Detecting Likely Maintained Invariants

Let’s say we need to modify an unfamiliar program...
We could make changes more safely if we knew the pre- and post- conditions each function maintains, but they are seldom documented...

Dynamic invariant detectors, such as Daikon, produce this kind of data.

A dynamic invariant detector instruments a target program to capture the values in variables visible at interesting program points. After execution, it creates, for selected program points, a set of relationships (potential invariants) that hold on all observations.

87 useful invariants generated/maintained by these 3 lines? Seems unlikely...
bittenBy can’t possibly maintain most of these invariants. They involve variables that are independent on every execution path. Some aren’t even read or written by the method.

What if we tighten up the definition of a likely invariant to include a predicate only if it relates dependent variables, including one that is modified by the method. Dependency includes control and data dependency through any number of intermediate variables.

public void bittenBy(Zombie z) {  
  
  Alien bittenBy(Y.Zombie z) : EXT42 -- this.cump == this.cump 
  Alien bittenBy(Y.Zombie z) : EXT42 -- this.cump -- this.cump == 0 
  Alien bittenBy(Y.Zombie z) : EXT42 -- this.nameToZombie == "THX322" 
  Alien bittenBy(Y.Zombie z) : EXT42 -- this.max = 50 
  Alien bittenBy(Y.Zombie z) : EXT42 -- this.max == 0 
  Alien bittenBy(Y.Zombie z) : EXT42 -- 0 <= this.max <= 63 
  Alien bittenBy(Y.Zombie z) : EXT42 -- this.turnsToZombie == 0 
  Alien bittenBy(Y.Zombie z) : EXT42 -- this.turnsToZombie == 0 
  Alien bittenBy(Y.Zombie z) : EXT42 -- 0 != this.turnsToZombie <= 63 
  Alien bittenBy(Y.Zombie z) : EXT42 -- Z重点工作 == boolean 
  Alien bittenBy(Y.Zombie z) : EXT42 -- this.cump <= this.cump 
  Alien bittenBy(Y.Zombie z) : EXT42 -- this.cump >= 60 
  Alien bittenBy(Y.Zombie z) : EXT42 -- 0 <= this.turnsToZombie <= 63 
  Alien bittenBy(Y.Zombie z) : EXT42 -- Zombie and has only one value 
  Alien bittenBy(Y.Zombie z) : EXT42 -- Zombie and = null 
  Alien bittenBy(Y.Zombie z) : EXT42 -- arg0.name.toString == "B00" 
  Alien bittenBy(Y.Zombie z) : EXT42 -- arg0.max = 25 

After running Daikon, we have output like the following...

16,899 invariants! There are only 850 lines of program text including whitespace and comments in the original source. Are there really 20 invariants maintained for each line of source code?
The 3 line bittenBy method is reporting 87 invariants...

Filtering on dependence reduces the 87 invariants to 5.

---

Special thanks to the University of Oregon Computer Science Department for funding this work during Summer 2012.

---

Daniel Elsworth and Michal Young

dellorw,michial@cs.uoregon.edu