

# Interactive fixes for software configuration

Yingfei Xiong, Peking University

Cooperation with:

Peking University

Bo Wang, Hansheng Zhang, Jie Wang,  
Haiyan Zhao, Wei Zhang

University of Waterloo

Leonardo Passos, Steven She, Krzysztof Czarnecki

University of Namur

Arnaud Hubaux

2013

# 北京大学软件工程研究所

- 国内最早开展软件工程专业研究、规模最大、最有影响力的软件工程专业研究团队
- 院士三名（含双聘一名），博士生导师**10**名，硕士生导师**13**名
- 在软件工程顶级会议发表论文数占大陆总数约三分之二\*
- 获得ACM SIGSOFT杰出论文奖三次，大陆共获奖四次，香港共获奖一次

\*统计截止2011年

# Variability Models & Configurators



Configuration

Item	Pre	Value
Object Pool Configuration		v3_0
ab Buffer Size (KB)		4
ab Object Size (Byte)		512
ab Object Pool Size		8
<input checked="" type="checkbox"/> Use Pre-Allocation		
ab Pre-Allocation Size		10
Allocation Time		
<input type="checkbox"/> Startup		
<input checked="" type="checkbox"/> First Access		
<input type="checkbox"/> Idle		



Variability Models

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

# eCos Configurator - Errors

The screenshot shows the eCos Configuration Tool window titled "unnamed3\* - eCos Configuration Tool". The interface includes a menu bar (File, Edit, View, Build, Tools, Help) and a toolbar with various icons. The main area is divided into a configuration tree on the left and a property table on the right.

The configuration tree shows the following structure:

- Configuration
  - Object Pool (v3\_0)
    - Buffer Size (KB): 4
    - Object Size (Byte): 512
    - Pool Size: 8
    - Preload (checked)
      - Preload Size: 10 (highlighted)
      - Allocation\_Time
        - Startup (unchecked)
        - First Access (checked)
        - Idle (unchecked)

Item	Property
PreloadSize	Requires PreloadSize <= PoolSize

Property	Value
Value	10
Default	10
Flavor	data
Requires	PreloadSize <= PoolSize
DefaultValue	10

# eCos Configurator - Inactive Options

The screenshot shows the eCos Configurator tool interface. The left pane displays a tree view of configuration options for an 'Object Pool' (v3\_0). The right pane shows a table of properties and values for the selected item.

Item	Property
Object Pool	PreloadSize
	Requires PreloadSize <= PoolSize

Property	Value
Macro	Startup
Enabled	False
Flavor	bool
Implements	Allocation_Time
ActiveIf	PreloadSize <= PoolSize / 2

The configuration options in the left pane are:

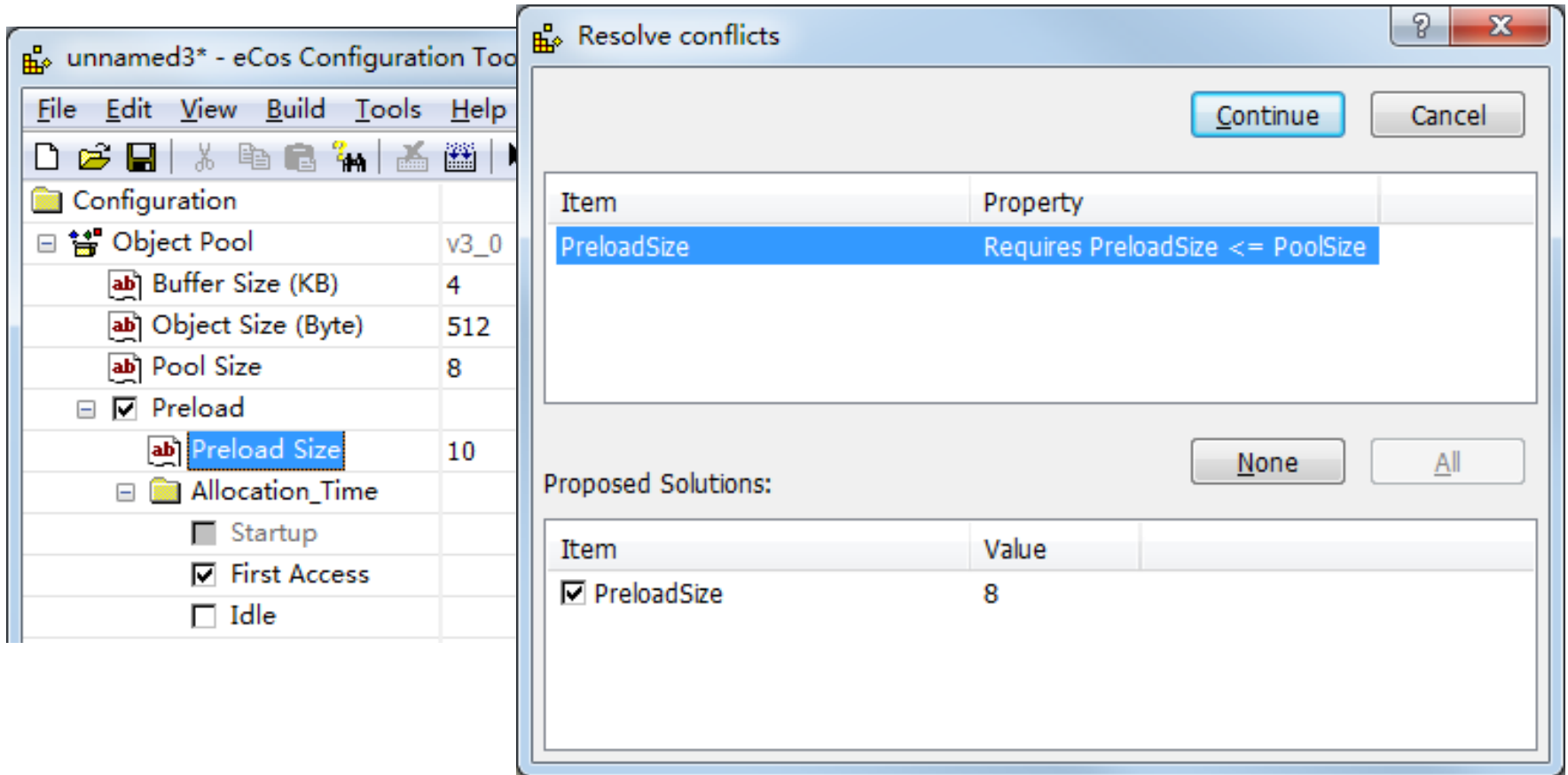
- Object Pool (v3\_0)
  - Buffer Size (KB): 4
  - Object Size (Byte): 512
  - Pool Size: 8
  - Preload (checked)
    - Preload Size: 10
    - Allocation\_Time
      - Startup (disabled, highlighted by a red callout bubble labeled 'disabled')
      - First Access (checked)
      - Idle (unchecked)

Error resolution and option activation both need to resolve violation of constraint.

# Survey

- 97 Linux users and 9 eCos users
- Resolving a violation is hard
  - 20% Linux users need "a few dozen minutes" to activate an option in average
  - 56% eCos users consider activation to be a problem

# eCos Configurator



Essentially, fixes work for both resolving errors and activating options

# Fix Incompleteness

The image shows two windows from the eCos Configuration Tool. The left window, titled 'unnamed3\* - eCos Configuration Tool', displays a configuration tree. The 'Object Pool' section is expanded, showing 'Preload Size' set to 10. A red circle highlights the 'Preload Size' value, and a red box labeled 'Disable' has an arrow pointing to the 'Preload' checkbox. The right window, titled 'Resolve conflicts', shows a conflict for 'PreloadSize' with the property 'Requires PreloadSize <= PoolSize'. A red box above the conflict table says 'Increase to any value >= 10'. The 'Proposed Solutions' table shows 'PreloadSize' with a value of 8, which is circled in red. A red box below this table says 'Further decrease to any value <= 8'.

Item	Property
PreloadSize	Requires PreloadSize <= PoolSize

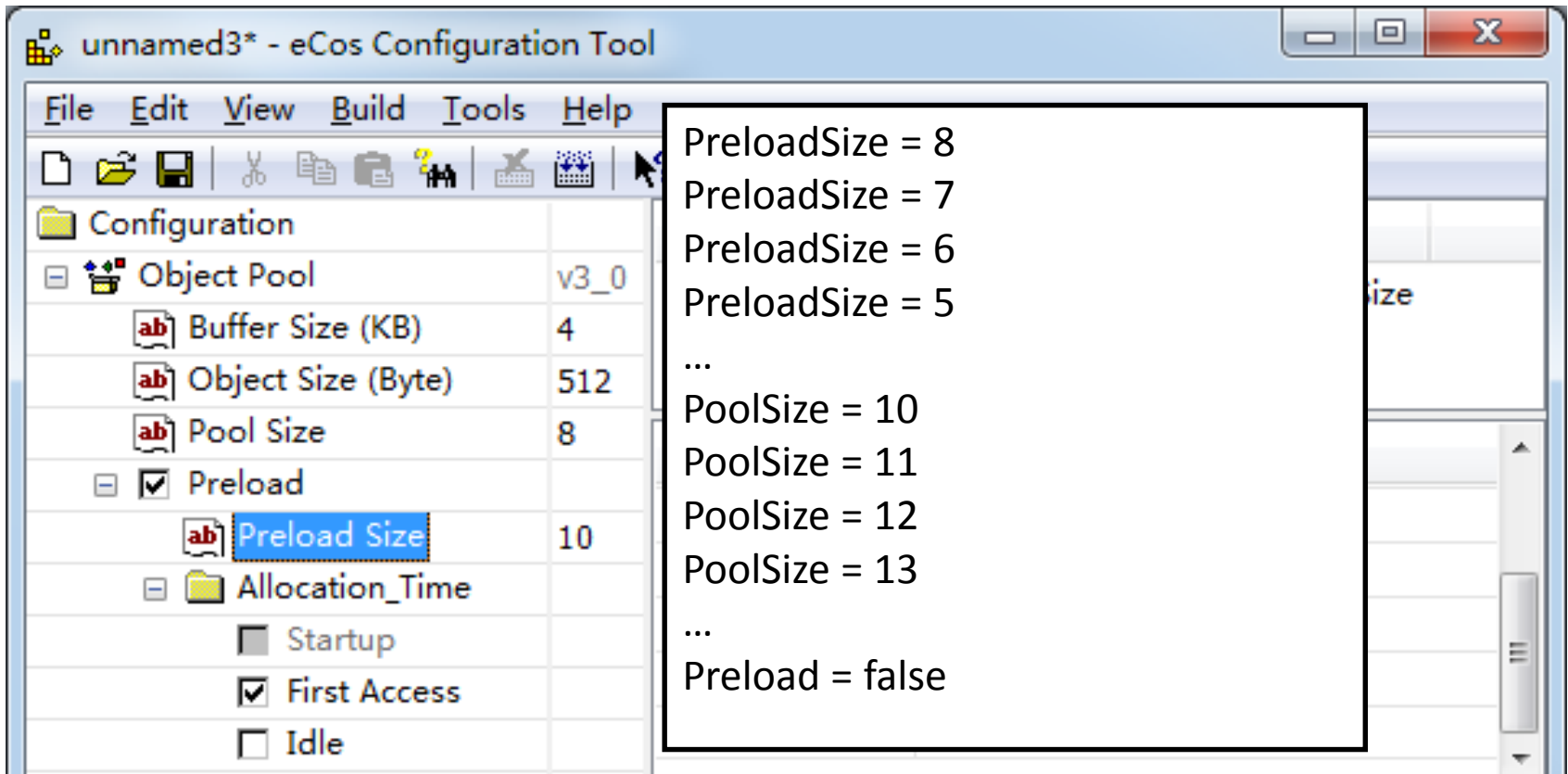
  

Item	Value
<input checked="" type="checkbox"/> PreloadSize	8

78% eCos users have encountered situations where the proposed fix is not useful



# How to complete fixes



The screenshot shows the 'eCos Configuration Tool' window. The configuration tree on the left is as follows:

Configuration Item	Value
Configuration	
Object Pool	v3_0
Buffer Size (KB)	4
Object Size (Byte)	512
Pool Size	8
Preload	<input checked="" type="checkbox"/>
Preload Size	10
Allocation_Time	
Startup	<input type="checkbox"/>
First Access	<input checked="" type="checkbox"/>
Idle	<input type="checkbox"/>

A list of fixes is shown in a text box on the right:

- PreloadSize = 8
- PreloadSize = 7
- PreloadSize = 6
- PreloadSize = 5
- ...
- PoolSize = 10
- PoolSize = 11
- PoolSize = 12
- PoolSize = 13
- ...
- Preload = false

# Our Solution – Range Fixes

The screenshot shows the 'eCos Configuration Tool' window for 'unnamed3\*'. The 'Object Pool' configuration is expanded, showing various properties. A callout box highlights three specific fixes:

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

Item	Property
PreloadSize	Requires PreloadSize <= PoolSize
Flavor	data
Requires	PreloadSize <= PoolSize
DefaultValue	10

Configuration details from the screenshot:

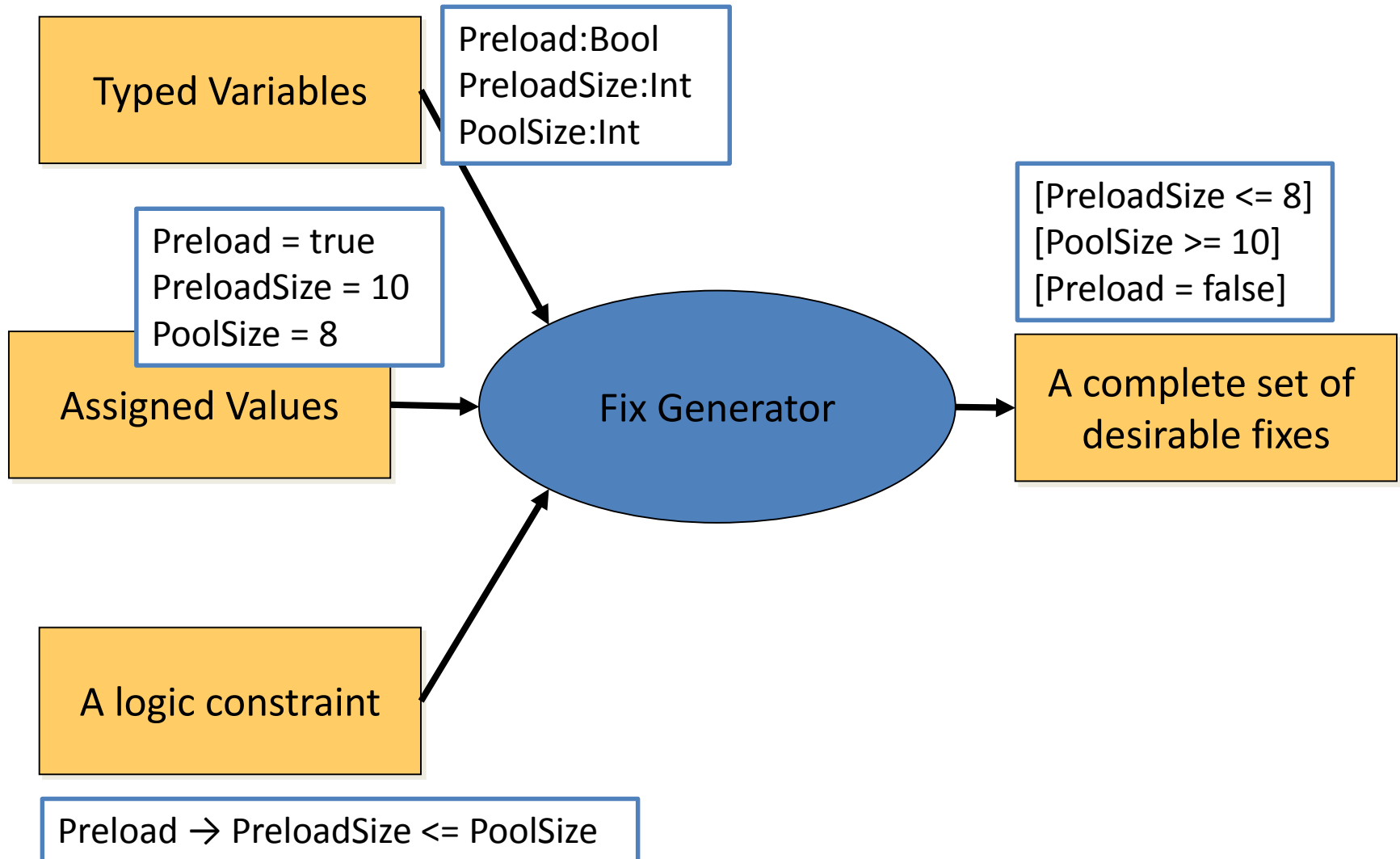
Item	Property
Object Pool	v3_0
Buffer Size (KB)	4
Object Size (Byte)	512
Pool Size	8
Preload	<input checked="" type="checkbox"/>
Preload Size	10
Allocation_Time	
Startup	<input type="checkbox"/>
First Access	<input checked="" type="checkbox"/>
Idle	<input type="checkbox"/>

# Our Contributions

- Defining the range fix generation problem
  - Three desirable properties of range fixes
- Proposing a range fix generation algorithm
- Exploring the constraint interaction problem
  - Summarizing and adapting three strategies used in existing work
  - Comparing the strategies empirically

# Fix Generation Problem

## – a General Definition



# Desired Properties of Fixes

Correctness	Minimality of variables	Maximality of ranges
Any change represented by a range fix will satisfy the constraint	There is no way to change a subset of variables to satisfy the constraint	A range fix represents the maximal ranges over the variables
<b>A desirable one:</b> [PreloadSize <=8]		
<b>Undesirable ones</b>		
[PreloadSize <= 9]	[PreloadSize <=8, Preload = false]	[PreloadSize <=7]

# Algorithm Outline

- Step 1: find the variables to change
  - Basic idea: translating to an SMT problem
    - ① treat configurations also as soft constraints
      1. [soft] Preload = true
      2. [soft] PreloadSize = 10
      3. [soft] PoolSize = 8
      4. [hard] Preload  $\rightarrow$  PreloadSize  $\leq$  PoolSize
    - ② ask an SMT solver for unsatisfiable cores
      - (1, 2, 3)
    - ③ pick one variable from each core
      - {Preload}, {PreloadSize}, {PoolSize}

# Algorithm Outline

- Step 2: find the range of the variables
  - Basic idea: simplify the constraint
    - Example: {PreloadSize}
    - ① replace unchangeable variables with their current values
      - true  $\rightarrow$  PreloadSize  $\leq$  8
    - ② simplify the constraint and convert to CNF
      - [PreloadSize  $\leq$  8]

# Constraint Interaction

The screenshot shows the 'eCos Configuration Tool' window. The main configuration table is as follows:

Item	Property
Object Pool	v3_0
Buffer Size (KB)	4
Object Size (Byte)	512
Pool Size	8
Preload	<input checked="" type="checkbox"/>
Preload Size	10
Allocation_Time	
Startup	<input type="checkbox"/>
First Access	<input checked="" type="checkbox"/>
Idle	<input type="checkbox"/>

A callout box highlights the following constraints:

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

The 'Preload Size' property is currently set to 10, which is greater than the 8 specified in the constraint. The 'Preload' property is checked, which is the opposite of the 'Preload = false' constraint.



# Constraint Interaction

The screenshot shows the eCos Configuration Tool interface. The left pane displays the configuration tree with 'Pool Size' set to 12. A red circle highlights the value 12, with a callout box below it saying 'Increase PoolSize'. The right pane shows a table of properties for the selected item, with a callout box above it saying 'Causing another error'. The table shows a 'Requires' constraint:  $PoolSize == BufferSize * 1024 / ObjectSize$ . Below this, a table shows the current values: File, Macro, Value (12), Default (0), Flavor (data), and Requires (PoolSize == BufferSize \* 1024 / ObjectSize). A callout box below the right pane says 'Interacting constraint'.

Item	Property
PoolSize	Requires PoolSize == BufferSize * 1024 / ObjectSize

Property	Value
File	unnamed3_install/include/pkgconf/hal.h
Macro	PoolSize
Value	12
Default	0
Flavor	data
Requires	PoolSize == BufferSize * 1024 / ObjectSize

Causing another error

Increase PoolSize

Interacting constraint

# Ignorance

Ignore the interaction

The screenshot shows the 'eCos Configuration Tool' window for 'unnamed3\*'. The configuration table is as follows:

Item	Property
Object Pool	v3_0
PreloadSize	Requires PreloadSize <= PoolSize
Buffer Size (KB)	4
Object Size (Byte)	512
Pool Size	8
Preload	<input checked="" type="checkbox"/>
Preload Size	10
Allocation_Time	
Startup	<input type="checkbox"/>
First Access	<input checked="" type="checkbox"/>
Idle	<input type="checkbox"/>
DefaultValue	10

A text box highlights the following configuration values:

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

# Elimination

Eliminate all changes that will violate other constraints

The screenshot shows the 'eCos Configuration Tool' window for 'unnamed3\*'. The left pane shows a tree view of configuration options under 'Configuration'. The right pane shows a table of properties for the selected 'Preload Size' item.

Item	Property
PreloadSize	Requires PreloadSize <= PoolSize

Configuration options (Left Pane):

- Object Pool (v3\_0)
  - Buffer Size (KB): 4
  - Object Size (Byte): 512
  - Pool Size: 8
  - Preload (checked)
    - Preload Size: 10
    - Allocation\_Time
      - Startup:
      - First Access:
      - Idle:

# Propagation

Propagate the change along other constraints

The screenshot shows the 'eCos Configuration Tool' window with the following configuration parameters:

Item	Property
Object Pool	v3_0
PreloadSize	Requires PreloadSize <= PoolSize
Buffer Size (KB)	4
Object Size (Byte)	512
Pool Size	8

Additional configuration options visible in the tool include:

- Preload
- Allocated
- Start
- First
- Idle

The highlighted constraints are:

- [PreloadSize <= 8]
- [PoolSize >= 10 & BufferSize = PoolSize / 2]
- [PoolSize >= 10 & ObjectSize = 4096 / PoolSize]
- [Preload = false]

# Translating to the basic case

- Assignments: Preload = true, PreloadSize = 10, PoolSize = 8, BufferSize = 4, ObjectSize = 512
- Constraints:
  - **Preload  $\rightarrow$  PreloadSize  $\leq$  PoolSize**
  - PoolSize == BufferSize \* 1024 / ObjectSize
- Ignorance:
  - Preload  $\rightarrow$  PreloadSize  $\leq$  PoolSize
- Elimination:
  - Preload  $\rightarrow$  PreloadSize  $\leq$  PoolSize  $\wedge$  PoolSize == 4 \* 1024 / 512
- Propagation:
  - Preload  $\rightarrow$  PreloadSize  $\leq$  PoolSize  $\wedge$  PoolSize == BufferSize \* 1024 / ObjectSize

# Comparison of Strategies

	Ignorance	Elimination	Propagation
Execution time	Shortest	Short	Possibly long
Complexity of fix lists	Simple	Simplest	Possibly complex
Introduction of new errors	Possible	Never	Never
Fix completeness	Complete (for one constraint)	Incomplete	Complete (for all constraints)

# Experiments

- Source
  - Version histories from 5 open source projects
- Steps
  - Compare each pair of consecutive versions
  - Replay the user changes in different orders
  - Generate fixes for the violations and compare with user changes

# Execution Time

	Ignorance	Elimination	Propagtion
Execution time	Average: 17ms Maximum: 20ms	Average: 20ms Maximum: 30ms	Average: 50ms Maximum: 250ms
Complexity of fix lists	Simple	Simplest	Possibly complex
Introduction of new errors	Possible	Never	Never
Fix completeness	Complete (for one constraint)	Incomplete	Complete (for all constraints)

Our algorithm is sufficiently fast for each strategy



# Complexity of fix lists

	Ignorance	Elimination	Propagtion
Execution time	Average: 17ms Maximum: 20ms	Average: 20ms Maximum: 30ms	Average: 50ms Maximum: 250ms
Complexity of fix lists (Number of variables in a list)	Max: 4 Median: 2 Average: 2.2	Max: 4 Median: 2 Average: 1.64	Max: 58 Median: 2 Average: 8.0
Introduction of new errors	Possible	Never	Never
Fix completeness	Complete (for one constraint)	Incomplete	Complete (for all constraints)

In propagation, 83% of the fix lists contain less than 10 variables

# Introduction of new errors

	Ignorance	Elimination	Propagtion
Execution time	Average: 17ms Maximum: 20ms	Average: 20ms Maximum: 30ms	Average: 50ms Maximum: 250ms
Complexity of fix lists (Number of variables in a list)	Max: 4 Median: 2 Average: 2.2	Max: 4 Median: 2 Average: 1.64	Max: 58 Median: 2 Average: 8.0
Introduction of new errors	44% of all violations	Never	Never
Fix completeness	Complete (for one constraint)	Incomplete	Complete (for all constraints)

# Fix completeness

	Ignorance	Elimination	Propagtion
Execution time	Average: 17ms Maximum: 20ms	Average: 20ms Maximum: 30ms	Average: 50ms Maximum: 250ms
Complexity of fix lists (Number of variables in a list)	Max: 4 Median: 2 Average: 2.2	Max: 4 Median: 2 Average: 1.64	Max: 58 Median: 2 Average: 8.0
Introduction of new errors	44% of all violations	Never	Never
Fix completeness (coverage of user changes)	100%	57%	100%

eCos configurator: 73%

# Problem: Large Fixes

	Ignorance	Elimination	Propagation
Execution time	Average: 17ms Maximum: 20ms	Average: 20ms Maximum: 30ms	Average: 50ms Maximum: 250ms
Complexity of fix lists (Number of variables in a list)	Max: 4 Median: 2 Average: 2.2	Max: 4 Median: 2 Average: 1.64	Max: 58 Median: 2 Average: 8.0
Introduction of new errors	Possible	Never	Never
Fix completeness	Complete (for one constraint)	Incomplete	Complete (for all constraints)

In propagation, 83% of the fix lists contain less than 10 variables

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

The screenshot shows the 'eCos Configuration Tool' window. The left pane displays a tree view of configuration items. The right pane shows a table of properties for the selected 'Preload Size' item.

Item	Property
PreloadSize	Requires PreloadSize <= PoolSize

[PreloadSize <= 8]

Property Value

Property	Value
Value	10
Default	10
Flavor	data

Showing only one fix to the user

# Provide feedback for each variable

- Accept the change (and pick a value)
- Reject the change
  - *Fix* duration
    - Current range is incorrect, future fixes can propose changes for this variable
  - *Error* duration
    - Current value is correct when fixing this error
  - *Permanent* duration
    - Current value is correct in the whole configuration process

# Our Approach

The screenshot shows the eCos Configuration Tool interface. The left pane displays a tree view of configuration items. The right pane shows a table of properties for the selected item, 'Preload Size'. A callout box points to the 'Preload Size' property in the right pane, indicating a constraint: `[PreloadSize <= 8]`. A blue callout box at the bottom states: `Reject with Fix Duration`.

Item	Property
PreloadSize	Requires PreloadSize <= PoolSize

Property	Value
Value	10
Default	10
Flavor	data

Reject with Fix Duration

# Our Approach

The screenshot shows the eCos Configuration Tool interface. The configuration tree on the left includes:

- Configuration
  - Object Pool (v3\_0)
    - Buffer Size (KB): 4
    - Object Size (Byte): 512
    - Pool Size: 8
  - Preload
    - Preload
  - Allocation\_T...

The right pane shows a table with the following data:

Item	Property
PreloadSize	Requires PreloadSize <= PoolSize

A callout box contains the expression:  $[PoolSize \geq 10 \ \& \ BufferSize = PoolSize / 2]$

Two callout boxes provide the logic for the expression:

- Accept with PoolSize = 16
- Reject with Error Duration

# Our Approach

The screenshot shows the 'eCos Configuration Tool' window. The left pane shows a tree view with 'Object Pool' expanded, showing 'Buffer Size (KB)' (4), 'Object Size (Byte)' (512), 'Pool Size' (8), and 'Preload' (checked, 10). The right pane shows a table of properties and values.

Item	Property
PreloadSize	Requires PreloadSize <= PoolSize
Property	Value
Value	10
Default	10
Flavor	data
	PreloadSize <= PoolSize

A callout box with a black border contains the text 'ObjectSize = 256'. A blue callout box with a white border contains the text 'Accept'.

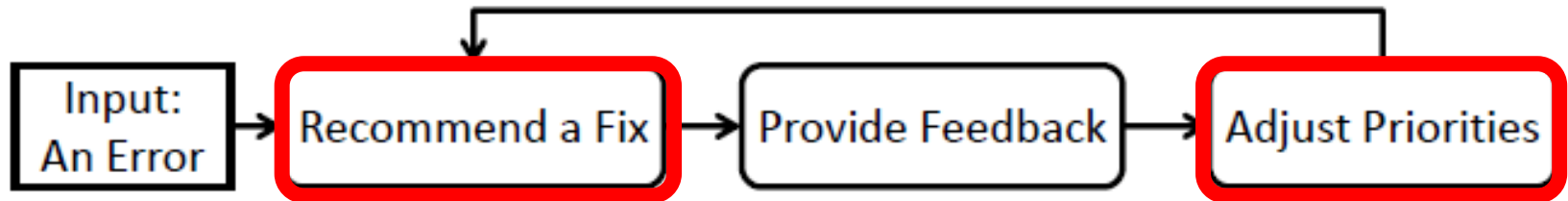
The user feedbacks are stored so that later fixes will be smarter.

# 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

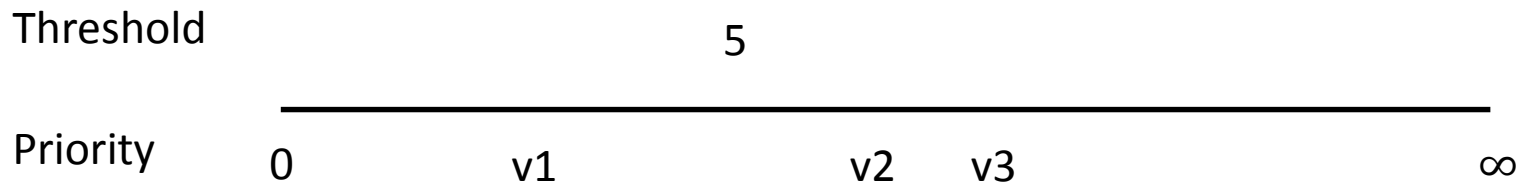
# Algorithm Overview

Each variable is assigned a priority, initially zero.



# Recommend a fix

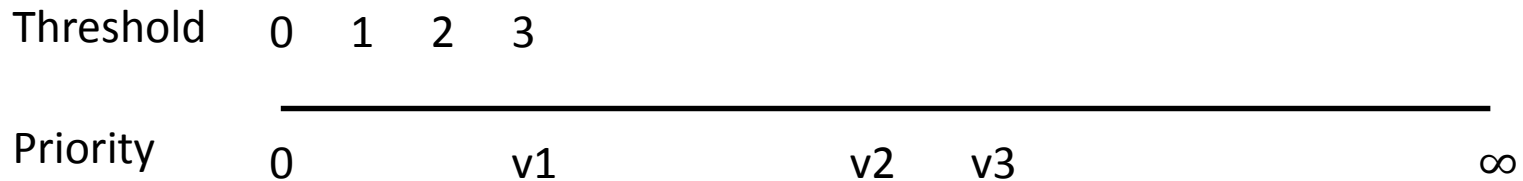
- Use a threshold to confine the fix generation scope
  - Variables are changeable only when priority  $\leq$  threshold.
  - Constraint [variable = current\_value] is added for variables whose priority  $>$  threshold





# Recommend a fix

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



# 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
- *Reject with Permanent* duration
  - priority = <max>

# Handling No fixes

- Provide users with the variables with *error* and *permanent* durations
- Users should change the durations

# 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

The image shows the SmartFixer GUI with three numbered callouts:

- 1**: Configuration panel showing build options like "Global build options", "Global command prefix", "Global compiler flags", "Global linker flags", "Build GDB stub ROM image", "Build common GDB stub ROM in", "Redboot HAL options", and "Mouse driver for virtex4".
- 2**: A list of fixes. The first fix is expanded, showing the following properties:

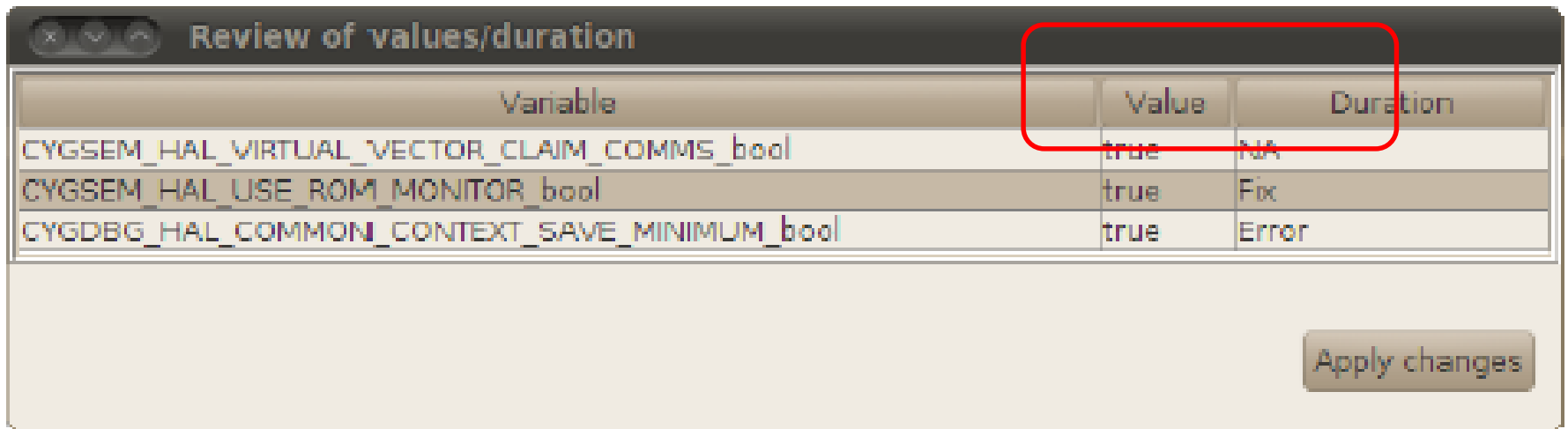
```
CYDBG_HAL_DEBUG_GDB_INCLUDE_STUBS_bool:=true
CYDBG_HAL_COMMON_INTERRUPTS_SAVE_MINIMUM_CONTEXT_bool:=false
CYDBG_HAL_COMMON_CONTEXT_SAVE_MINIMUM_bool:=false
CYGSEM_HAL_VIRTUAL_VECTOR_CLAIM_COMMS_bool:=true
CYGSEM_HAL_USE_ROM_MONITOR_bool:=false
```
- 3**: A "Fix" dialog box with a table of fix units and their assignments. The table is as follows:

Select	Fix Unit	Assignment (var = value, ...)	Duration
<input checked="" type="checkbox"/>	CYDBG_HAL_DEBUG_GDB_INCLUDE_STUBS_bool:=true		NA
<input checked="" type="checkbox"/>	CYDBG_HAL_COMMON_INTERRUPTS_SAVE_MINIMUM_CONTEXT_bool:=false		NA
<input type="checkbox"/>	CYDBG_HAL_COMMON_CONTEXT_SAVE_MINIMUM_bool:=false		Error
<input type="checkbox"/>	CYGSEM_HAL_VIRTUAL_VECTOR_CLAIM_COMMS_bool:=true		Error
<input type="checkbox"/>	CYGSEM_HAL_USE_ROM_MONITOR_bool:=false		Error

Buttons at the bottom of the dialog include "Select all", "Reset", and "Apply changes".

(a) SmartFixer: Interactive process GUI for fix resolution

# Smart Fixer: providing feedbacks



Review of values/duration

Variable	Value	Duration
CYGSEM_HAL_VIRTUAL_VECTOR_CLAIM_COMMS_bool	true	NA
CYGSEM_HAL_USE_ROM_MONITOR_bool	true	Fix
CYGDBG_HAL_COMMON_CONTEXT_SAVE_MINIMUM_bool	true	Error

Apply changes

# Evaluation

- Sources
  - Version history from 2 open source projects that cause large fix lists
  - Simulate the user change from the default configurations to the final configurations

Project	Architecture	Variables	Constraints	Errors (initial conf.)
ReconOS	virtex4	933	330	56
	xilinx	765	272	48

# Evaluations

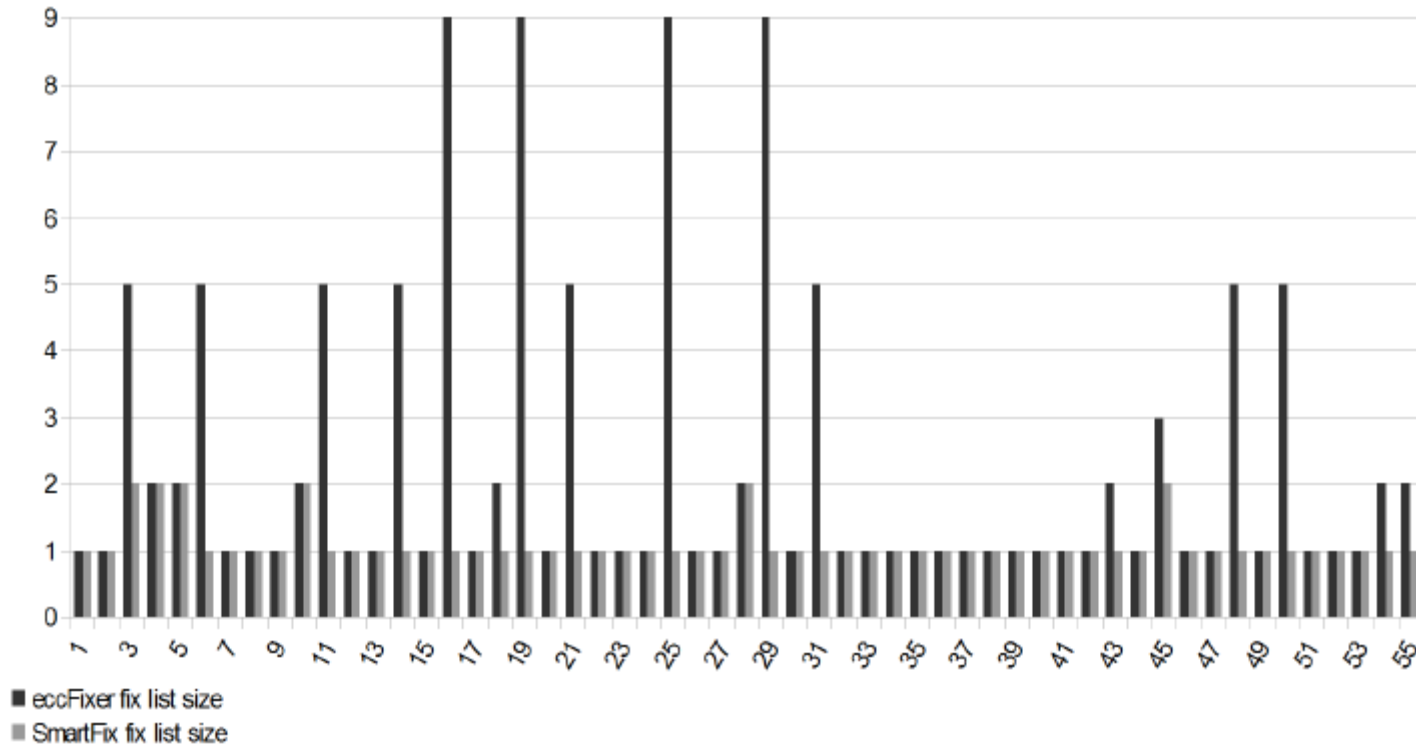
- Steps:
  - Generate a fix for each error, simulate the user feedback

Situation #	Current Value	Fix Changes	Final Value	Operation
1	$a = 1$	$a < 1$	$a = 2$	Reject Fix duration
2	$a = 1$	$a > 1$	$a = 2$	Accept Assign new value
2s	$a = 2$	$a > 2$	$a = 2$	Reject Error duration

- Count the number of fixes and variables



# Evaluation Results – virtex4 (1/2)

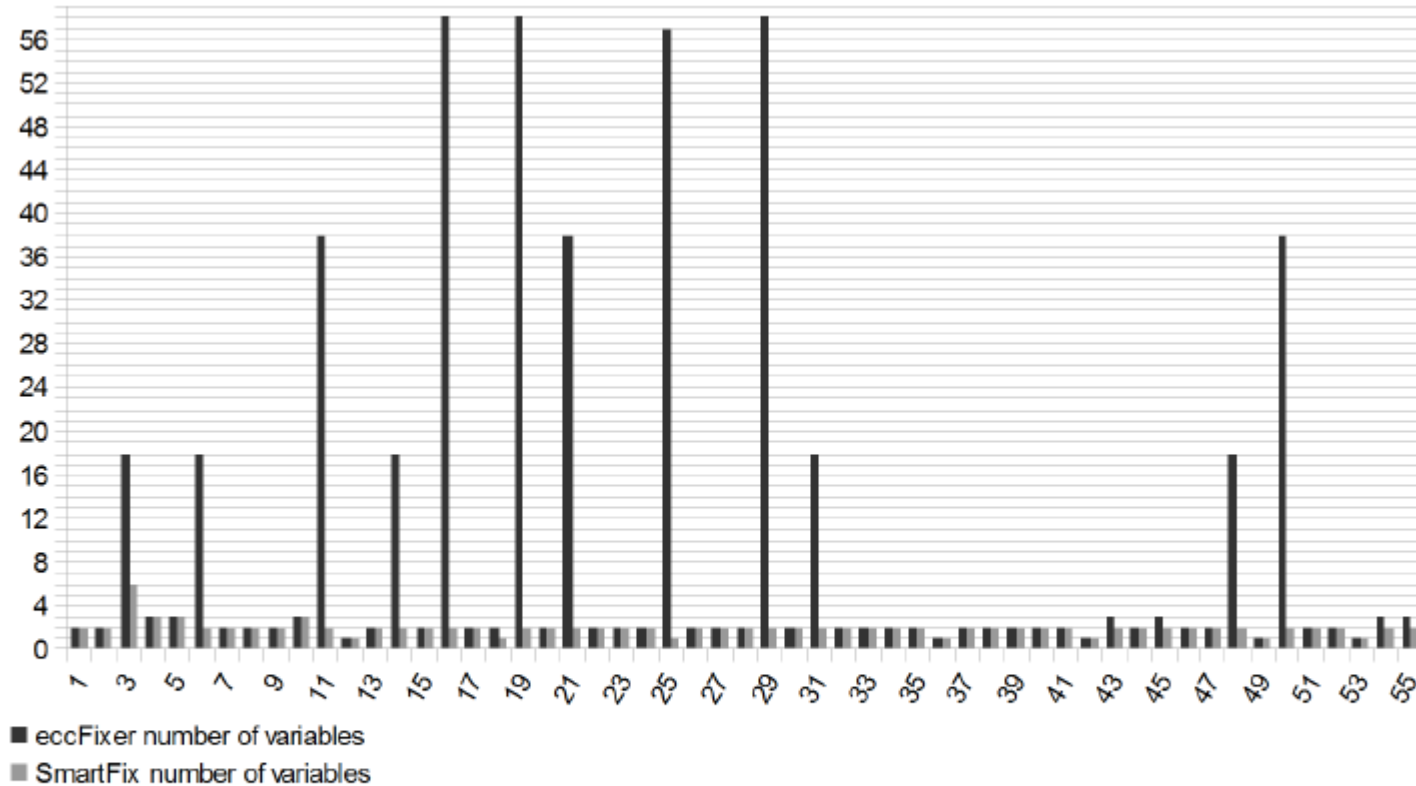


(a) Fix list size

**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 – virtex4 (2/2)

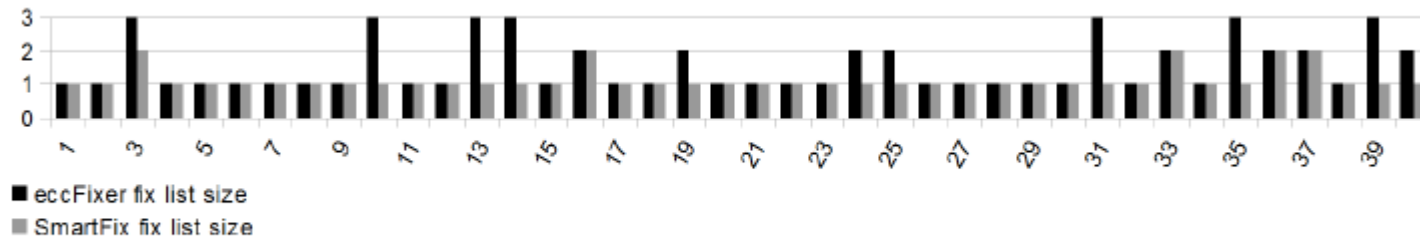


(b) Number of variables

Fig. 6: Experimental results for virtex4

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

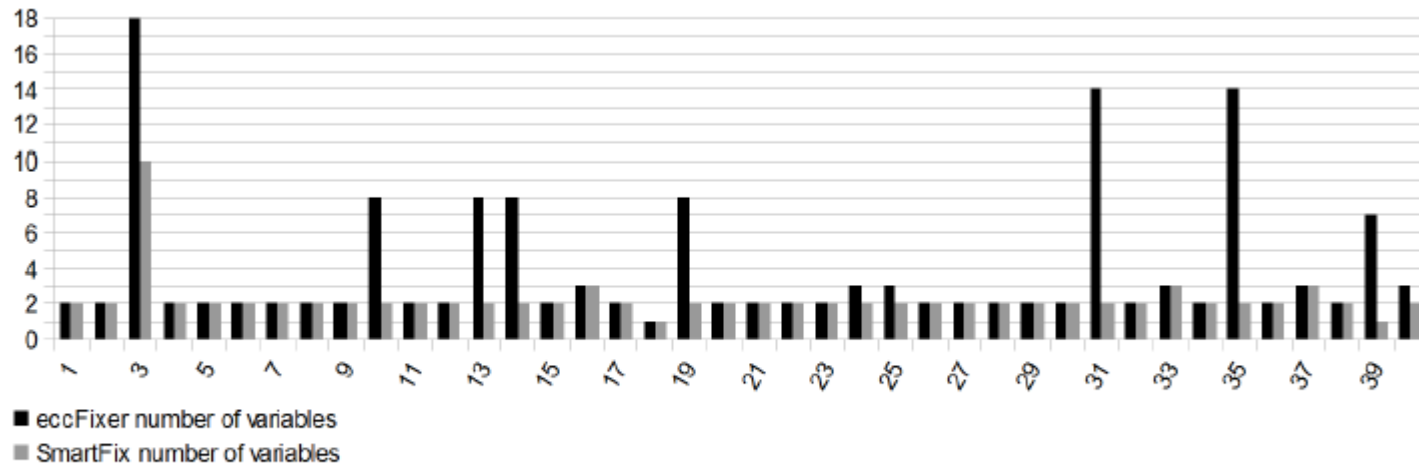
# Evaluation Results – xilinx (1/2)



(a) Fix list size

**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)



(b) Number of variables

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

# Summary

- Error Resolution is difficult in configuring large systems
- Range fixes can be generated efficiently
- Large fix list could be controlled by priorities

# Thank you for your attention!

- **References:**
- Yingfei Xiong, Arnaud Hubaux, Steven She, Krzysztof Czarnecki. Generating Range Fixes for Software Configuration. In ICSE'12: Proceedings of 34th International Conference on Software Engineering, pages 89-99, June 2012.
- Bo Wang, Leonardo Passos, Yingfei Xiong, Krzysztof Czarnecki, Haiyan Zhao, Wei Zhang. SmartFixer: Fixing Software Configurations based on Self-adaptive Priorities. In SPLC'13: Proceedings of 17th International Software Product Line Conference, August 2013.
- Arnaud Hubaux, Yingfei Xiong, Krzysztof Czarnecki. A User Survey of Configuration Challenges in Linux and eCos. VaMoS'12: Sixth International Workshop on Variability Modelling of Software-intensive Systems, January 2012.
- Leonardo Passos, Marko Novakovic, Yingfei Xiong, Thorsten Berger, Krzysztof Czarnecki, Andrzej Wasowski. A Study of Non-Boolean Constraints in Variability Models of an Embedded Operating System. FOSD'11: 3rd International Workshop on Feature-Oriented Software Development, June 2011.