TCM: Test Case Mutation to Improve Crash Detection in Android

Presenter: Yavuz Koroglu

Yavuz Koroglu and Alper Sen

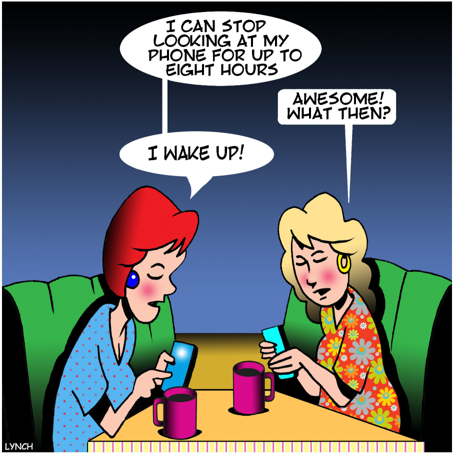
Dependable Systems Group (DSG)
Bogazici University, Istanbul, Turkey
http://depend.cmpe.boun.edu.tr
{yavuz.koroglu,alper.sen}@boun.edu.tr

Fundamental Approaches to Software Engineering (FASE'18)
@ European Joint Conferences on Theory and Practice of Software (ETAPS'18)

Overview

1. Introduction
2. Test Case Mutation (TCM)
3. Case Studies
4. TCM: A Working Example
5. Experiments
6. Conclusion

Motivation



CartoonStock.com

Mobile GUI Applications are Ubiquitous

- We use mobile phones often **(3 hours/day)**
- Mostly on mobile applications **(90% of the time spent)**

Android Market is Growing

- **2.6 billion** mobile phone users

Android has the Largest Share

- **82.8%** of all apps are for Android

Publicly Available Automated Android GUI Testing Tools

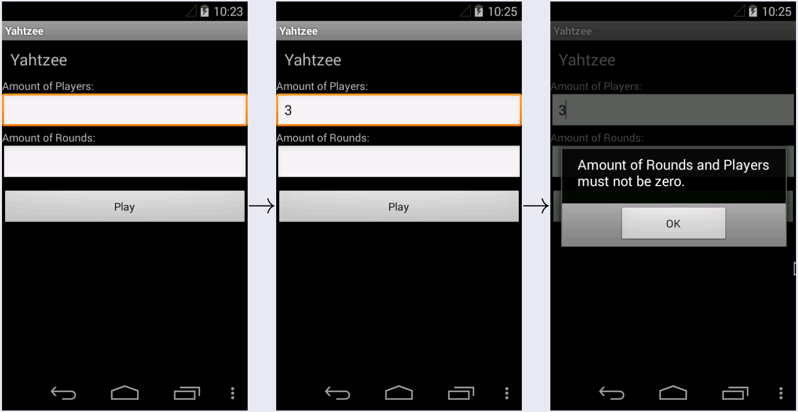
- **Monkey:** Random Explorer
- **A^{3}E:** Depth-First Explorer
- **SwiftHand:** Model Based Depth-First Explorer
- **DynoDroid:** Biased-Random Explorer
- **Sapienz:** Search-Based Explorer
- **QLearning-Based Exploration (QBE)**: Machine Learning Based Explorer on top of AndroFrame. (Published @ ICST'18)

All Above Tools Perform Positive Testing

No focus on **negative testing**.

An Example Crash Found by Only TCM

An Automatically Generated Test Case



Yahtzee

Amount of Players:

Amount of Rounds:

Play

→

Yahtzee

Amount of Players: 3

Amount of Rounds:

Play

→

Yahtzee

Amount of Players: 3

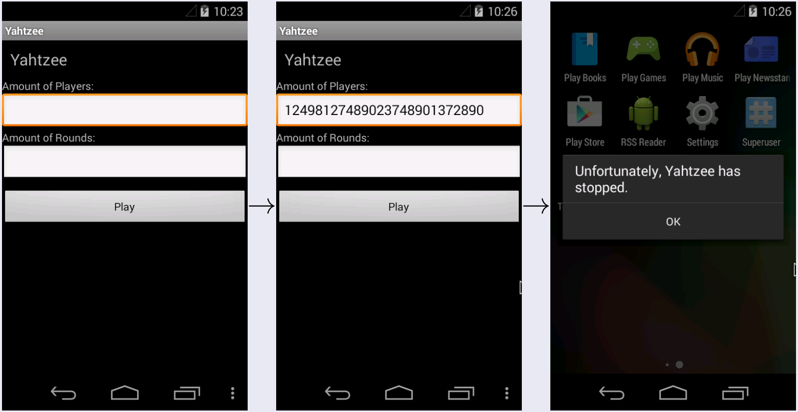
Amount of Rounds:

Amount of Rounds and Players must not be zero.

OK

An Example Crash Found by Only TCM

Mutated Test Case



Yahtzee

Yahtzee

Amount of Players:

Amount of Rounds:

Yahtzee

Yahtzee

Amount of Players: 12498127489023748901372890

Amount of Rounds:

Unfortunately, Yahtzee has stopped.

Test Case Mutation (TCM) Overview

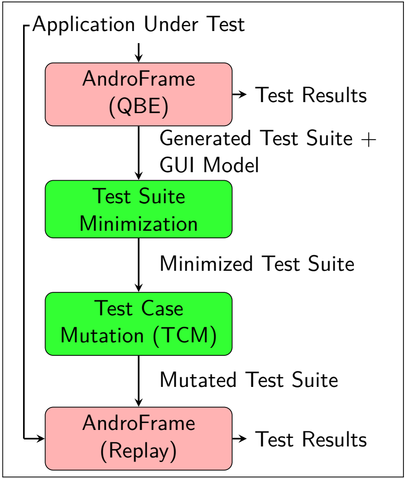


Figure: Test Case Mutation (TCM) Overview

Main Idea

- To **mutate** existing test cases to **enrich** the test suite.

Main Flow

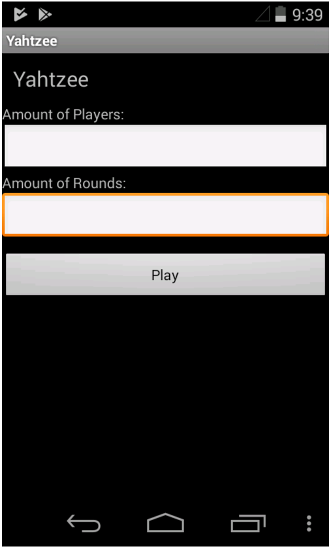
1. **Generate** the Test Suite and GUI Model
2. **Minimize** the Test Suite w.r.t. GUI Model
3. **Mutate** the Test Suite
4. **Replay** the Test Suite

GUI Model (M)

In general,

- Most applications do **NOT** have a model.
- Generate the application model **dynamically**.
- The model is an **Extended Labeled Transition System (ELTS)** where
  1. **Nodes** are GUI **states**.
  2. **Edges** are transitions via GUI **actions**.

GUI State (v)



9:39

Yahtzee

Yahtzee

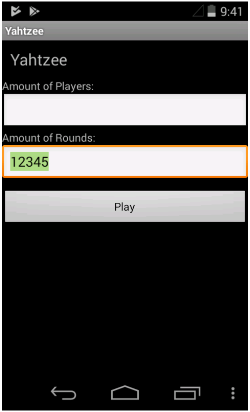
Amount of Players:

Amount of Rounds:

1. **Java Package Name**
2. **Activity Name**
   (An activity roughly corresponds to an Android screen)
3. **Contextual Attributes**
   (WiFi, Orientation etc.)
4. **GUI Components (widgets)**
   on the screen

GUI Action (z)

User-triggered events: **text, click, swipe** etc.



9:41

Yahtzee

Yahtzee

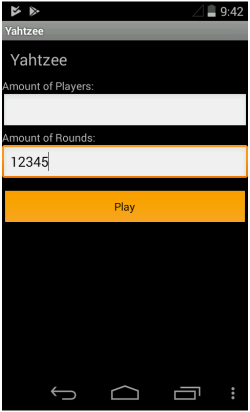
Amount of Players:

Amount of Rounds:

12345

Play

text



9:42

Yahtzee

Yahtzee

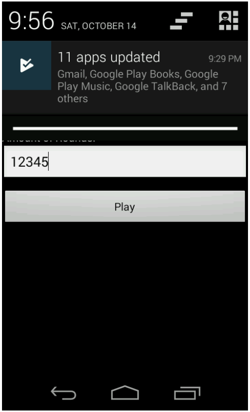
Amount of Players:

Amount of Rounds:

12345

Play

click



9:56 SAT, OCTOBER 14

11 apps updated 9:29 PM

Gmail, Google Play Books, Google Play Music, Google TalkBack, and 7 others

12345

Play

swipe

AndroFrame: Automated Test Generation Framework

What is AndroFrame (QBE)?

It is a

- **Fully-automated**,
- **Black-box**,
- **Modular**,
- **Automata Learning**, and
- **Machine Learning** guided

replayable test case generation framework.

Important

- We build TCM on top of AndroFrame (QBE).

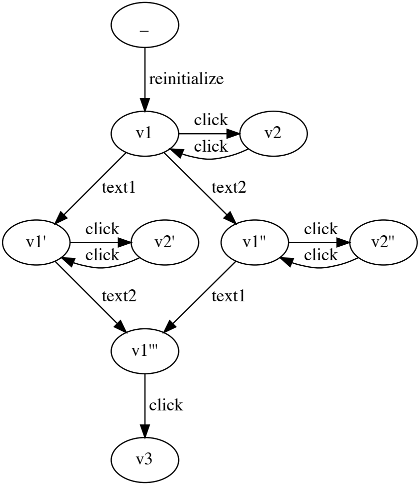


Figure: Example Model of the Yahtzee App

Definitions for Test Case Mutation

Mutation Operator (δ)

A function which

- **takes** a test case t and
- **returns** a new test case t'.

$$\delta(t) = t'$$

Definitions for Test Case Mutation

Mutation Operator (δ)

A function which

- **takes** a test case t and
- **returns** a new test case t'.

$$\delta(t) = t'$$

Test Case (t)

A **sequence of transitions** which

- Start from **the initial state** (v_0) of the GUI Model.

Definitions for Test Case Mutation

Mutation Operator (δ)

A function which

- **takes** a test case t and
- **returns** a new test case t'.

$$\delta(t) = t'$$

Test Case (t)

A **sequence of transitions** which

- Start from **the initial state** (v_0) of the GUI Model.

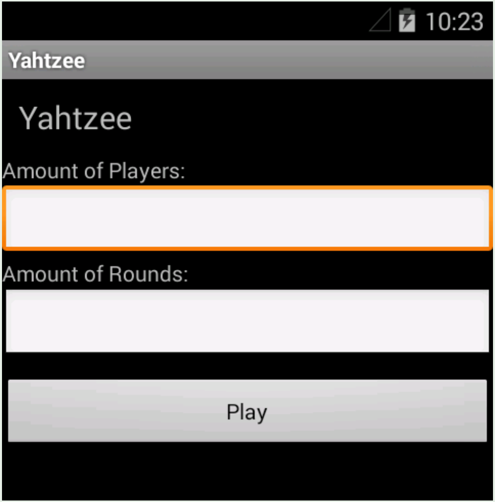
Transition (v_i, v_{i+1}, z_i, d_i)

A **4-tuple**:

1. v_i: Prev State
2. v_{i+1}: Next State
3. z_i: Action
4. d_i: Delay in seconds

Definitions for Test Case Mutation

Example


10:23

Yahtzee

Yahtzee

Amount of Players:

Amount of Rounds:

Transition (v_i, v_{i+1}, z_i, d_i)

A **4-tuple**:

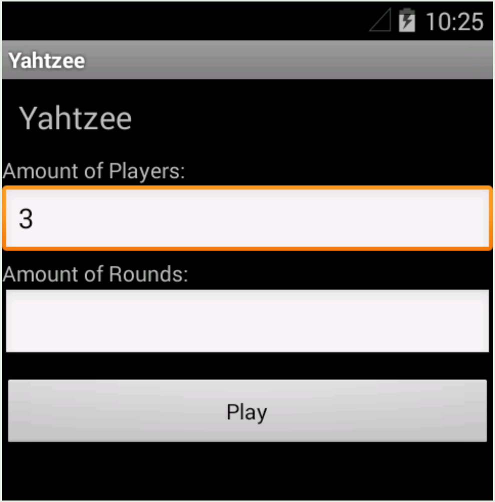
1. v_i: Prev State
2. v_{i+1}: Next State
3. z_i: Action
4. d_i: Delay in seconds

Example

1. $(\_, v1, reinitialize, 15)$

Definitions for Test Case Mutation

Example



Yahtzee

Amount of Players: 3

Amount of Rounds:

Transition (v_i, v_{i+1}, z_i, d_i)

A **4-tuple**:

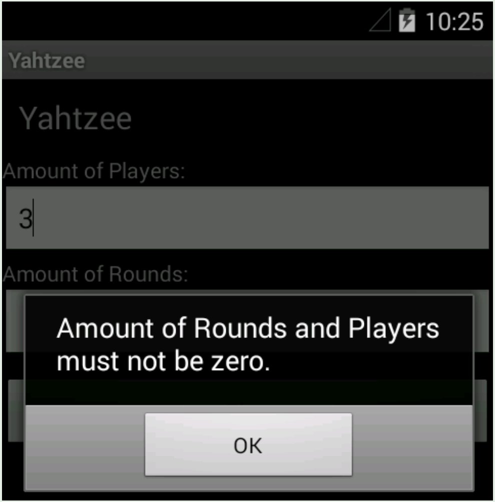
1. v_i: Prev State
2. v_{i+1}: Next State
3. z_i: Action
4. d_i: Delay in seconds

Example

1. $(\_, v1, \mathit{reinitialize}, 15)$
2. $(v1, v1, \mathit{text}1, 1)$

Definitions for Test Case Mutation

Example



Yahtzee

Amount of Players:

3

Amount of Rounds:

Amount of Rounds and Players must not be zero.

Transition (v_i, v_{i+1}, z_i, d_i)

A **4-tuple**:

1. v_i: Prev State
2. v_{i+1}: Next State
3. z_i: Action
4. d_i: Delay in seconds

Example

1. $(\_, v1, \mathit{reinitialize}, 15)$
2. $(v1, v1, \mathit{text}1, 1)$
3. $(v1, v2, \mathit{click}, 2)$

Mutation Operators

M1 **Loop Stressing:** Repeatedly execute all looping actions.

M2 **Pause-Resume:** Inject a pause-resume pair between all consecutive actions.

M3 **Change Text:** Replace a random text action with an abnormal text. (*emptytext*, *dottext*, and *longtext*)

M4 **Toggle Contextual State:** Inject random contextual state toggling between all consecutive actions. (Toggle WiFi, GPS, Bluetooth etc.)

M5 **Remove Delays:** Set all delays to 0.

M6 **Faster Swipe:** Increase the speed of all swipe actions.

Mutation Operators

M1 **Loop Stressing:** Repeatedly execute all looping actions.

Mutation Operators

M1 **Loop Stressing:** Repeatedly execute all looping actions.
Example:
$t = (\_, v1, \mathrm{reinit}, 15), (v1, v1, \mathrm{click}, 2), (v1, v2, \mathrm{swipe}\ "500", 2), (v2, v2, \mathrm{back}, 2)$

Mutation Operators

M1 **Loop Stressing:** Repeatedly execute all looping actions.
Example:
$t = (\_, v1, \text{reinit}, 15), (v1, v1, \text{click}, 2), (v1, v2, \text{swipe "500"}, 2), (v2, v2, \text{back}, 2)$

$\delta(t) =$

$(\_, v1, \text{reinit}, 15), (v1, v1, \text{click}, 1)^m, (v1, v2, \text{swipe "500"}, 2), (v2, v2, \text{back}, 1)^m$

Mutation Operators

M1 **Loop Stressing:** Repeatedly execute all looping actions.

M2 **Pause-Resume:** Inject a pause-resume pair between all consecutive actions.

Mutation Operators

M1 **Loop Stressing:** Repeatedly execute all looping actions.

M2 **Pause-Resume:** Inject a pause-resume pair between all consecutive actions.

Example:

$t = \ldots, (v1, v2, \text{click}, 2), (v2, v3, \text{swipe "500"}, 2), \ldots$

Mutation Operators

M1 **Loop Stressing:** Repeatedly execute all looping actions.

M2 **Pause-Resume:** Inject a pause-resume pair between all consecutive actions.

Example:

$t = \ldots, (v1, v2, \text{click}, 2), (v2, v3, \text{swipe "500"}, 2), \ldots$

$\delta(t) =$

$\ldots, (v1, v2, \text{click}, 2,), (v2, \_, \text{doze off}, 2), (\_, v2, \text{doze on}, 2), (v2, v3, \text{swipe "500"}, 2), \ldots$

Mutation Operators

M1 **Loop Stressing:** Repeatedly execute all looping actions.

M2 **Pause-Resume:** Inject a pause-resume pair between all consecutive actions.

M3 **Change Text:** Replace a random text action with an abnormal text. (*emptytext*, *dottext*, and *longtext*)

Mutation Operators

M1 **Loop Stressing:** Repeatedly execute all looping actions.

M2 **Pause-Resume:** Inject a pause-resume pair between all consecutive actions.

M3 **Change Text:** Replace a random text action with an abnormal text. (*emptytext*, *dottext*, and *longtext*)
Example:
$t = \ldots, (v1, v1, \text{text "www.url.com"}, 4), \ldots$

Mutation Operators

M1 **Loop Stressing:** Repeatedly execute all looping actions.

M2 **Pause-Resume:** Inject a pause-resume pair between all consecutive actions.

M3 **Change Text:** Replace a random text action with an abnormal text. (*emptytext*, *dottext*, and *longtext*)

Example:

$t = \ldots, (v1, v1, \text{text "www.url.com"}, 4), \ldots$

$\delta(t) = \ldots, (v1, v1, \text{text "."}, 4), \ldots$

Mutation Operators

M1 **Loop Stressing:** Repeatedly execute all looping actions.

M2 **Pause-Resume:** Inject a pause-resume pair between all consecutive actions.

M3 **Change Text:** Replace a random text action with an abnormal text. (*emptytext*, *dottext*, and *longtext*)

M4 **Toggle Contextual State:** Inject random contextual state toggling between all consecutive actions. (Toggle WiFi, GPS, Bluetooth etc.)

Mutation Operators

M1 **Loop Stressing:** Repeatedly execute all looping actions.

M2 **Pause-Resume:** Inject a pause-resume pair between all consecutive actions.

M3 **Change Text:** Replace a random text action with an abnormal text. (*emptytext*, *dottext*, and *longtext*)

M4 **Toggle Contextual State:** Inject random contextual state toggling between all consecutive actions. (Toggle WiFi, GPS, Bluetooth etc.)

Example:

$t = \ldots, (v1, v2, \text{click}, 2), (v2, v3, \text{swipe "500"}, 2), \ldots$

Mutation Operators

M1 **Loop Stressing:** Repeatedly execute all looping actions.

M2 **Pause-Resume:** Inject a pause-resume pair between all consecutive actions.

M3 **Change Text:** Replace a random text action with an abnormal text. (*emptytext*, *dottext*, and *longtext*)

M4 **Toggle Contextual State:** Inject random contextual state toggling between all consecutive actions. (Toggle WiFi, GPS, Bluetooth etc.)

Example:

$t = \ldots, (v1, v2, \text{click}, 2), (v2, v3, \text{swipe "500"}, 2), \ldots$

$\delta(t) = \ldots, (v1, v2, \text{click}, 2), (v2, v2, \text{GPS toggle}, 2), (v2, v3, \text{swipe "500"}, 2), \ldots$

Mutation Operators

M1 **Loop Stressing:** Repeatedly execute all looping actions.

M2 **Pause-Resume:** Inject a pause-resume pair between all consecutive actions.

M3 **Change Text:** Replace a random text action with an abnormal text. (*emptytext*, *dottext*, and *longtext*)

M4 **Toggle Contextual State:** Inject random contextual state toggling between all consecutive actions. (Toggle WiFi, GPS, Bluetooth etc.)

M5 **Remove Delays:** Set all delays to 0.

Mutation Operators

M1 **Loop Stressing:** Repeatedly execute all looping actions.

M2 **Pause-Resume:** Inject a pause-resume pair between all consecutive actions.

M3 **Change Text:** Replace a random text action with an abnormal text. (*emptytext*, *dottext*, and *longtext*)

M4 **Toggle Contextual State:** Inject random contextual state toggling between all consecutive actions. (Toggle WiFi, GPS, Bluetooth etc.)

M5 **Remove Delays:** Set all delays to 0.
Example:
$t = \ldots, (v1, v2, \text{swipe "500"}, 2), \ldots$

Mutation Operators

M1 **Loop Stressing:** Repeatedly execute all looping actions.

M2 **Pause-Resume:** Inject a pause-resume pair between all consecutive actions.

M3 **Change Text:** Replace a random text action with an abnormal text. (*emptytext*, *dottext*, and *longtext*)

M4 **Toggle Contextual State:** Inject random contextual state toggling between all consecutive actions. (Toggle WiFi, GPS, Bluetooth etc.)

M5 **Remove Delays:** Set all delays to 0.
Example:

$t = \ldots, (v1, v2, \text{swipe "500"}, 2), \ldots$

$\delta(t) = \ldots, (v1, v2, \text{swipe "500"}, 0), \ldots$

Mutation Operators

M1 **Loop Stressing:** Repeatedly execute all looping actions.

M2 **Pause-Resume:** Inject a pause-resume pair between all consecutive actions.

M3 **Change Text:** Replace a random text action with an abnormal text. (*emptytext*, *dottext*, and *longtext*)

M4 **Toggle Contextual State:** Inject random contextual state toggling between all consecutive actions. (Toggle WiFi, GPS, Bluetooth etc.)

M5 **Remove Delays:** Set all delays to 0.

M6 **Faster Swipe:** Increase the speed of all swipe actions.

Mutation Operators

M1 **Loop Stressing:** Repeatedly execute all looping actions.

M2 **Pause-Resume:** Inject a pause-resume pair between all consecutive actions.

M3 **Change Text:** Replace a random text action with an abnormal text. (*emptytext*, *dottext*, and *longtext*)

M4 **Toggle Contextual State:** Inject random contextual state toggling between all consecutive actions. (Toggle WiFi, GPS, Bluetooth etc.)

M5 **Remove Delays:** Set all delays to 0.

M6 **Faster Swipe:** Increase the speed of all swipe actions.
Example:
$t = \ldots, (v1, v2, \text{swipe "500"}, 2), \ldots$

Mutation Operators

M1 **Loop Stressing:** Repeatedly execute all looping actions.

M2 **Pause-Resume:** Inject a pause-resume pair between all consecutive actions.

M3 **Change Text:** Replace a random text action with an abnormal text. (*emptytext*, *dottext*, and *longtext*)

M4 **Toggle Contextual State:** Inject random contextual state toggling between all consecutive actions. (Toggle WiFi, GPS, Bluetooth etc.)

M5 **Remove Delays:** Set all delays to 0.

M6 **Faster Swipe:** Increase the speed of all swipe actions.
Example:
$t = \ldots, (v1, v2, \text{swipe "500"}, 2), \ldots$

$\delta(t) = \ldots, (v1, v2, \text{swipe "0"}, 2), \ldots$

Mutation Operators

M1 **Loop Stressing:** Repeatedly execute all looping actions.

M2 **Pause-Resume:** Inject a pause-resume pair between all consecutive actions.

M3 **Change Text:** Replace a random text action with an abnormal text. (*emptytext*, *dottext*, and *longtext*)

M4 **Toggle Contextual State:** Inject random contextual state toggling between all consecutive actions. (Toggle WiFi, GPS, Bluetooth etc.)

M5 **Remove Delays:** Set all delays to 0.

M6 **Faster Swipe:** Increase the speed of all swipe actions.

Case Study I

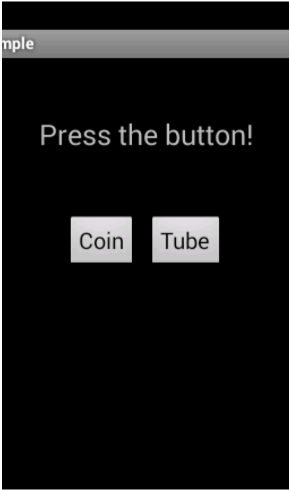


Figure: Soundboard App

- Produces **sounds** whenever buttons are clicked.
- Apply **(M1) Loop Stressing**.
- **Crashes** the 3rd party library.
- **Result:** All sounds are muted.

Case Study II

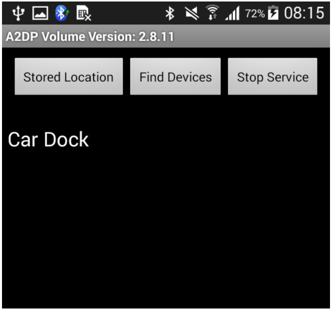


Figure: A2DPVol App

- Bluetooth Application.
- Normally, bluetooth is **off**.
- Apply **(M4) Toggle Contextual State**.
- **Result:** "Find Devices" button crashes the app.

Case Study II

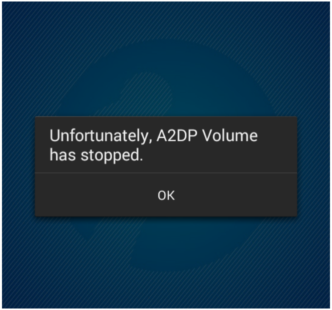
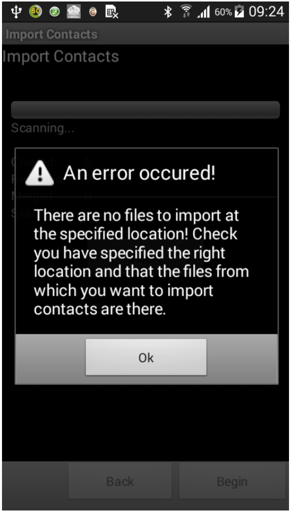

Unfortunately, A2DP Volume has stopped.

Figure: A2DPVol App

- Bluetooth Application.
- Normally, bluetooth is **off**.
- Apply **(M4) Toggle Contextual State**.
- **Result:** "Find Devices" button crashes the app.

Case Study III



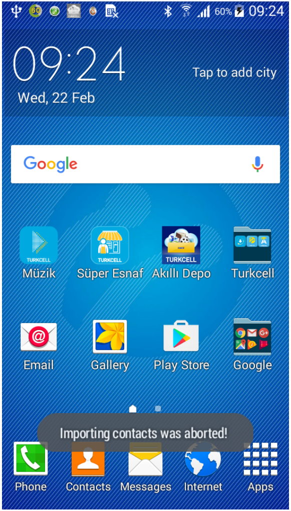
An error occured!

There are no files to import at the specified location! Check you have specified the right location and that the files from which you want to import contacts are there.

Figure: Import Contacts App

- **Existing test case** uncovered a warning message.
  (NOT A CRASH)
- Apply **(M2) Pause-Resume**.
- **Result:** Crash.

Case Study III

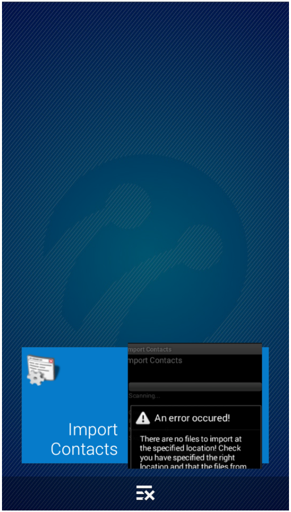


Importing contacts was aborted!

Figure: Import Contacts App

- **Existing test case** uncovered a warning message. (NOT A CRASH)
- Apply **(M2) Pause-Resume**.
- **Result:** Crash.

Case Study III



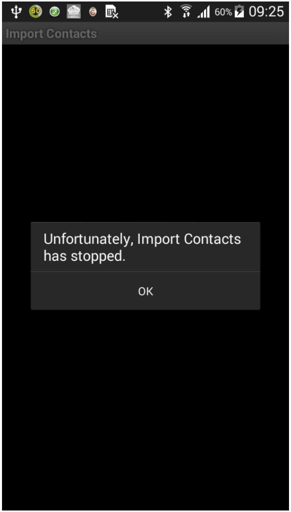
An error occured!

There are no files to import at the specified location! Check you have specified the right location and that the files from

Figure: Import Contacts App

- **Existing test case** uncovered a warning message.
  (NOT A CRASH)
- Apply **(M2) Pause-Resume**.
- **Result:** Crash.

Case Study III

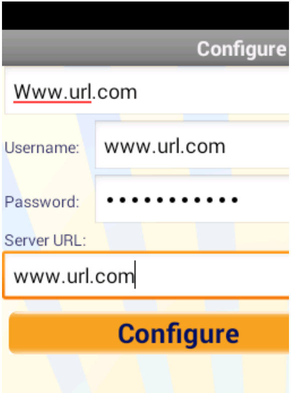


Unfortunately, Import Contacts has stopped.

Figure: Import Contacts App

- **Existing test case** uncovered a warning message.
  (NOT A CRASH)
- Apply **(M2) Pause-Resume**.
- **Result:** Crash.

Case Study IV



Www.url.com

Username: www.url.com

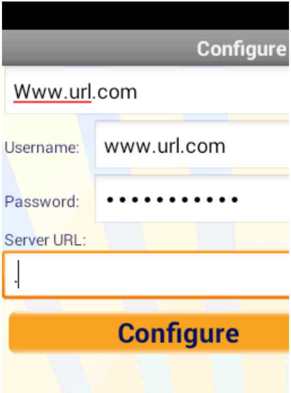
Password: ••••••••••••

Server URL: www.url.com

Figure: aCal App

- Calender application.
- URL box expects a URL.
- Apply **(M3) Change Text**.
- **Result:** Crash.

Case Study IV



Www.url.com

Username: www.url.com

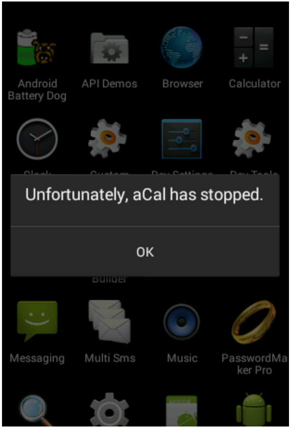
Password: ••••••••••••

Server URL:

Figure: aCal App

- Calender application.
- URL box expects a URL.
- Apply **(M3) Change Text**.
- **Result:** Crash.

Case Study IV

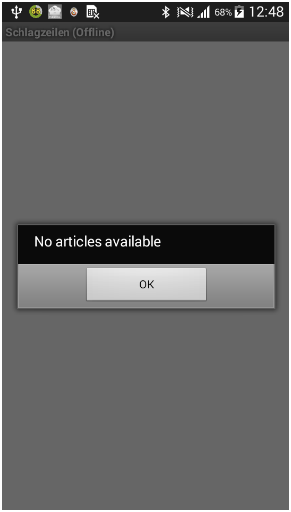


Unfortunately, aCal has stopped.

Figure: aCal App

- Calender application.
- URL box expects a URL.
- Apply **(M3) Change Text**.
- **Result:** Crash.

Case Study V



12:48

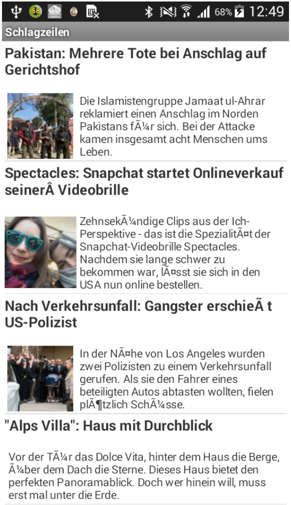
Schlagzeilen (Offline)

No articles available

Figure: Mirrored App

- News application.
- WiFi is currently **off**.
- Application correctly gives warning.
- Apply **(M4) Toggle Contextual State**.
- **Result:** Application crashes whenever a topic is clicked.

Case Study V



Schlagzeilen

Pakistan: Mehrere Tote bei Anschlag auf Gerichtshof

Die Islamistengruppe Jamaat ul-Ahrar reklamiert einen Anschlag im Norden Pakistans fÃ¼r sich. Bei der Attacke kamen insgesamt acht Menschen ums Leben.

Spectacles: Snapchat startet Onlineverkauf seinerÂ Videobrille

ZehnsekÃ¼ndige Clips aus der Ich-Perspektive - das ist die SpezialitÃ¤t der Snapchat-Videobrille Spectacles. Nachdem sie lange schwer zu bekommen war, lÃ¤sst sie sich in den USA nun online bestellen.

Nach Verkehrsunfall: Gangster erschieÃ t US-Polizist

In der NÃ¤he von Los Angeles wurden zwei Polizisten zu einem Verkehrsunfall gerufen. Als sie den Fahrer eines beteiligten Autos abtasten wollten, fielen plÃ¶tzlich SchÃ¼sse.

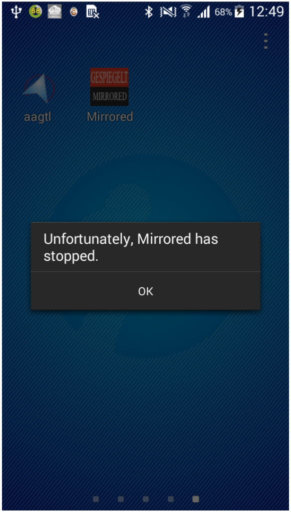
"Alps Villa": Haus mit Durchblick

Vor der TÃ¼r das Dolce Vita, hinter dem Haus die Berge, Ã¼ber dem Dach die Sterne. Dieses Haus bietet den perfekten Panoramablick. Doch wer hinein will, muss erst mal unter die Erde.

Figure: Mirrored App

- News application.
- WiFi is currently **off**.
- Application correctly gives warning.
- Apply **(M4) Toggle Contextual State**.
- **Result:** Application crashes whenever a topic is clicked.

Case Study V

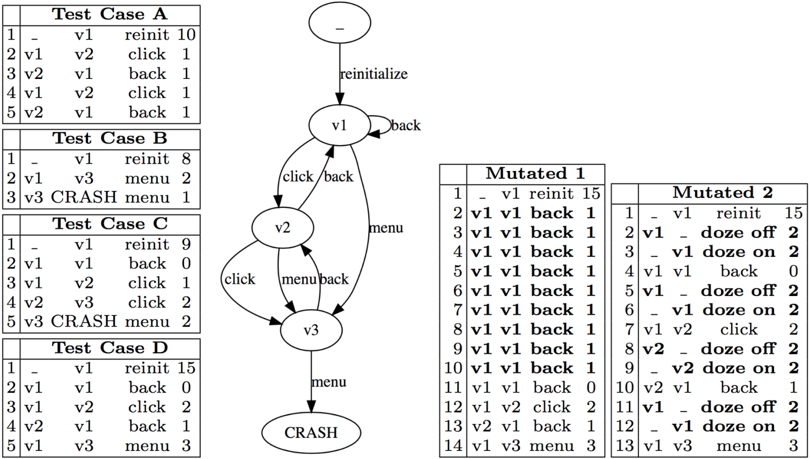


Unfortunately, Mirrored has stopped.

Figure: Mirrored App

- News application.
- WiFi is currently **off**.
- Application correctly gives warning.
- Apply **(M4) Toggle Contextual State**.
- **Result:** Application crashes whenever a topic is clicked.

A Working Example



	Test Case A			
1	_	v1	reinit	10
2	v1	v2	click	1
3	v2	v1	back	1
4	v1	v2	click	1
5	v2	v1	back	1

	Test Case B			
1	_	v1	reinit	8
2	v1	v3	menu	2
3	v3	CRASH	menu	1

	Test Case C			
1	_	v1	reinit	9
2	v1	v1	back	0
3	v1	v2	click	1
4	v2	v3	click	2
5	v3	CRASH	menu	2

	Test Case D			
1	_	v1	reinit	15
2	v1	v1	back	0
3	v1	v2	click	2
4	v2	v1	back	1
5	v1	v3	menu	3

(a) Test Cases

(b) AUT Model

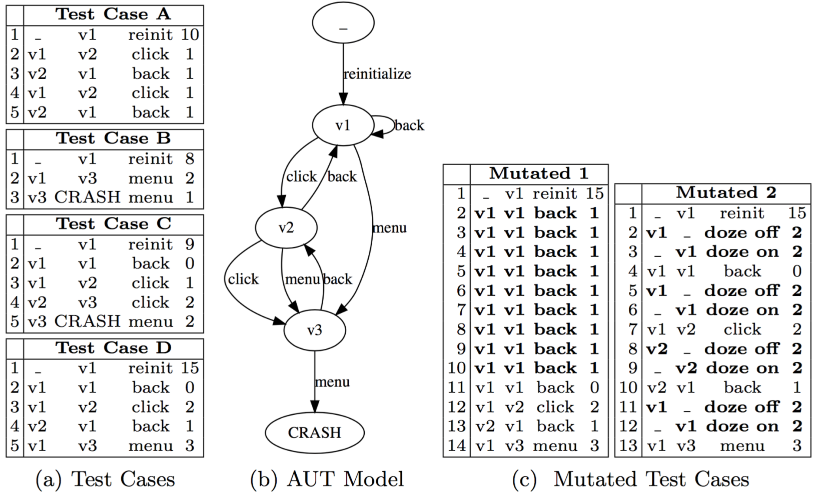
	Mutated 1			
1	_	v1	reinit	15
2	**v1**	**v1**	**back**	**1**
3	**v1**	**v1**	**back**	**1**
4	**v1**	**v1**	**back**	**1**
5	**v1**	**v1**	**back**	**1**
6	**v1**	**v1**	**back**	**1**
7	**v1**	**v1**	**back**	**1**
8	**v1**	**v1**	**back**	**1**
9	**v1**	**v1**	**back**	**1**
10	**v1**	**v1**	**back**	**1**
11	v1	v1	back	0
12	v1	v2	click	2
13	v2	v1	back	1
14	v1	v3	menu	3

	Mutated 2			
1	_	v1	reinit	15
2	**v1**	**_**	**doze off**	**2**
3	**_**	**v1**	**doze on**	**2**
4	v1	v1	back	0
5	**v1**	**_**	**doze off**	**2**
6	**_**	**v1**	**doze on**	**2**
7	v1	v2	click	2
8	**v2**	**_**	**doze off**	**2**
9	**_**	**v2**	**doze on**	**2**
10	v2	v1	back	1
11	**v1**	**_**	**doze off**	**2**
12	**_**	**v1**	**doze on**	**2**
13	v1	v3	menu	3

(c) Mutated Test Cases

(a) and (b) are generated by AndroFrame (QBE) for a toy app.

A Working Example



	Test Case A			
1	_	v1	reinit	10
2	v1	v2	click	1
3	v2	v1	back	1
4	v1	v2	click	1
5	v2	v1	back	1

	Test Case B			
1	_	v1	reinit	8
2	v1	v3	menu	2
3	v3	CRASH	menu	1

	Test Case C			
1	_	v1	reinit	9
2	v1	v1	back	0
3	v1	v2	click	1
4	v2	v3	click	2
5	v3	CRASH	menu	2

	Test Case D			
1	_	v1	reinit	15
2	v1	v1	back	0
3	v1	v2	click	2
4	v2	v1	back	1
5	v1	v3	menu	3

(a) Test Cases

(b) AUT Model

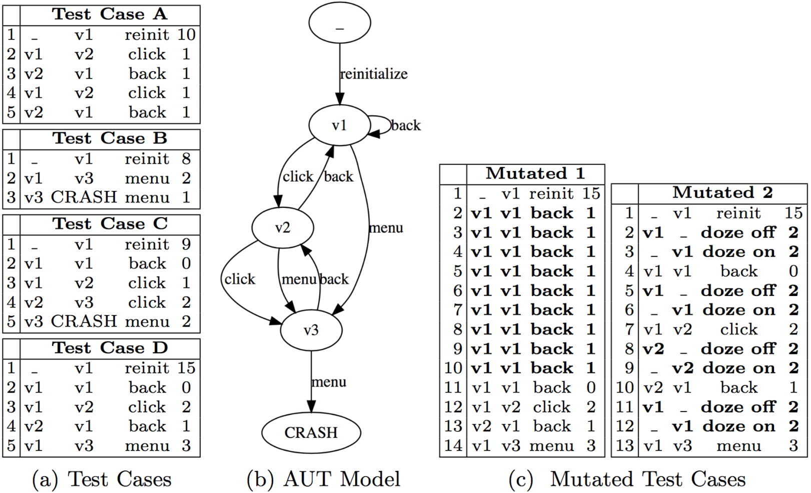
	Mutated 1			
1	_	v1	reinit	15
2	**v1**	**v1**	**back**	**1**
3	**v1**	**v1**	**back**	**1**
4	**v1**	**v1**	**back**	**1**
5	**v1**	**v1**	**back**	**1**
6	**v1**	**v1**	**back**	**1**
7	**v1**	**v1**	**back**	**1**
8	**v1**	**v1**	**back**	**1**
9	**v1**	**v1**	**back**	**1**
10	**v1**	**v1**	**back**	**1**
11	v1	v1	back	0
12	v1	v2	click	2
13	v2	v1	back	1
14	v1	v3	menu	3

	Mutated 2			
1	_	v1	reinit	15
2	**v1**	_	**doze off**	**2**
3	_	**v1**	**doze on**	**2**
4	v1	v1	back	0
5	**v1**	_	**doze off**	**2**
6	_	**v1**	**doze on**	**2**
7	v1	v2	click	2
8	**v2**	_	**doze off**	**2**
9	_	**v2**	**doze on**	**2**
10	v2	v1	back	1
11	**v1**	_	**doze off**	**2**
12	_	**v1**	**doze on**	**2**
13	v1	v3	menu	3

(c) Mutated Test Cases

Want to **increase** crashes in the test suite $TS = \{A, B, C, D\}$

A Working Example



	Test Case A			
1	_	v1	reinit	10
2	v1	v2	click	1
3	v2	v1	back	1
4	v1	v2	click	1
5	v2	v1	back	1

	Test Case B			
1	_	v1	reinit	8
2	v1	v3	menu	2
3	v3	CRASH	menu	1

	Test Case C			
1	_	v1	reinit	9
2	v1	v1	back	0
3	v1	v2	click	1
4	v2	v3	click	2
5	v3	CRASH	menu	2

	Test Case D			
1	_	v1	reinit	15
2	v1	v1	back	0
3	v1	v2	click	2
4	v2	v1	back	1
5	v1	v3	menu	3

(a) Test Cases

(b) AUT Model

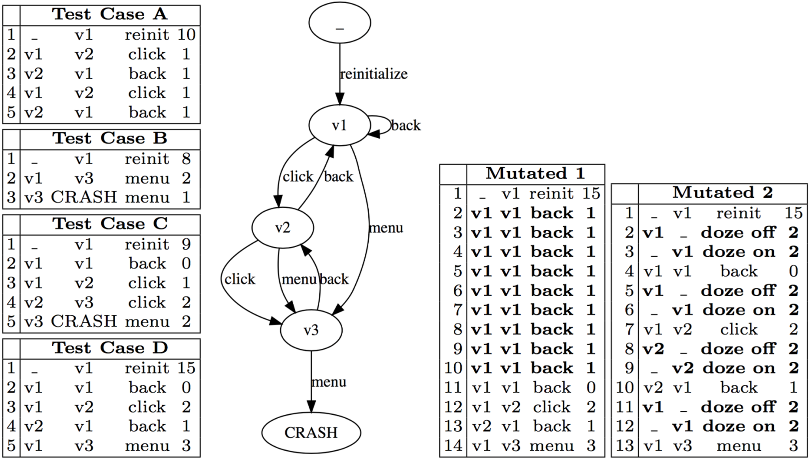
	Mutated 1			
1	_	v1	reinit	15
2	**v1**	**v1**	**back**	**1**
3	**v1**	**v1**	**back**	**1**
4	**v1**	**v1**	**back**	**1**
5	**v1**	**v1**	**back**	**1**
6	**v1**	**v1**	**back**	**1**
7	**v1**	**v1**	**back**	**1**
8	**v1**	**v1**	**back**	**1**
9	**v1**	**v1**	**back**	**1**
10	**v1**	**v1**	**back**	**1**
11	v1	v1	back	0
12	v1	v2	click	2
13	v2	v1	back	1
14	v1	v3	menu	3

	Mutated 2			
1	_	v1	reinit	15
2	**v1**	**_**	**doze off**	**2**
3	**_**	**v1**	**doze on**	**2**
4	v1	v1	back	0
5	**v1**	**_**	**doze off**	**2**
6	**_**	**v1**	**doze on**	**2**
7	v1	v2	click	2
8	**v2**	**_**	**doze off**	**2**
9	**_**	**v2**	**doze on**	**2**
10	v2	v1	back	1
11	**v1**	**_**	**doze off**	**2**
12	**_**	**v1**	**doze on**	**2**
13	v1	v3	menu	3

(c) Mutated Test Cases

Take **non-crashing** test cases $\{A, D\}$

A Working Example



	Test Case A			
1	_	v1	reinit	10
2	v1	v2	click	1
3	v2	v1	back	1
4	v1	v2	click	1
5	v2	v1	back	1

	Test Case B			
1	_	v1	reinit	8
2	v1	v3	menu	2
3	v3	CRASH	menu	1

	Test Case C			
1	_	v1	reinit	9
2	v1	v1	back	0
3	v1	v2	click	1
4	v2	v3	click	2
5	v3	CRASH	menu	2

	Test Case D			
1	_	v1	reinit	15
2	v1	v1	back	0
3	v1	v2	click	2
4	v2	v1	back	1
5	v1	v3	menu	3

(a) Test Cases

(b) AUT Model

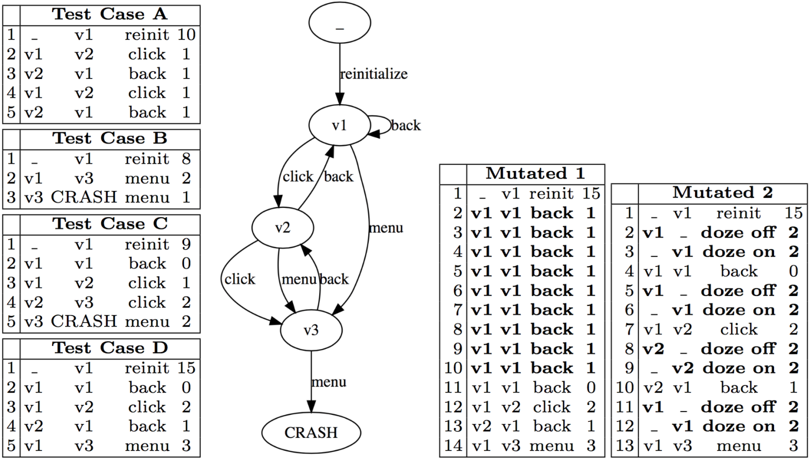
	Mutated 1			
1	_	v1	reinit	15
2	**v1**	**v1**	**back**	**1**
3	**v1**	**v1**	**back**	**1**
4	**v1**	**v1**	**back**	**1**
5	**v1**	**v1**	**back**	**1**
6	**v1**	**v1**	**back**	**1**
7	**v1**	**v1**	**back**	**1**
8	**v1**	**v1**	**back**	**1**
9	**v1**	**v1**	**back**	**1**
10	**v1**	**v1**	**back**	**1**
11	v1	v1	back	0
12	v1	v2	click	2
13	v2	v1	back	1
14	v1	v3	menu	3

	Mutated 2			
1	_	v1	reinit	15
2	**v1**	**_**	**doze off**	**2**
3	**_**	**v1**	**doze on**	**2**
4	v1	v1	back	0
5	**v1**	**_**	**doze off**	**2**
6	**_**	**v1**	**doze on**	**2**
7	v1	v2	click	2
8	**v2**	**_**	**doze off**	**2**
9	**_**	**v2**	**doze on**	**2**
10	v2	v1	back	1
11	**v1**	**_**	**doze off**	**2**
12	**_**	**v1**	**doze on**	**2**
13	v1	v3	menu	3

(c) Mutated Test Cases

D **subsumes** A (i.e. D has all transitions that A has)

A Working Example



	Test Case A			
1	_	v1	reinit	10
2	v1	v2	click	1
3	v2	v1	back	1
4	v1	v2	click	1
5	v2	v1	back	1

	Test Case B			
1	_	v1	reinit	8
2	v1	v3	menu	2
3	v3	CRASH	menu	1

	Test Case C			
1	_	v1	reinit	9
2	v1	v1	back	0
3	v1	v2	click	1
4	v2	v3	click	2
5	v3	CRASH	menu	2

	Test Case D			
1	_	v1	reinit	15
2	v1	v1	back	0
3	v1	v2	click	2
4	v2	v1	back	1
5	v1	v3	menu	3

(a) Test Cases

(b) AUT Model

	Mutated 1			
1	_	v1	reinit	15
2	**v1**	**v1**	**back**	**1**
3	**v1**	**v1**	**back**	**1**
4	**v1**	**v1**	**back**	**1**
5	**v1**	**v1**	**back**	**1**
6	**v1**	**v1**	**back**	**1**
7	**v1**	**v1**	**back**	**1**
8	**v1**	**v1**	**back**	**1**
9	**v1**	**v1**	**back**	**1**
10	**v1**	**v1**	**back**	**1**
11	v1	v1	back	0
12	v1	v2	click	2
13	v2	v1	back	1
14	v1	v3	menu	3

	Mutated 2			
1	_	v1	reinit	15
2	**v1**	_	**doze off**	**2**
3	_	**v1**	**doze on**	**2**
4	v1	v1	back	0
5	**v1**	_	**doze off**	**2**
6	_	**v1**	**doze on**	**2**
7	v1	v2	click	2
8	**v2**	_	**doze off**	**2**
9	_	**v2**	**doze on**	**2**
10	v2	v1	back	1
11	**v1**	_	**doze off**	**2**
12	_	**v1**	**doze on**	**2**
13	v1	v3	menu	3

(c) Mutated Test Cases

Remove *A* and get $\{D\}$ (**Minimization**)

A Working Example

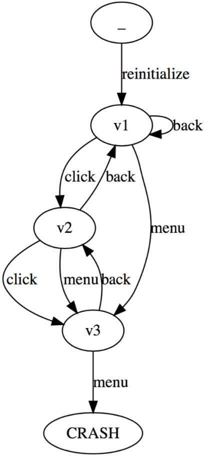
	Test Case A			
1	_	v1	reinit	10
2	v1	v2	click	1
3	v2	v1	back	1
4	v1	v2	click	1
5	v2	v1	back	1

	Test Case B			
1	_	v1	reinit	8
2	v1	v3	menu	2
3	v3	CRASH	menu	1

	Test Case C			
1	_	v1	reinit	9
2	v1	v1	back	0
3	v1	v2	click	1
4	v2	v3	click	2
5	v3	CRASH	menu	2

	Test Case D			
1	_	v1	reinit	15
2	v1	v1	back	0
3	v1	v2	click	2
4	v2	v1	back	1
5	v1	v3	menu	3

(a) Test Cases



(b) AUT Model

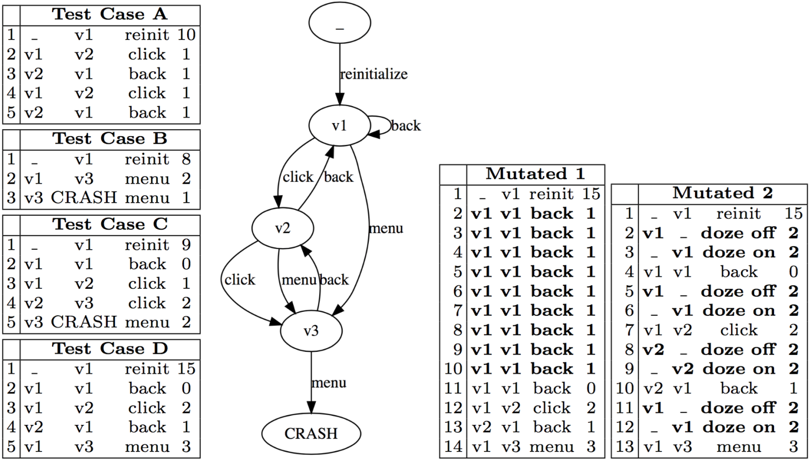
	Mutated 1			
1	_	v1	reinit	15
2	**v1**	**v1**	**back**	**1**
3	**v1**	**v1**	**back**	**1**
4	**v1**	**v1**	**back**	**1**
5	**v1**	**v1**	**back**	**1**
6	**v1**	**v1**	**back**	**1**
7	**v1**	**v1**	**back**	**1**
8	**v1**	**v1**	**back**	**1**
9	**v1**	**v1**	**back**	**1**
10	**v1**	**v1**	**back**	**1**
11	v1	v1	back	0
12	v1	v2	click	2
13	v2	v1	back	1
14	v1	v3	menu	3

	Mutated 2			
1	_	v1	reinit	15
2	**v1**	**_**	**doze off**	**2**
3	**_**	**v1**	**doze on**	**2**
4	v1	v1	back	0
5	**v1**	**_**	**doze off**	**2**
6	**_**	**v1**	**doze on**	**2**
7	v1	v2	click	2
8	**v2**	**_**	**doze off**	**2**
9	**_**	**v2**	**doze on**	**2**
10	v2	v1	back	1
11	**v1**	**_**	**doze off**	**2**
12	**_**	**v1**	**doze on**	**2**
13	v1	v3	menu	3

(c) Mutated Test Cases

Why minimization? To **narrow down** all possible mutations.

A Working Example



	Test Case A			
1	_	v1	reinit	10
2	v1	v2	click	1
3	v2	v1	back	1
4	v1	v2	click	1
5	v2	v1	back	1

	Test Case B			
1	_	v1	reinit	8
2	v1	v3	menu	2
3	v3	CRASH	menu	1

	Test Case C			
1	_	v1	reinit	9
2	v1	v1	back	0
3	v1	v2	click	1
4	v2	v3	click	2
5	v3	CRASH	menu	2

	Test Case D			
1	_	v1	reinit	15
2	v1	v1	back	0
3	v1	v2	click	2
4	v2	v1	back	1
5	v1	v3	menu	3

(a) Test Cases

(b) AUT Model

	Mutated 1			
1	_	v1	reinit	15
2	**v1**	**v1**	**back**	**1**
3	**v1**	**v1**	**back**	**1**
4	**v1**	**v1**	**back**	**1**
5	**v1**	**v1**	**back**	**1**
6	**v1**	**v1**	**back**	**1**
7	**v1**	**v1**	**back**	**1**
8	**v1**	**v1**	**back**	**1**
9	**v1**	**v1**	**back**	**1**
10	**v1**	**v1**	**back**	**1**
11	v1	v1	back	0
12	v1	v2	click	2
13	v2	v1	back	1
14	v1	v3	menu	3

	Mutated 2			
1	_	v1	reinit	15
2	**v1**	**_**	**doze off**	**2**
3	**_**	**v1**	**doze on**	**2**
4	v1	v1	back	0
5	**v1**	**_**	**doze off**	**2**
6	**_**	**v1**	**doze on**	**2**
7	v1	v2	click	2
8	**v2**	**_**	**doze off**	**2**
9	**_**	**v2**	**doze on**	**2**
10	v2	v1	back	1
11	**v1**	**_**	**doze off**	**2**
12	**_**	**v1**	**doze on**	**2**
13	v1	v3	menu	3

(c) Mutated Test Cases

Apply **random mutations** to *D* to get Mutated 1 and Mutated 2

A Working Example

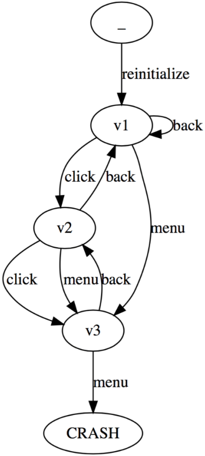
	Test Case A			
1	_	v1	reinit	10
2	v1	v2	click	1
3	v2	v1	back	1
4	v1	v2	click	1
5	v2	v1	back	1

	Test Case B			
1	_	v1	reinit	8
2	v1	v3	menu	2
3	v3	CRASH	menu	1

	Test Case C			
1	_	v1	reinit	9
2	v1	v1	back	0
3	v1	v2	click	1
4	v2	v3	click	2
5	v3	CRASH	menu	2

	Test Case D			
1	_	v1	reinit	15
2	v1	v1	back	0
3	v1	v2	click	2
4	v2	v1	back	1
5	v1	v3	menu	3

(a) Test Cases



(b) AUT Model

	Mutated 1			
1	_	v1	reinit	15
2	**v1**	**v1**	**back**	**1**
3	**v1**	**v1**	**back**	**1**
4	**v1**	**v1**	**back**	**1**
5	**v1**	**v1**	**back**	**1**
6	**v1**	**v1**	**back**	**1**
7	**v1**	**v1**	**back**	**1**
8	**v1**	**v1**	**back**	**1**
9	**v1**	**v1**	**back**	**1**
10	**v1**	**v1**	**back**	**1**
11	v1	v1	back	0
12	v1	v2	click	2
13	v2	v1	back	1
14	v1	v3	menu	3

	Mutated 2			
1	_	v1	reinit	15
2	**v1**	_	**doze off**	**2**
3	_	**v1**	**doze on**	**2**
4	v1	v1	back	0
5	**v1**	_	**doze off**	**2**
6	_	**v1**	**doze on**	**2**
7	v1	v2	click	2
8	**v2**	_	**doze off**	**2**
9	_	**v2**	**doze on**	**2**
10	v2	v1	back	1
11	**v1**	_	**doze off**	**2**
12	_	**v1**	**doze on**	**2**
13	v1	v3	menu	3

(c) Mutated Test Cases

All test cases in (c) result in **crashes**.

Experimental Setup

- 14 x Android-x86 VirtualBox guests (with Android 4.4.r5)
- **100 Android applications** randomly selected from F-Droid benchmarks
- Observed **the number of distinct crashes** across time.
  - **Common technique:** Determine crash distinctness by parsing reported stack traces.
- **20 minutes** for each application by each tool
  - TCM, AndroFrame (QBE), Sapienz, Monkey, PUMA, and A^{3}E.
- For TCM,
  1. 10 minutes, AndroFrame (QBE) **generates test cases**.
  2. 10 minutes, TCM **executes mutated test cases**.

Experimental Results

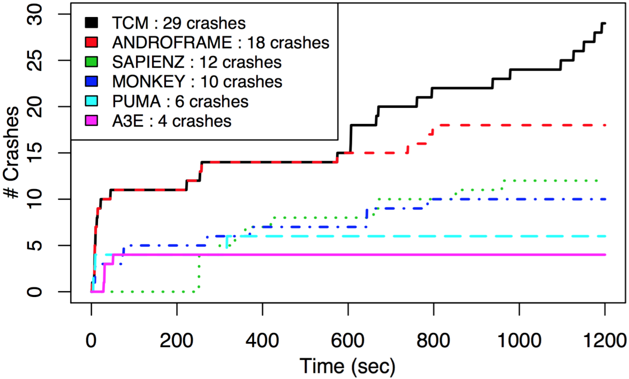


Figure: Number of Distinct Crashes Detected Across 20 Minutes

For TCM, we use AndroFrame (QBE) for the first 10 minutes.

Experimental Results

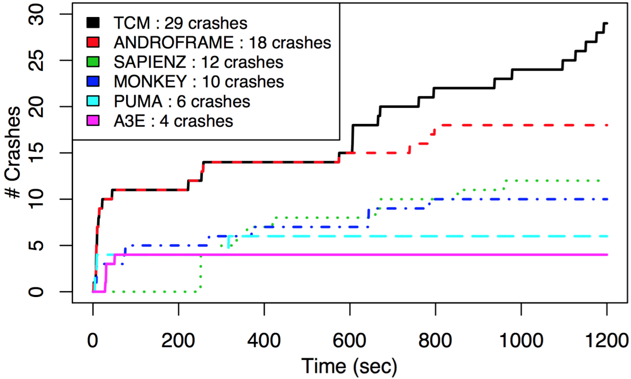


Figure: Number of Distinct Crashes Detected Across 20 Minutes

TCM detects **more crashes** than other tools.

Experimental Results

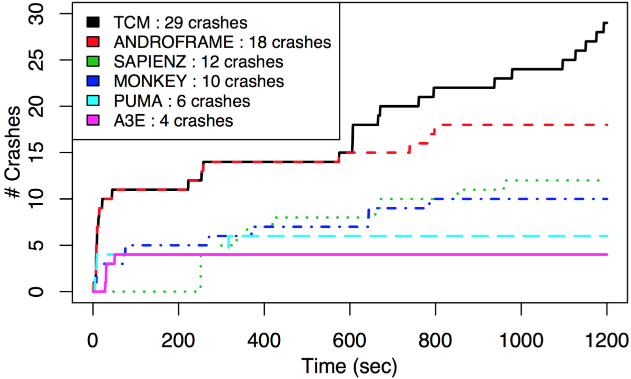


Figure: Number of Distinct Crashes Detected Across 20 Minutes

Other tools **converge** in 20 minutes whereas TCM does not.

Conclusion

Summary

- Proposed **Test Case Mutation (TCM)**.
- **6 mutation operators** for test cases of Android GUIs.
- Obtained test cases **automatically** using AndroFrame (QBE).
- Performed case studies and experiments to show the **effectiveness of TCM**.

Future Work

- **More GUI actions** (e.g. rotation and double-click)
- Sample mutations from a probability distribution based on **crash rates** instead of random.
- Find **the most optimal timings** for AndroFrame and TCM.

Thank You! Any Questions?

Appendix A: Crash Patterns

Table: Relating Crash Patterns and Mutation Operators

Crash Patterns	Mutation Operators
C1. Unhandled Exceptions	M1, M3, M6
C2. External Errors	M1, M4, M5, M6
C3. Resource Unavailability	M2, M5
C4. Semantic Errors	M3
C5. Network-Based Crashes	M4, M5, M6

C1. Unhandled Exceptions

Misuse of libraries or GUI components.
e.g. **Stressing**.

Appendix A: Crash Patterns

Table: Relating Crash Patterns and Mutation Operators

Crash Patterns	Mutation Operators
C1. Unhandled Exceptions	M1, M3, M6
C2. External Errors	M1, M4, M5, M6
C3. Resource Unavailability	M2, M5
C4. Semantic Errors	M3
C5. Network-Based Crashes	M4, M5, M6

C2. External Errors

1. Interact with another app **without sufficient permissions**,
2. Receive an intent with an **invalid bundle**,
3. Send an **invalid intent** and fail to receive answer,
4. **Shared memory is freed** by another application, etc.

Appendix A: Crash Patterns

Table: Relating Crash Patterns and Mutation Operators

Crash Patterns	Mutation Operators
C1. Unhandled Exceptions	M1, M3, M6
C2. External Errors	M1, M4, M5, M6
C3. Resource Unavailability	M2, M5
C4. Semantic Errors	M3
C5. Network-Based Crashes	M4, M5, M6

C3. Resource Unavailability

When the AUT is paused, it releases system resources.
The AUT may crash if it is **unable to allocate the system resources back** when it is resumed.

Appendix A: Crash Patterns

Table: Relating Crash Patterns and Mutation Operators

Crash Patterns	Mutation Operators
C1. Unhandled Exceptions	M1, M3, M6
C2. External Errors	M1, M4, M5, M6
C3. Resource Unavailability	M2, M5
C4. Semantic Errors	M3
C5. Network-Based Crashes	M4, M5, M6

C4. Semantic Errors

Fail to handle certain inputs given by the user.
e.g. Invalid texts.

Appendix A: Crash Patterns

Table: Relating Crash Patterns and Mutation Operators

Crash Patterns	Mutation Operators
C1. Unhandled Exceptions	M1, M3, M6
C2. External Errors	M1, M4, M5, M6
C3. Resource Unavailability	M2, M5
C4. Semantic Errors	M3
C5. Network-Based Crashes	M4, M5, M6

C5. Network-Based Crashes

- Server **unreachable**,
- Connectivity is **disabled** etc.

Appendix B: Table of GUI Actions

Table: List of GUI Actions for our Automated Testing Tool

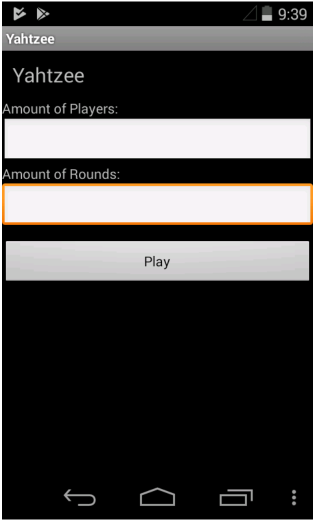
Non-contextual	Param1	Param2	Param3	Param4	Param5
click	x	y	-	-	-
longclick	x	y	-	-	-
text	x	y	string	-	-
swipe	x1	y1	x2	y2	duration
menu	-	-	-	-	-
back	-	-	-	-	-

Contextual	Parameters
connectivity	on/off/toggle
bluetooth	on/off/toggle
location	gps/gps&network/off/toggle
planemode	on/off/toggle
doze	on/off/toggle

Special	Param1	Param2	Param3	Param4	Param5
reinit	package	activity	-	-	-

Appendix C: Automatic Generation of GUI Models Example

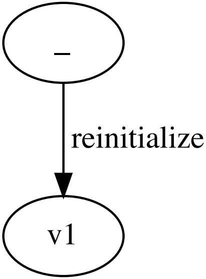
Action: reinitialize com.tum.yahtzee MainActivity



Yahtzee

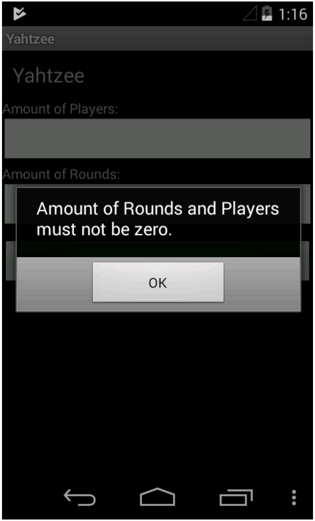
Amount of Players:

Amount of Rounds:

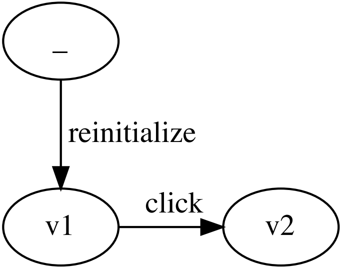


Appendix C: Automatic Generation of GUI Models Example

Action: click 200 390 (click play)

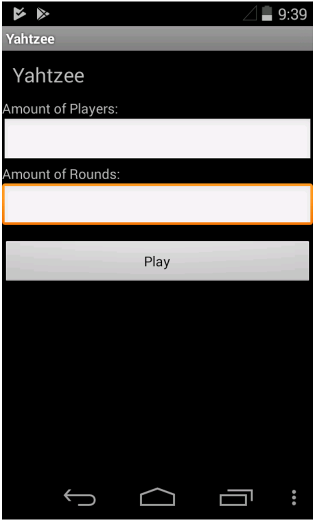


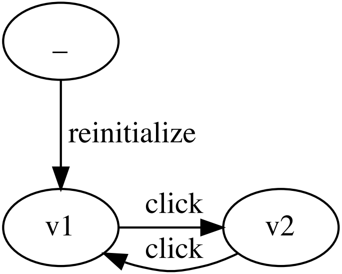
Amount of Rounds and Players must not be zero.



Appendix C: Automatic Generation of GUI Models Example

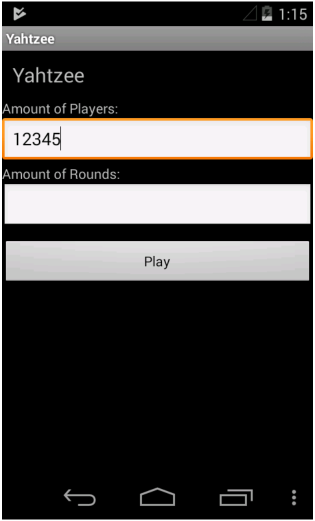
Action: click 200 410 (click ok)





Appendix C: Automatic Generation of GUI Models Example

Action: text 200 270 12345 (text1)



1:15

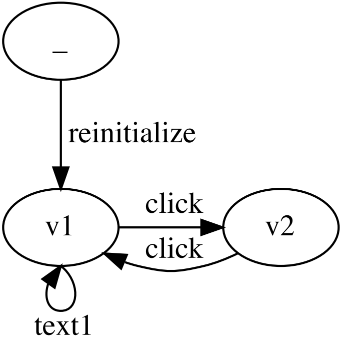
Yahtzee

Yahtzee

Amount of Players:

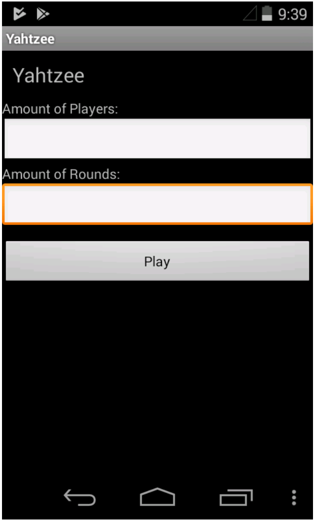
12345

Amount of Rounds:



Appendix C: Automatic Generation of GUI Models Example

Action: reinitialize com.tum.yahtzee MainActivity

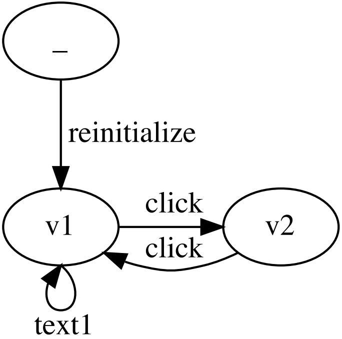

9:39

Yahtzee

Yahtzee

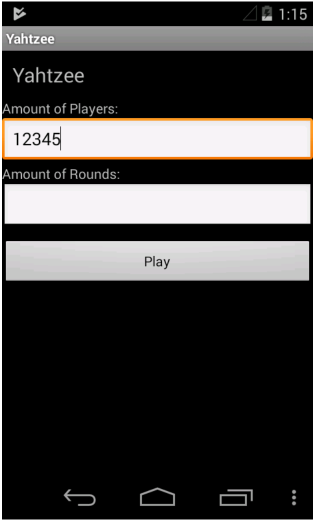
Amount of Players:

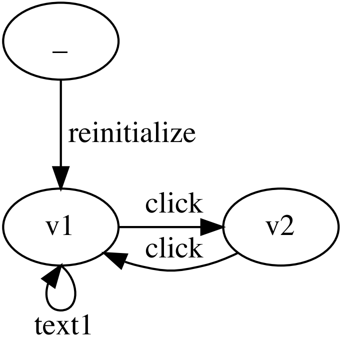
Amount of Rounds:



Appendix C: Automatic Generation of GUI Models Example

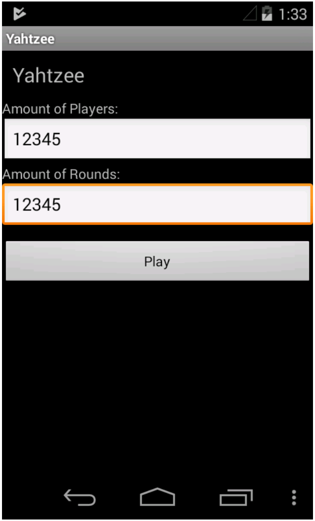
Action: text 200 270 12345 (text1)





Appendix C: Automatic Generation of GUI Models Example

Action: text 200 330 12345 (text2)



1:33

Yahtzee

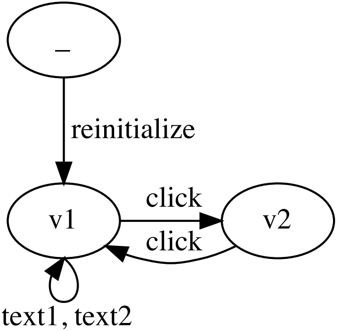
Yahtzee

Amount of Players:

12345

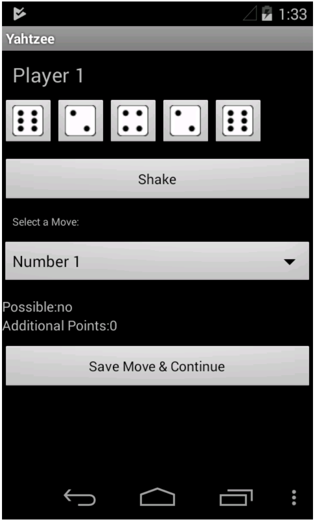
Amount of Rounds:

12345



Appendix C: Automatic Generation of GUI Models Example

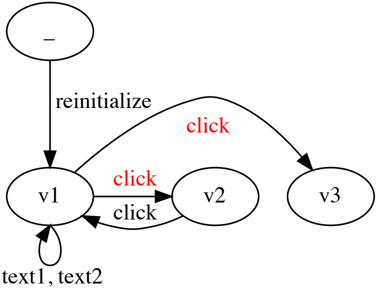
Action: click 200 390 (click play)



Select a Move:

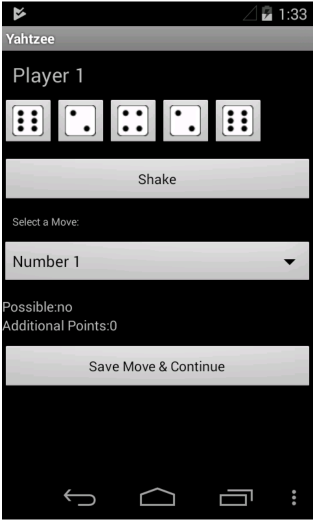
Number 1

Possible:no
Additional Points:0



Appendix C: Automatic Generation of GUI Models Example

Action: click 200 390 (click play)



1:33

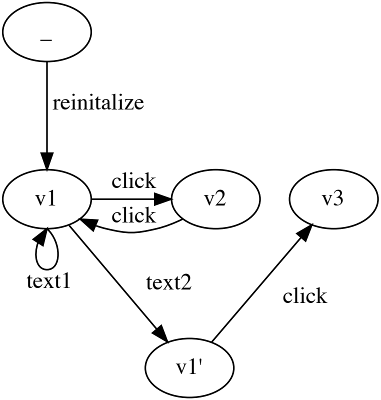
Yahtzee

Player 1

Select a Move:

Number 1

Possible:no
Additional Points:0



Appendix D: Benchmark Characteristics

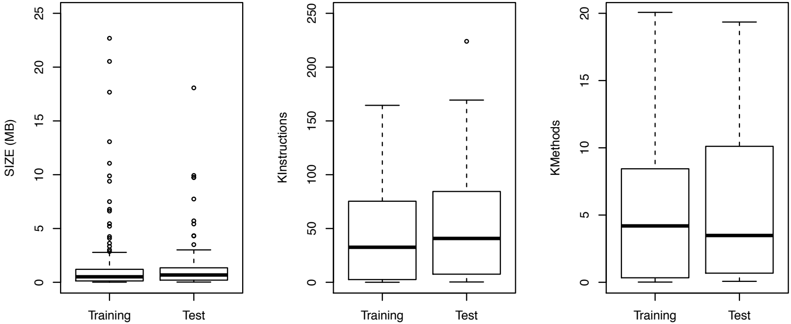


Figure: Characteristics of Training and Test Sets

Between

- 0.01-25 MB, 1000-250000 instructions, and 10-20000 methods