

CIS 842: Specification and Verification of Reactive Systems

Lecture INTRO-Bogor-Simulation: Executing (Simulating) Concurrent Systems in Bogor

Copyright 2004, Matt Dwyer, John Hatcliff, and Robby. The syllabus and all lectures for this course are copyrighted materials and may not be used in other course settings outside of Kansas State University in their current form or modified form without the express written permission of one of the copyright holders. During this course, students are prohibited from selling notes to or being paid for taking notes by any person or commercial firm without the express written permission of one of the copyright holders.

Objectives

- Understand how a model-checker's random simulation and user-guided simulation mode can be applied to explore paths through a system's computation tree.
- Be able to apply Bogor in both random simulation mode and user-guided simulation mode.

Outline

- Bogor random simulation mode
- Bogor user-guided simulation mode

SumToN

```
system SumToN {
  const PARAM { N = 3 };
  typealias byte int wrap (0,255);

  byte x;
  byte t1;
  byte t2;

  thread Thread10 {
    loc loc0:
      when x != 0 do { t1 := x; }
      goto loc1;

    loc loc1:
      do { t2 := x; }
      goto loc2;

    loc loc2:
      do { x := t1 + t2; }
      goto loc0;
  }
}
```

```
thread Thread20 {
  loc loc0:
    when x != 0 do { t1 := x; }
    goto loc1;

  loc loc1:
    do { t2 := x; }
    goto loc2;

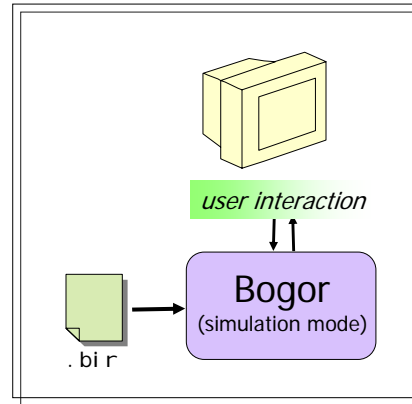
  loc loc2:
    do { x := t1 + t2; }
    goto loc0;
}

thread Thread00 {
  loc loc0:
    do {
      x := 1;
    } goto loc1;

  loc loc1:
    do { assert (x != PARAM.N); }
    return;
}
}
```

Assessment

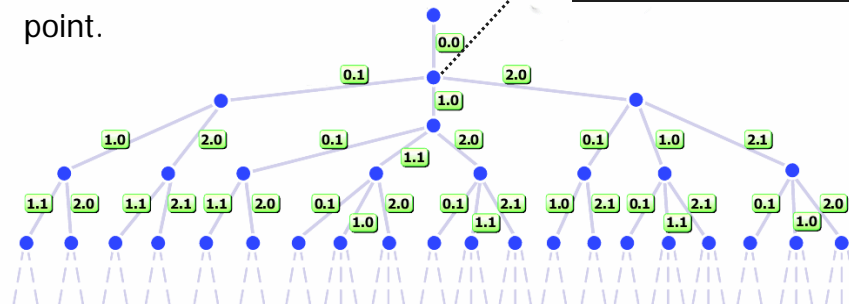
- Even though this is a very small system, it is already tedious for us to try to find a violating trace.
- Bogor can act as a *simulator* to help us find a violating trace.
- Bogor simulates in two ways:
 - random simulation
 - user-guided simulation
- Same as many other model-checkers (e.g., Spin)



Random Simulation

- In a random simulation, Bogor randomly chooses a branch at a choice point.

At choice points, Bogor randomly chooses one of the enabled transitions.



Assessment

- Bogor in random simulation mode...
 - sometimes it can find an error
 - e.g., when assertion read $x \neq 1$
 - most of the time it won't find an error
 - e.g., when assertion read $x \neq 3$
 - in this case, Bogor runs forever, and you can notice the overflow error messages associated with the variables of type byte

Assessment

- Bogor in guided simulation mode...
 - the user can guide Bogor to the error
 - but this requires that the user already know or at least have a good idea about how the error can occur!
 - tedious and error prone
 - infeasible on all but very short traces
 - cannot be used in practice to obtain an exhaustive search of all possible traces
 - can be useful in practice if the user simply wants to explore the behavior, e.g., of a particularly troublesome section of code

On To Exhaustive Exploration...

- Bogor's random simulation
 - isn't that useful for finding bugs
 - only explores one execution trace
- Bogor's guided simulation
 - is only useful on short traces where the user already has a good idea of how a property violation might arise
 - only feasible to explore a few execution traces
- The main strength of Bogor is its automatic exhaustive search capabilities
 - this is why people use model-checkers!
 - this is what this course is all about