Relational Summaries for Interprocedural Analysis

Remy Boutonnet (UGA)

VERIMAG - PACSS

June 4th, 2018

Program analysis and abstract interpretation

Most interesting properties in program analysis are undecidable.

Abstract Interpretation gives safe approximate answers to undecidable questions.

Linear Relation Analysis

```
assume n >= 0;
i := 0;
-- 1: i = 0 and n >= 0

-- 2: i >= 0 and i <= n
while i < n
-- 3: i >= 0 and i <= n-1
    i := i + 1;
-- 4: i >= 1 and i <= n
end;
-- 5: i = n and n >= 0
```

→ Discovers automatically systems of linear equalities and inequalities. Powerful relational analysis but expensive.

Motivation

Improving the scalability of linear relation analysis on large programs with procedures, objects or synchronous modules.

Interprocedural analysis has a long story.

Disjunctive relational summaries

A modular interprocedural analysis to improve the scalability of Linear Relation Analysis.

Applied to LRA, but based on a general framework called **disjunctive relational abstract interpretation**.

Principle: computing disjunctions of abstract input-output relations.

$$\sigma_p = \{P_1(X_0, X), ..., P_n(X_0, X)\}$$

Disjunctive relational summaries

Automatic refinement of procedure summaries according to local reachability and summaries of called procedures.

Improvements of summary computation: widening limited by precondition, loop-exit refinement.

Example: the div procedure

```
procedure div (a, b, q, r)
begin
    assert(a >= 0 && b >= 1);
    0:
        q := 0;
        r := a;
    1:
    2:  while r >= b
    3:
            r := r - b;
            q := q + 1;
    4:
        end;
    5:
end
```

The summary of div is $\sigma_{div} = \{R_1, R_2\}$ such that:

$$
\begin{aligned}
R_1 \quad = \quad & (a_0 \geq b_0 \wedge b_0 \geq 1 \wedge r \geq 0 \\
& \wedge\ q \geq 1 \wedge q + r \geq 1 \\
& \wedge\ b \geq r + 1 \\
& \wedge\ a + 1 \geq b + q + r \\
& \wedge\ a = a_0 \wedge b = b_0)
\end{aligned}
$$

$$
\begin{aligned}
R_2 \quad = \quad & (a_0 < b_0 \wedge a_0 \geq 0 \\
& \wedge\ q = 0 \wedge r = a \\
& \wedge\ a = a_0 \wedge b = b_0)
\end{aligned}
$$

Summaries of recursive procedures

```
procedure f91 (x, y)
begin
    z, t : int;
    if x > 100 then
        y := x-10;
    else
        z := x+11;
        f91(z,t);
        f91(t,y);
    end;
end
```

The summary of McCarthy's 91 function is such that:

$$\begin{aligned} R_1 &= (x \leq 89 \wedge y = 91) \\ R_2 &= (90 \leq x \leq 100 \wedge y = 91) \\ R_3 &= (x \geq 101 \wedge y = x - 10) \end{aligned}$$

Summaries of synchronous modules and objects

Synchronous modules are implemented by step procedures with memory remanent between invocations.

Objects have an internal state (*attributes*), possibly modified by methods calls.

$\rightarrow$ Summaries of procedures with remanent memory.