WODEL-EDU: AN MDE SOLUTION FOR THE GENERATION AND EVALUATION OF DIAGRAM-BASED EXERCISES

P. Gómez-Abajo, A. Rico-Fernández, E. Guerra, J. de Lara
Modelling & Software Engineering Research Group
http://miso.es
Universidad Autónoma de Madrid (Spain)
OVERVIEW OF THE APPROACH
OVERVIEW OF THE APPROACH

1. create the MM
2. define concrete-syntax
3. create seed models
4. define mutations
5. textual representation
6. exercises description
7. exercises solving
8. automated grading

domain expert

modelDraw
seed model synthesizer

Wodel
modelText
mutaText

eduTest

professor

Wodel-Edu demo
Thank you!

Wodel & Wodel-Edu website:
http://gomezabajo.github.io/Wodel/
CONCLUSIONS & FUTURE WORK

• Generation of exercises for DFA
• Currently, the evaluation of the exercises for DFA in a real university course is a work under review
• Generated for Moodle, mobile apps and the web
• Five different kinds of exercises currently supported

• Future work
  • New kinds of exercises
  • Exercises for other domains: class diagrams, electronic circuits, and so on
  • Evaluate the generated exercises in real university courses
PROBLEM

• There is a lack of frameworks to generate domain-independent exercises

• There is a need to generate a high number of exercises

• An assistance tool of this kind would alleviate much of the professors’ work

PROPOSED SOLUTION

• Extension of the Wodel tool for model mutation to generate and evaluate diagram-based exercises for any domain: Wodel-Edu
WODEL

DSL Wodel for model mutation with:

- High level mutation primitives
- Domain independent
- Compiled to Java code

Extensible execution services

- Mutant validation
- Equivalent mutants' detection
- Registry of the applied mutation operators
- Extensible for post-processing applications

Development services

- Seed model synthesis
- Mutation footprints
WODEL EXAMPLE

deterministic_2: Transition.allInstances()->select(t | t.src = self)->size() = Symbol.allInstances()->size()

<table>
<thead>
<tr>
<th>states</th>
<th>src</th>
<th>transitions</th>
<th>alphabet</th>
</tr>
</thead>
<tbody>
<tr>
<td>State</td>
<td></td>
<td>Transition</td>
<td>Symbol</td>
</tr>
<tr>
<td>name: String</td>
<td>isInitial: boolean</td>
<td>tar: State</td>
<td>symbol: String</td>
</tr>
</tbody>
</table>

one_initial: self.states->one(s | s.isInitial)
some_final: self.states->exists(s | s.isFinal)
alphabet: self.alphabet->forAll (a1, a2 | a1.symbol = a2.symbol implies a1 = a2)
diff_state_name: self.states->forAll(s1, s2 | s1 <> s2 implies s1.name <> s2.name)
connected: self.states->forAll(s | s.isInitial or Set(s)->closure(s1 |
    self.transitions->select(t | t.tar = s1)->collect(src))->exists(s2 | s2.isInitial))
deterministic_1: self.transitions->forAll(t1, t2 | (t1 <> t2 and t1.src = t2.src) implies t1.symbol <> t2.symbol))

generate 4 mutants in "out/" from "models/"
metamodel "http://fa.com"

with blocks {
  mts "modifies the target State of a Transition"
  {
    modify target tar from one Transition to other State
  }
}
WODEL-EDU PROCESS
WODEL-EDU ARCHITECTURE

- WODEL-EDU engine
  - SEED SYNTH.
  - MODEL DRAW
  - MODEL TEXT
  - MUTATEX
  - EDU TEST
  - CODE GENERATOR

- Seed models
- WODEL editor
- WODEL program
- Domain meta-model
- Equivalent detection
- Alternative textual representations

- «checks»
- «conforms-to»
- Moodle
- Web
- Mobile app
ALTERNATIVE RESPONSE

Seed model: \( 'a*b*' \)

Mutation operators

Mutant model

Exercise 1

Exercise 3

Mutant model

Accepts only 'a*b*'?

Accepts only 'a*b*'?
MULTIPLE DIAGRAM CHOICE

Which of these only accepts ‘a*b’?

exercise 1

seed model: ‘a*b’

mutation operators

mutant model

mutant model

mutant model

Which of these only accepts ‘a*b’?
MULTIPLE EMENDATION CHOICE

exercise 1

How to correct this DFA to accept only \'(ab)^*ba'\?

- Change State q3 to final
- Change State q0 to non-final
- Change Transition b from State q3 to State q2 with the new target State q4
- Change Transition a from State q2 to State q4 with the new target State q2
MULTIPLE EMENDATION CHOICE

How to correct this DFA to accept only \((ab)^*ba\)?

- \(\times\) Change State q3 to final
- \(\checkmark\) Change State q0 to non-final
- \(\checkmark\) Change Transition b from State q3 to State q2 with the new target State q0
- \(\times\) Change Transition a from State q2 to State q4 with the new target State q2

exercise 1

mutant model-1

seed model: \((ab)^*ba\)

mutation operators-1

mutation operators-2

mutant model-1

mutant model-2
MATCH PAIRS

**exercise 1**

Select the language that is produced by the following changes:

- Change Transition $a$ from State $q_0$ to State $q_1$ with the new target State $q_0$.
- Change Transition $b$ from State $q_1$ to State $q_1$ with the new target State $q_0$.
- Change Transition $b$ from State $q_0$ to State $q_1$ with the new target State $q_0$.
MISSING WORDS

seed model: 'ba* | ba*b'

exercise 1

mutant model

Modify the above automaton to accept only 'ba* | ba*b':

- Change Transition $\downarrow a$ from State $\downarrow q0$ to State $\downarrow q3$ with the new target State $\downarrow q2$

- Change State $\downarrow q2$ to $\downarrow$ non-final