

FSML++ + Testing

Carsten Hartenfels Benjamin Haßel

2014-02-26

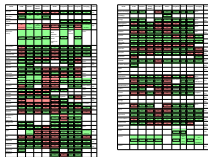
Contents

1. Frameworks
2. What we do
3. TestDataGeneration
4. Run

1. Frameworks

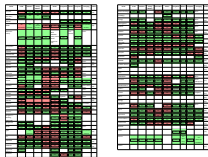
1. Frameworks

- Many available...



1. Frameworks

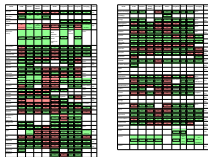
- ▶ Many available...



- ▶ ...but unsuitable to our approach

1. Frameworks

- ▶ Many available...



- ▶ ...but unsuitable to our approach
- ▶ Simple algorithm

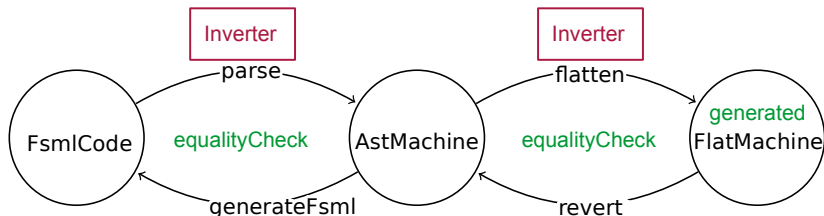
2. What we do

2. What we do

- ▶ Identity Testing

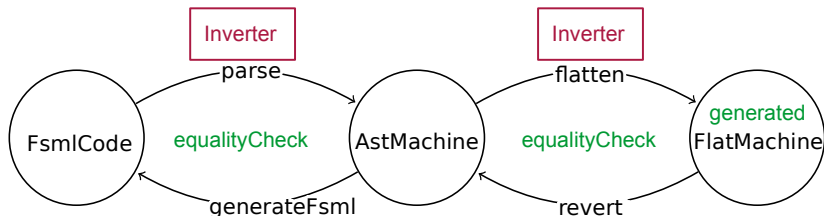
2. What we do

- ▶ Identity Testing
- ▶ Testing parser, abstract syntax and flat representation



2. What we do

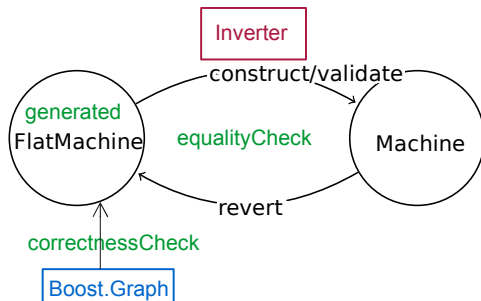
- ▶ Identity Testing
- ▶ Testing parser, abstract syntax and flat representation



- ▶ BUT not checking any constraints

2. What we do

- Solution: Oracle



3. TestDataGeneration

► Algorithm (2 states / 25 configurations)

Goedel Number	Transition			
0				
1	$s0 \rightarrow s0$			
2	$s1 \rightarrow s0$			
3	$s0 \rightarrow s1$			
4	$s1 \rightarrow s1$			
5	$s0 \rightarrow s0$	$s0 \rightarrow s0$		
6	$s1 \rightarrow s0$	$s0 \rightarrow s0$		
7	$s0 \rightarrow s1$	$s0 \rightarrow s0$		
8	$s1 \rightarrow s1$	$s0 \rightarrow s0$		
9	$s0 \rightarrow s0$	$s1 \rightarrow s0$		
10	$s1 \rightarrow s0$	$s1 \rightarrow s0$		
11	$s0 \rightarrow s1$	$s1 \rightarrow s0$		
12	$s1 \rightarrow s1$	$s1 \rightarrow s0$		
13	$s0 \rightarrow s0$	$s0 \rightarrow s1$		
14	$s1 \rightarrow s0$	$s0 \rightarrow s1$		
15	$s0 \rightarrow s1$	$s0 \rightarrow s1$		
16	$s1 \rightarrow s1$	$s0 \rightarrow s1$		
17	$s0 \rightarrow s0$	$s1 \rightarrow s1$		
18	$s1 \rightarrow s0$	$s1 \rightarrow s1$		
19	$s0 \rightarrow s1$	$s1 \rightarrow s1$		
20	$s1 \rightarrow s1$	$s1 \rightarrow s1$		
21	$s0 \rightarrow s0$	$s0 \rightarrow s0$	$s0 \rightarrow s0$	
22	$s1 \rightarrow s0$	$s0 \rightarrow s0$	$s0 \rightarrow s0$	
23	$s0 \rightarrow s1$	$s0 \rightarrow s0$	$s0 \rightarrow s0$	
24	$s1 \rightarrow s1$	$s0 \rightarrow s0$	$s0 \rightarrow s0$	
25	$s0 \rightarrow s0$	$s1 \rightarrow s0$	$s0 \rightarrow s0$	

3. TestDataGeneration

► State&Step-Generation

```
static FlatStep
genStep(const cpp_int& pos, const cpp_int& num, const cpp_int& states)
{
    const cpp_int source = (pos / num) % states;
    const cpp_int target = (pos / (num * states)) % states;
    return {"s" + source.str(), "input" + pos.str(), "", "s" + target.str()};
}

const cpp_int states = stateCount;
const cpp_int pow = states * states;
for (cpp_int off = 1, no = pow, num = 1; off <= t; off += no, no *= pow, num *= pow)
    fm.addStep(genStep(t - off, num, states));

return fm;
```

Thank You All For Listening

GitHub: <https://github.com/hartenfels/FSMLplusplus/>