Destiny Support

Robert Krug

Department of Computer Science
University of Texas at Austin

March 1st, 2006

Outline

1. What is Destiny
   - Destiny — an Overview
   - Input
   - The Rulebase
   - Output

2. Destiny Support
   - The Destiny Model
   - Guessing Theorems

Destiny — an Overview

- Input: Java byte code and source
- The rule base
- Output: XML to be parsed by ACL2 (or PVS, HOL, . . .)

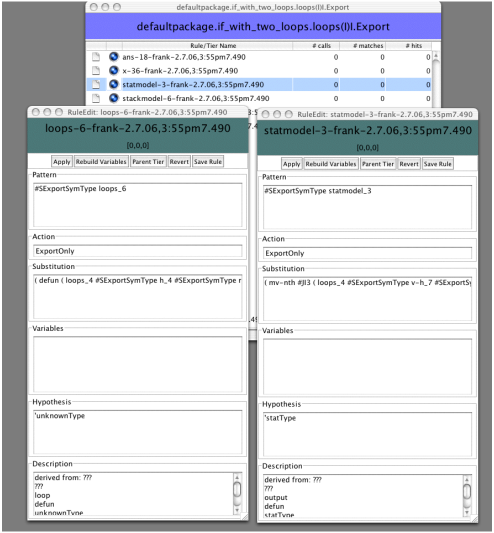
Destiny Input — Java

```
public class if_with_two_loops {

    public static boolean main (int x) {
        int i = loops (x);
        return i == x;}

    static int loops (int x) {
        int ans = 0;
        if (x < 10)
            for (int i=0; i<x; i++) {
                ans++;}
        else
            for (int i=1; i<=x; i++) {
                ans++;}
```

The Rulebase



defaultpackage.if_with_two_loops.loops(I)I.Export

Rule/Tier Name	# calls	# matches	# hits
ans-18-frank-2.7.06,3:55pm7.490	0	0	0
x-36-frank-2.7.06,3:55pm7.490	0	0	0
statmodel-3-frank-2.7.06,3:55pm7.490	0	0	0
stackmodel-6-frank-2.7.06,3:55pm7.490	0	0	0

RuleEdit: loops-6-frank-2.7.06,3:55pm7.490

loops-6-frank-2.7.06,3:55pm7.490

[0,0,0]

Pattern
#SExportSymType loops_6

Action
ExportOnly

Substitution
(defun (loops_4 #SExportSymType h_4 #SExportSymType r

Variables

Hypothesis
'unknownType

Description
derived from: ???
???
loop
defun
unknownType

RuleEdit: statmodel-3-frank-2.7.06,3:55pm7.490

statmodel-3-frank-2.7.06,3:55pm7.490

[0,0,0]

Pattern
#SExportSymType statmodel_3

Action
ExportOnly

Substitution
(mv-nth #JI3 (loops_4 #SExportSymType v-h_7 #SExportSy

Variables

Hypothesis
'statType

Description
derived from: ???
???
output
defun
statType

The Rulebase

- Loop definitions
- Axioms
- Conjectures

Destiny Output — a Parsed Loop

```
(defun loops_11 (h nh stack stat x ans i)
  (declare (xargs :measure (nfix (1+ (- x i)))))
  (if (and (integerp i)
           (integerp ans)
           (integerp x)
           (static-area-p stat)
           (stack-p stack)
           (heap-counter-p nh)
           (heap-p h))
      (if (< i x)
          (loops_11 h nh
                    (pushframe
                     (storecat1
                      (+ i 1)
                      2
                      (storecat1 (+ ans 1) 1
                                 (storecat1 x 0
                                            (pushcat1 (+ i 1)
                                                      (popop (getframe stack))))))
                     (popframe stack))
                    stat x (+ ans 1)
                    (+ i 1))
        (mv h nh stack stat x ans i))
    (mv h nh stack stat x ans i)))
```

Destiny Output — a Parsed Conjecture

```
(defthm LOOPS-11-TERMINATES-NORMALLY-FRANK-2.7.06\,3\:55PM7.490
  (implies (and (heap-p v-h_7)
                (heap-counter-p v-nh_8)
                (stack-p v-stack_8)
                (static-area-p v-stat_8)
                (integerp x_28)
                (< x_28 10))
           (let ((x_30 (mv-nth 4
                              (loops_11 v-h_7 v-nh_8
                                        (pushframe
                                         (storecat1 0 2
                                                    (storecat1 0 1
                                                               (storecat1 x_28 0
                                                                          (pushcat1 0 (getfran
                                         (popframe v-stack_8))
                                        v-stat_8 x_28 0 0)))
                (i_5 (mv-nth 6
                             (loops_11 v-h_7 v-nh_8
                                       (pushframe
                                        (storecat1 0 2
                                                   (storecat1 0 1
                                                              (storecat1 x_28 0
                                                                         (pushcat1 0 (getframe
                                        (popframe v-stack_8))
                                       v-stat_8 x_28 0 0))))
             (<= x_30 i_5))))
```

Destiny Output — a Parsed Conjecture

```
(defthm HEAP-IS-INVARIANT-AT-2.7.06\,5\:37PM50.566-FRANK-2.7.06\,3\:55PM7.490
  (implies (and (heap-p v-h)
                (heap-counter-p v-nh)
                (stack-p v-stack)
                (static-area-p v-stat)
                (integerp x)
                (< x 10))
           (let ((heapmodel (mv-nth
                             0
                             (loops_11
                              v-h v-nh
                              (pushframe
                               (storecat1
                                0 2
                                (storecat1 0 1
                                           (storecat1 x 0 (pushcat1 0 (getframe v-stack)))))
                               (popframe v-stack))
                              v-stat x 0 0))))
             (equal heapmodel v-h))))
```

Destiny Output — a Parsed Conjecture

```
(defthm CALLING-STACK-IS-INVARIANT-AT-2.7.06\,5\:39PM20.770-FRANK-2.7.06\,3\:55PM7.490
  (implies (and (heap-p v-h)
                (heap-counter-p v-nh)
                (stack-p v-stack)
                (static-area-p v-stat)
                (integerp x)
                (< x 10))
           (let ((stackmodel (mv-nth
                              2
                              (loops_11
                               v-h v-nh
                                (pushframe
                                 (storecat1
                                  0 2
                                  (storecat1 0 1
                                             (storecat1 x 0 (pushcat1 0 (getframe v-stack)))))
                                 (popframe v-stack))
                               v-stat x 0 0))))
             (equal (popframe stackmodel)
                    (popframe v-stack)))))
```

Destiny Output — a Parsed Conjecture

```
(defthm ANS-=-X--ANS-22-FROM-LOOPS-4-FRANK-2.7.06\,6\:59PM2.32
  (implies (and (heap-p v-h) (heap-counter-p v-nh)
                (stack-p v-stack) (static-area-p v-stat)
                (integerp x) (<= 10 x) (<= 0 x))
           (let ((ans (mv-nth 5
                              (loops_4 v-h v-nh
                                       (pushframe
                                        (storecat1 1 2
                                                   (storecat1 0 1
                                                              (storecat1 x 0
                                                                         (pushcat1 1
                                                                                   <12 lines
                                        (popframe
                                         (pushframe
                                          (storecat1 (getlocal 0 (getframe v-stack))
                                                     0 (pushcat1 42 (getframe v-stack)))
                                          (pushframe
                                           (storecat1 (getlocal 1 (getframe v-stack))
                                                      1
                                                      (storecat1 (getlocal 0 (getframe v-stac
                                                                 0 (getframe v-stack)))
                                           (popframe v-stack)))))
                                       v-stat x 0 1))))
             (= ans x))))
```

The Destiny Model

```
(encapsulate ((array-p (name type arity) t)
              (heap-p (x) t)
              (heap-counter-p (x) t)
              (stack-p (x) t)
              (static-area-p (x) t)
              (unknown-type-p (x) t)

              (refh (name subaddress) t)
              (valueh (ref heap) t)
              (pushh (ref value heap) t)
              (pushFrame (frame stack) t)
              (popFrame (stack) t)
              (getFrame (stack) t)
              (storeCat1 (var value frame) t)
              (storeCat2 (var value frame) t)
              (getLocal (offset frame) t)
              (pushCat1 (value frame) t)
              (pushCat2 (value frame) t)
              (popop (frame) t)

              (frame-p (x) t)
              (ref-p (x) t))
```

The Destiny Model

```
(local
 (defun stack-p (x)
   (or (equal x 'dummy-stack)
       (and (consp x)
            (true-listp x)))))

(defthm stack-p-stack
  (stack-p 'dummy-stack))

(defthm stack-p-pushFrame
  (implies (and (frame-p frame)
                (stack-p stack))
           (stack-p (pushFrame frame stack))))

(defthm popFrame-pushFrame
  (implies (and (frame-p frame)
                (stack-p stack))
           (equal (popFrame (pushFrame frame stack))
                  stack)))
```

Guessing Theorems

- Invariance
- Stack Related
- Cone of Influence
- Other

Invariance

```
(defthm loops_11-0-invariant
   (equal (car (loops_11 h nh stack stat x ans i))
          h))

(defthm loops_11-1-invariant
   (equal (mv-nth 1 (loops_11 h nh stack stat x ans i))
          nh))

(defthm loops_11-3-invariant
   (equal (mv-nth 3 (loops_11 h nh stack stat x ans i))
          stat))

(defthm loops_11-4-invariant
   (equal (mv-nth 4 (loops_11 h nh stack stat x ans i))
          x))
```

Stack Related

```
(encapsulate ()
 (local
  (defthm loops_11-stack-is-irrelevant-to-mv-nth-6-helper
       (implies (and (stack-p stack) (stack-p stack-2))
                (equal (mv-nth 6 (loops_11 h nh stack stat x ans i))
                       (mv-nth 6 (loops_11 h nh stack-2 stat x ans i))))))
 (local
  (in-theory (disable loops_11-stack-is-irrelevant-to-mv-nth-6-helper)))

 (defthm loops_11-stack-is-irrelevant-to-mv-nth-6
     (implies (and (stack-p stack)
                   (syntaxp (not (equal stack ''dummy-stack))))
              (equal (mv-nth 6 (loops_11 h nh stack stat x ans i))
                     (mv-nth 6 (loops_11 h nh 'dummy-stack
                                         stat x ans i))))
     :hints (("GOAL" :use
              (:instance loops_11-stack-is-irrelevant-to-mv-nth-6-helper
                         (stack-2 'dummy-stack)))))
)
```

Cone of Influence

```
(defun loops_11-ind-fn (h nh stack stat x ans i)
  (declare (xargs :measure (nfix (1+ (- x i)))))
  (if (and (integerp i)
           (integerp ans)
           (integerp x)
           (static-area-p stat)
           (stack-p stack)
           (heap-counter-p nh)
           (heap-p h))
      (if (< i x)
          (loops_11-ind-fn h nh stack stat x
                           (+ ans 1)
                           (+ i 1))
        (mv h nh stack stat x ans i))
    (mv h nh stack stat x ans i)))
```

Cone of Influence

```
(defthm loops_11-ind-thm
  (equal x x)
  :rule-classes
  ((:induction :pattern (loops_11 h nh stack
                                  stat x ans i)
               :condition t
               :scheme (loops_11-ind-fn h nh stack
                                        stat x ans i))))
```

Cone of Influence

```
(defthm loops_11-mv-nth-5-is-irrelevant-to-mv-nth-6
  (implies (and (syntaxp (destiny-rewriting-goal-literal mfc state))
                (bind-free
                 (destiny-bind-irrelevant-var 'loops_11
                                              5 6 (list h nh stack stat x ans1 i)
                                              'ans2
                                              mfc state)
                 (ans2))
                (syntaxp (not (equal ans1 ans2)))
                (integerp ans1)
                (integerp ans2))
           (equal (mv-nth 6 (loops_11 h nh stack stat x ans1 i))
                  (mv-nth 6 (loops_11 h nh stack stat x ans2 i)))))
```

Other

```
(defthm clear_4-array-length-invariant
   (equal (valueh (refh x ’array_length_marker)
                  (car (clear_4 h nh stack stat x i)))
          (valueh (refh x ’array_length_marker)
                  h)))
```