Model Checking

...

Yegor Bugayenko

Lecture #9 out of 10
90 minutes

All videos are in this YouTube playlist.

All visual and text materials presented in this slidedeck are either originally made by the author or taken from public Internet sources, such as website. Copyright belongs to their respected authors.

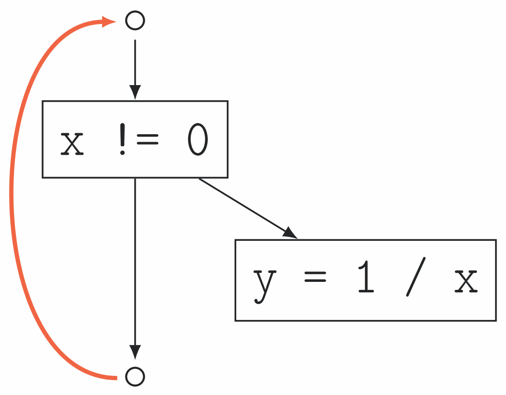
Chapter #1:

Motivating Example

Div by Zero

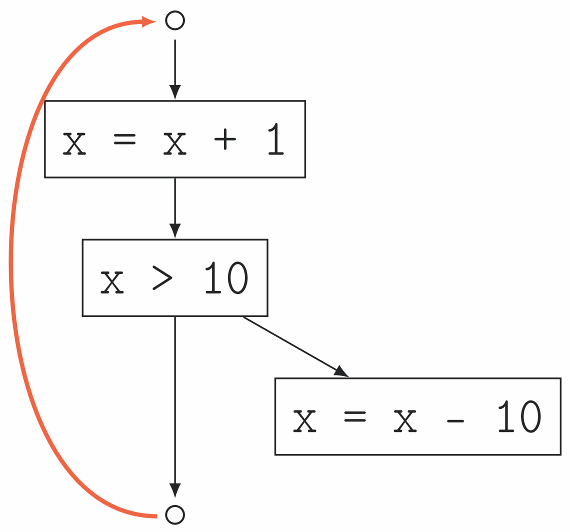
```
// Process no. 1:

extern int x;
extern double y;
int measure() {
  if (x != 0) {
    y = 1.0 / x;
  }
}
```



```
// Process no. 2:

extern int x;
void roll() {
  x += 1;
  if (x > 10) {
    x -= 10;
  }
}
```



Can we detect “division by zero” using symbolic execution? Is “division by zero” the only error here?

ProMeLa (Process Meta Language)

```
extern int x;
extern double y;
int measure() {
  if (x != 0) {
    y = 1.0 / x;
  }
}
void roll() {
  x += 1;
  if (x > 10) {
    x -= 10;
  }
}
```

```
int x; bool dbz;
active proctype measure() {
  do :: true ->
    if
    :: (x != 0) -> dbz = (x == 0)
    :: skip
    fi
  od
}
active proctype roll() {
  do :: true ->
    x = x + 1;
    if
    :: x > 10 -> x = x - 10
    :: skip
    fi
  od
}
```

SPIN (Simple ProMeLa Interpreter)

```
int x; bool dbz;
active proctype measure() {
  do :: true ->
    if
    :: (x != 0) -> dbz = (x == 0)
    :: skip
    fi
  od
}
active proctype roll() {
  do :: true ->
    x = x + 1;
    if
    :: x > 10 -> x = x - 10
    :: skip
    fi;
    printf("x = %d\n", x);
  od
}
```

```
$ spin main.pml | head
          x = 1
          x = 2
          x = 3
          x = 4
          x = 5
          x = 6
          x = 7
          x = 8
          x = 9
          x = 10
$ spin main.pml | tail
...
```

Just checkout this repo and run `make`, the `spin` binary will be compiled.

Monitoring Process

```
int x; bool dbz;
active proctype measure() {
  do :: true ->
    if
    :: (x != 0) -> dbz = (x == 0)
    :: skip
    fi
  od
}
active[2] proctype roll() {
  do :: true ->
    x = x + 1;
    if
    :: x > 10 -> x = x - 10
    :: skip
    fi
  od
}
```

```
active proctype monitor() {
  do :: true ->
    assert(!dbz);
    assert(x >= 0);
  od
}
```

Pay attention to the `[2]` suffix after the `active` keyword. It tells SPIN to start two instances of the `roll` process.

Model-less Checking

Fail on Assertion

```
int x; bool dbz;
active proctype measure() {
  do :: true ->
    if
    :: (x != 0) -> dbz = (x == 0)
    :: skip
    fi
  od }
active[2] proctype roll() {
  do :: true ->
    x = x + 1;
    if
    :: x > 10 -> x = x - 10
    :: skip
    fi
  od }
active proctype monitor() {
  do :: true -> assert(!dbz); assert(x >= 0); od
}
```

```
$ spin main.pml
spin: main.pml:22, Error: assertion violated
spin: text of failed assertion: assert((x>=0))
#processes: 4
    x = -9
    dbz = 0
584:  proc  3 (monitor:1) main.pml:22 (state 3)
584:  proc  2 (roll:1) main.pml:17 (state 7)
584:  proc  1 (roll:1) main.pml:18 (state 9)
584:  proc  0 (measure:1) main.pml:9 (state 8)
4 processes created
```

Chapter #2:

Model-less Checking

Java PathFinder

Further Reading/Watching

Introduction lecture by Joost-Pieter Katoen

A Primer on Model Checking by Mordechai Ben-Ari