Design and Analysis of Real-Time Systems SS 2013

Jan Reineke
Andreas Abel

Deadline: Thursday, June 6, 2013, 14:15

Assignment 5

Problem 1: Short Questions (3+3+3+3 Points)

The goal of the following questions is to test your intuitive understanding of some important concepts discussed in the course so far.

1. The abstraction and concretization functions α and γ should be monotone. Explain why that makes sense.
2. Explain the local consistency condition.
3. On slide 53 of the lecture on May 2^{nd} we defined abstract expression evaluation. Why is this sound? Can you find an example that shows that it is not necessarily the most precise way to evaluate expressions?
4. Why is the collecting semantics defined as a least fixed point and not, e.g., the greatest fixed point? Can you find a program for which the collecting semantics has several fixed points?

Problem 2: Fixed Points (2+2 Points)

Prove or disprove the following claims.

1. If x is a fixed point of a function f, then x is also a fixed point of $f^2(x)$.
2. If x is a fixed point of f^n, then x is also a fixed point of f^{n+1}.

Problem 3: Sign/Parity Analysis (2+2+2 Points)

Consider the following program:

```
if (a>0) then (
    x := 0;
) else (
    x := 4;
)
x := x+2;
a := 3/x;
```

1. Is it possible to detect that no division by zero can occur in the last statement using *Parity Analysis*/*Simple Sign Analysis*/*Extended Sign Analysis*?
2. Modify the program such that only the *Parity Analysis* is able to show that no division by zero can occur.
3. Modify the program such that both the *Parity Analysis* and the *Extended Sign Analysis* can be used to show that no division by zero can occur, but not the *Simple Sign Analysis*.

Problem 4: Condition Evaluation (8 Points)

The analyses you implemented on the previous assignment sheet were not able to propagate information from the conditions of *if* and *while statements* to their corresponding bodies. Consider for example the following program:

```
a := 2;
if (x<0) then (
    a := -x;
)
b := 5/a;
```

Our previous analyses were not able to detect that `a` is positive at the end of the program.

In this exercise, we consider an analysis that can handle conditions of the form *(Variable op Expression)*, where $op \in \{=, <>, <, <=, >, >=\}$.

1. Implement this extension for the *Simple Sign Analysis* of the previous assignment.
2. Run your implementation on the example program. Submit your code and a screenshot of the results.

Problem 5: Build Your Own Analysis (6+4+10 Points)

Consider the following program:

```
/*  1: */  program p
/*  2: */
/*  3: */  begin
/*  4: */  while (a > 0) do (
/*  5: */      a := a-5;
/*  6: */  )
/*  7: */  y := -27;
/*  8: */  y := y+34;
/*  9: */  y := y-7;
/* 10: */  if (a<-2) then (
/* 11: */      x: = 2;
/* 12: */  ) else (
/* 13: */      x := y*a;
/* 14: */      x := x+5;
/* 15: */  )
/* 16: */  while (a<5) (
/* 17: */      x := x+1;
/* 18: */      a := a+1;
/* 19: */  )
/* 20: */  a := 5/x;
/* 21: */
/* 22: */  end
```

The goal of this problem is to design an analysis that can detect whether a division by zero can occur in the last statement.

1. Find a suitable complete lattice that satisfies the ascending chain condition, and draw a Hasse diagram for your lattice.
2. Define an abstraction function α, and a concretization function γ.
3. Implement your analysis in PAG/WWW and run it on the example program. Submit your code and a screenshot of the results.