhang¹, and Hong Mei¹

¹Peking University
²University of Waterloo
Modern Software Product Lines

Variability Models

Configuration

Feature Models, Linux Kconfig, eCos CDL, pure::variants, ...

Variability Models
eCos Configurator - Errors

Fixing an error is hard! [Hubaux et al., VaMoS’11]
Existing Approaches Generate Fixes
[Xiong et al., ICSE’12]

[PreloadSize <= 8]
[PoolSize >= 10]
[Preload = false]

Other fix generating approaches: [White et al., SPLC’08] [eCos Configurator]
Problem: Large Fixes

• The fixes are sometimes large in size and number
• Fixes for eCos [Xiong et al. ICSE’12]:
  – Fix lists contain up to nine fixes
  – Fixes change up to nine variables
  – In total, 17% of all fix lists contain changes over more than 10 variables

Users have to read through potentially large lists and decide the most desirable fix
How to guide the users to identify their desirable fixes?
Our Solution

• Use the idea of priority
  – The priority of a variable represents the likelihood of its current value being desirable to the user.

• Two Basic ideas:
  – Generate fixes that only change variables with lower priorities
  – Dynamically adjust the priority of variables through implicit translation of user feedback
Our Contribution

• A priority-based approach to locating a desirable fix through user feedbacks
• An algorithm to implement the approach using any fix generation algorithm
• An empirical evaluation that shows the overall reduction of choices exposed to the user
Our Contribution

• A priority-based approach to locating a desirable fix through user feedbacks

• An algorithm to implement the approach using any fix generation algorithm

• An empirical evaluation that shows the overall reduction of choices exposed to the user
Our Approach

<table>
<thead>
<tr>
<th>Item</th>
<th>Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>PreloadSize</td>
<td>Requires PreloadSize &lt;= PoolSize</td>
</tr>
</tbody>
</table>

[PreloadSize <= 8]

Showing only one fix to the user
Provide feedback for each variable

• Accept the change
• Reject the change
  – *Fix* duration
    • I do not want to change the variable to the suggested value/range for now; perhaps later.
  – *Error* duration
    • I do not want to change this variable during this error resolution
  – *Permanent* duration
    • I do not want to change this variable in the whole configuration process
Our Approach

[PreloadSize <= 8]

Reject with Error
Our Approach

The user feedbacks are stored as priorities so that later fixes will be smarter.
Our Contribution

• A priority-based approach to locating a desirable fix through user feedbacks

• An algorithm to implement the approach using any fix generation algorithm

• An empirical evaluation that shows the overall reduction of choices exposed to the user
Algorithm Overview

Each variable is assigned a priority, initially zero.
Recommend a fix

- Use a threshold to confine the fix generation scope
  - Variables are changeable only when priority <= threshold.
  - Constraint [variable = current_value] is added for variables whose priority > threshold
Recommend a fix

- Initial threshold for an error = 0
- Invoke the fix generator
  - Randomly pick one fix from the generated fix list
  - Threshold += 1 if no fix is generated, and invoke again

<table>
<thead>
<tr>
<th>Threshold</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Priority</td>
<td>0</td>
<td>v1</td>
<td>v2</td>
<td>v3</td>
</tr>
</tbody>
</table>
Adjust Priorities

• New value is assigned
  – priority = 0
• Reject with Fix duration
  – priority +=1
• Reject with Error duration
  – priority binds to <threshold> +1
  – will be updated when threshold increases during the error resolution
• Reject with Permanent duration
  – priority = <max>
Handling No fixes

• Provide users with the variables with *error* and *permanent* durations

• Users should cancel some of the durations
  – The priorities of cancelled variables are reset to 0
Our Contribution

• A priority-based approach to locating a desirable fix through user feedbacks
• An algorithm to implement the approach using any fix generation algorithm
• An empirical evaluation that shows the overall reduction of choices exposed to the user
Supporting Tool: Smart Fixer

(a) SmartFixer: Interactive process GUI for fix resolution
Smart Fixer: providing feedbacks

<table>
<thead>
<tr>
<th>Variable</th>
<th>Value</th>
<th>Duration</th>
</tr>
</thead>
<tbody>
<tr>
<td>CYGSEM_HAL_VIRTUAL_VECTOR_CLAIM_COMMS_bool</td>
<td>true</td>
<td>NA</td>
</tr>
<tr>
<td>CYGSEM_HAL_USE_ROM_MONITOR_bool</td>
<td>true</td>
<td>Fix</td>
</tr>
<tr>
<td>CYGDBG_HAL_COMMONI_CONTEXT_SAVE_MINIMUM_bool</td>
<td>true</td>
<td>Error</td>
</tr>
</tbody>
</table>
Evaluation

• Sources
  – Version history from 2 open source projects that cause large fix lists.

<table>
<thead>
<tr>
<th>Project</th>
<th>Architecture</th>
<th>Variables</th>
<th>Constraints</th>
<th>Errors</th>
</tr>
</thead>
<tbody>
<tr>
<td>ReconOS</td>
<td>virtex4</td>
<td>933</td>
<td>330</td>
<td>56</td>
</tr>
<tr>
<td></td>
<td>xilinx</td>
<td>765</td>
<td>272</td>
<td>48</td>
</tr>
</tbody>
</table>

  – Initial configuration already contains error.
  – The final configuration has all error resolved.
Evaluations

- **Steps:**
  - Resolve each error by simulating the user feedback

<table>
<thead>
<tr>
<th>Situation #</th>
<th>Current Value</th>
<th>Fix Changes</th>
<th>Final Value</th>
<th>Operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>a = 1</td>
<td>a &lt; 1</td>
<td>a = 2</td>
<td>Reject Fix duration</td>
</tr>
<tr>
<td>2</td>
<td>a = 1</td>
<td>a &gt; 1</td>
<td>a = 2</td>
<td>Accept Assign new value</td>
</tr>
<tr>
<td>3</td>
<td>a = 2</td>
<td>a &gt; 2</td>
<td>a = 2</td>
<td>Reject Error duration</td>
</tr>
</tbody>
</table>

- Count the number of fixes and variables
The number of fixes is decreased in 31% of the errors. In average, there is a reduction of 22%, with a maximum reduction of 89% in the number of fixes.
The number of variables is decreased by 23% in average, with a maximum reduction of 98%
Evaluation Results – xilinx (1/2)

The number of fixes is decreased in 28% of the errors. In average, there is a reduction of 16%, with a maximum reduction of $2/3$ in the number of fixes.
Evaluation Results - xilinx (2/2)

The number of variables is decreased by 18% in average, with a maximum reduction of 86%
Summary & Future Work

• Adopt self-adaptive priorities to guide users to the desirable fixes
  – Five assignment and adjustment rules
• Can be built on any fix generators
• Produce a good result in eCos configuration
• Future work: Need user-involved evaluation
Thank you for your attention!
Variability Modeling Languages

- Linux KConfig
- eCos CDL
- Feature models

- Usually come with a configurator
  - Can detect errors
eCos Configurator - Errors

- A error is a violation of a constraint.
Fix Errors

• How to fix errors?
  – Understand the constraints
  – Navigate the variables

• Fixing errors is hard.
  – Constraints are complex
    • Linux (56 variables in one constraint)
  – Real world variability models contain thousands of variables.
Survey

• 97 Linux users and 9 eCos users
• 20% Linux users need "a few dozen minutes" to fixing an error in average
• 56% eCos users consider fixing to be a problem
Existing Approaches
Generate Fixes for Errors
Fixes

- Fixes is a set of changes for the variables

- A change is an assignment for a variable

- Executing the fixes will satisfy all the constraints in the variability model
eCos Configurator - Fixes

The VIRTEX4 HAL package provides the support needed to run eCos on a XILINX PowerPC 4000.
The number of fixes is decreased in 31% of the errors. In average, there is a reduction of 22%, with a maximum reduction of 89% in the number of fixes.
Evaluation Results (2/4)

The number of variables is decreased by 23% in average, with a maximum reduction of 98%.

*Fig. 6: Experimental results for virtex4*
The number of fixes is decreased in 28% of the errors. In average, there is a reduction of 16%, with a maximum reduction of 2/3 in the number of fixes.
Evaluation Results (4/4)

The number of variables is decreased by 18% in average, with a maximum reduction of 86%