Development of Safety-Critical Embedded Systems
PAG/WWW

Jörg Herter

2013/01/08

What's PAG ...?

... a YACC for program analyses?

- PAG supports instances of monotone frameworks
- Input: concise specifications (of program analyses)
- Output: ANSI C code (of a program analyzer)
- Advantages:
  1. rapid implementation
  2. integrated debugging facilities
  3. short specification

... and PAG/WWW?

A web interface to PAG.

`http://www.program-analysis.com`

Main differences to PAG:

- Restricted features
- Simplified specification language
- Fixed input language

  `WHILE`: `http://www.program-analysis.com/while.html`

PAG/WWW vs. PAG

PAG/WWW

1. for WHILE programs only
2. restricted specification languages
3. restricted syntax
4. intended for educational purposes
5. restricted to a certain iteration algorithm

PAG

1. full system not bound to a specific language
2. additional specification features
3. more complicated syntax
4. intended for industrial and for research purposes
5. different interprocedural iteration algorithms

PAG/WWW Demo "Reaching Definitions"

Reaching definitions

"Which variable definitions/assignments reach which program points."

Want to compute for each program point *u* the set:
$\{(v, p) | v \in \mathcal{VARS} \wedge p \in \mathcal{LOCS} \wedge$ definition of v at p reaches u$\}$

Reaching definitions

Lattice (where the analysis results are from):

$$\{(v, p) | v \in \mathcal{VARS} \wedge p \in \mathcal{LOCS}\} = \mathcal{VARS} \times \mathcal{LOCS}$$

Least upper bound (how we combine information from diff. paths):

$$\cup$$

Edge effects (how to adjust information traversing a cfg-edge):

???

PAG/WWW Demo "Reaching Definitions"

Reaching definitions (textual results)

`http://www.program-analysis.com`

PAG/WWW Demo "Reaching Definitions"

How to interpret the textual results:

- Labels program automatically
- Shows entry and exit information
- Procedure parameters are underlined

PAG/WWW Demo "Reaching Definitions"

Reaching definitions (graphical results)

`http://www.program-analysis.com`

PAG/WWW Demo "Reaching Definitions"

How to interpret the graphical results:

- A picture for each computation step
- Exit information beside outgoing edges
- Entry information not displayed
- Color legend:
  - Red: information is about to be changed
  - Blue: nodes in the worklist
- Node labels: numbered elementary statements

Analysis Specification

Different parts of an analysis specification:

1. TYPE: define the analysis lattice
2. PROBLEM: define analysis parameters
3. TRANSFER: define the transfer functions
4. SUPPORT: define additional functions

Specification in a specialized functional language FULA (ML like)

`http://www.program-analysis.com/fula_grammar.html`

Lattice Specification

Predefined datatypes:

- `snum` signed integer
- `bool` boolean
- `str` string
- `Label` program label
- `Var` program variable
- `Proc` program procedure
- `Expression` non-trivial program expression

Lattice Specification

Lattice construction:

- `set(<ld>)` Set over <ld>
- `list(<ld>)` List over <ld>; NOT a lattice!
- `<ld1> * <ld2>` Tuple space
- `<ld1> -> <ld2>` Function space
- `flat(<ld>)` Flat lattice
- `lift(<ld>)` Lifted lattice

Lattice Specification

Predefined sets:

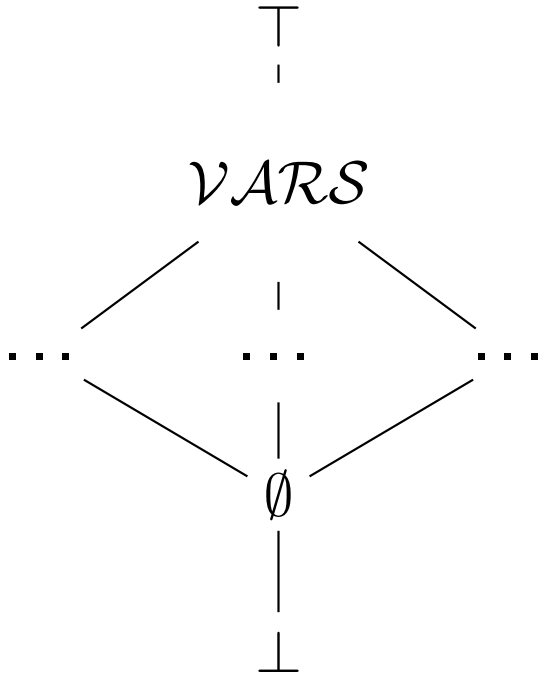
- `LabelSet` set(Label)
- `VarSet` set(Var)
- `ProcSet` set(Proc)
- `ExpressionSet` set(Expression)
- `ExpressionList` list(Expression)

Remarks: Fixed-point Iteration

1. In PAG, the worklist is initialized *only* with the start node.
2. Algorithm stops when worklist is empty, new nodes are added when a node value changes
3. The programmer must ensure that each node of interest is visited at least once.

$\Rightarrow$ Add an additional, artificial bottom element if the bottom element of the lattice has a meaningful value

Remarks: Fixed-point Iteration cont'd



Live Variables Analysis

"Which variables are live, i.e., may be read before written before program termination, at which program points."

Want to compute for each program point p the set

$$\{v | \text{v is live at p}\}$$

Lattice: $2^{\mathcal{VARS}}$, where $\mathcal{VARS}$ is the set of all program variables
Least upper bound: $\cup$; MOP: $\mathcal{L}[u]^* = \bigcup\{[\![\pi]\!]^\sharp\emptyset | \pi : u \to^* \text{end}\}$
Edge effects: ???

PROBLEM Specification

- direction forward or backward
- carrier the analysis lattice
- init the initial value
- init_start value for the extremal node
- combine combination function

PAG/WWW Demo "Live Variables"

Live variables (problem specification)

`http://www.program-analysis.com`

Transfer (functions) Specification

In the TRANSFER section,

- define the exit value in terms of the entry value(@), or vice versa for backward problems
- give a definition by cases for the WHILE statements
- optional matching of the edge type

PAG/WWW Demo "Live Variables"

Live variables (transfer section specification)

`http://www.program-analysis.com`

Support (functions) Specification

In the SUPPORT section,

- define additional functions
- where each function needs a type declaration
- and definition by cases possible

PAG/WWW Demo "Live Variables"

Live variables (support section specification and analysis demo)

`http://www.program-analysis.com`

PAG/WWW Demo

Available Expressions Analysis and Constant Propagation

`http://www.program-analysis.com`