Automatic Test Case Generation To Maximize Def-Use Coverage

Srijan R. Shetty

Professor Subhajit Roy

Indian Institute of Technology, Kanpur

srijans@cse.iitk.ac.in

April 20, 2015

Overview

- Motivation
  - Problem Statement
  - Related Work
- Preliminaries
  - SSA
  - DU Chains
  - HotPath SSA
- Automatic Test Case Generation
  - Problem Statatement
  - Formulation
  - Additional Constraints
- Future
- Insight

Section 1

Motivation

Problem Statement

For a given *function*, generate *unit test cases* which maximize *def-use* coverage.

Related Work

Tools like PEX[1], KLEE[2] can generate test suites with high code coverage.

Code Coverage

A number of metrics can be used to measure code coverage [3].

- Function Coverage
- Branch Coverage
- Condition Coverage
- Path Coverage

Comparision with Existing Work

Def-Use Coverage[1] like Path Coverage cannot be quantified directly and have not been extensively studied.

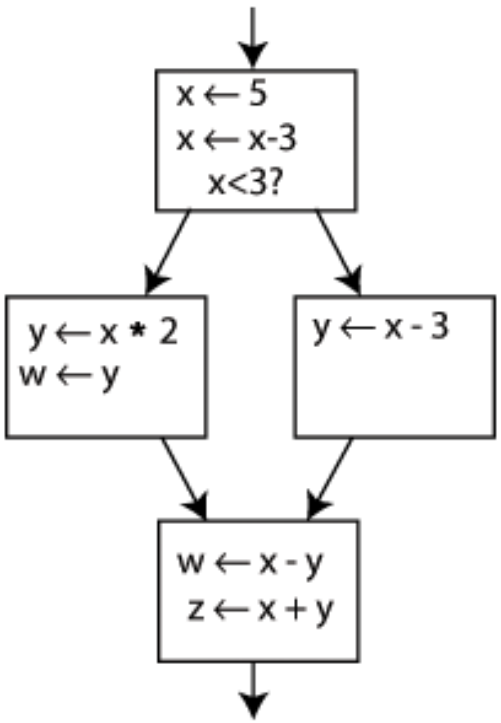
[1] defined later

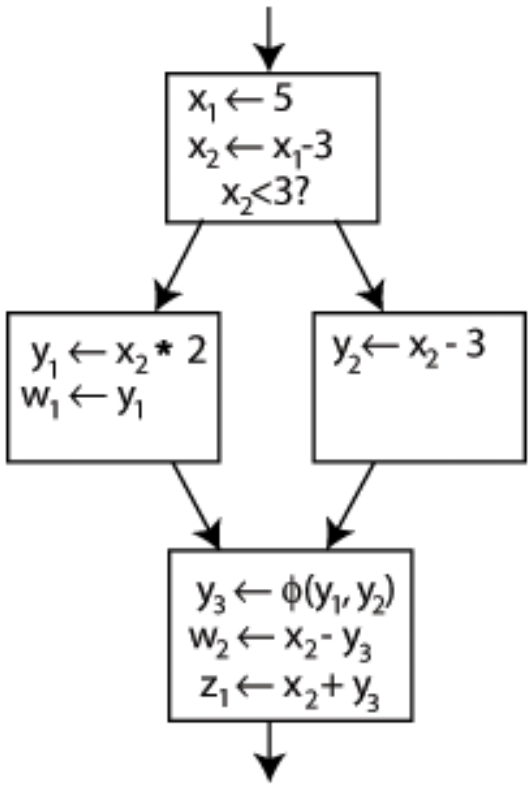
Section 2

Preliminaries

SSA

Every variable has a single definition.[2]

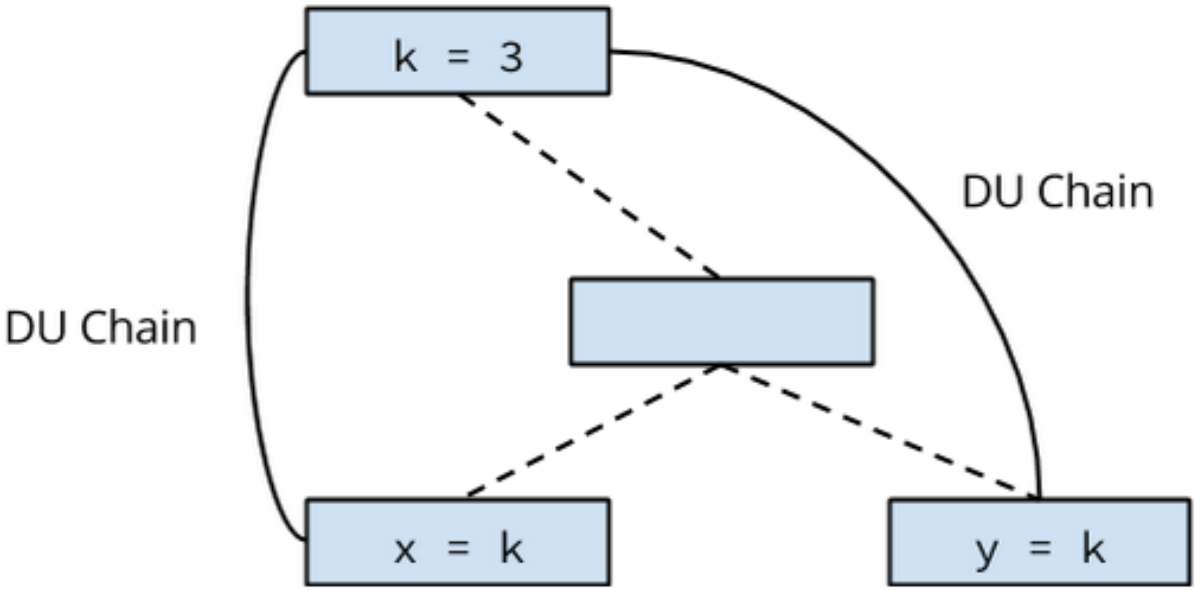




[2]Images from Wikipedia

DU Chains

Each definition of a variable is linked with its usage in a SSA graph.

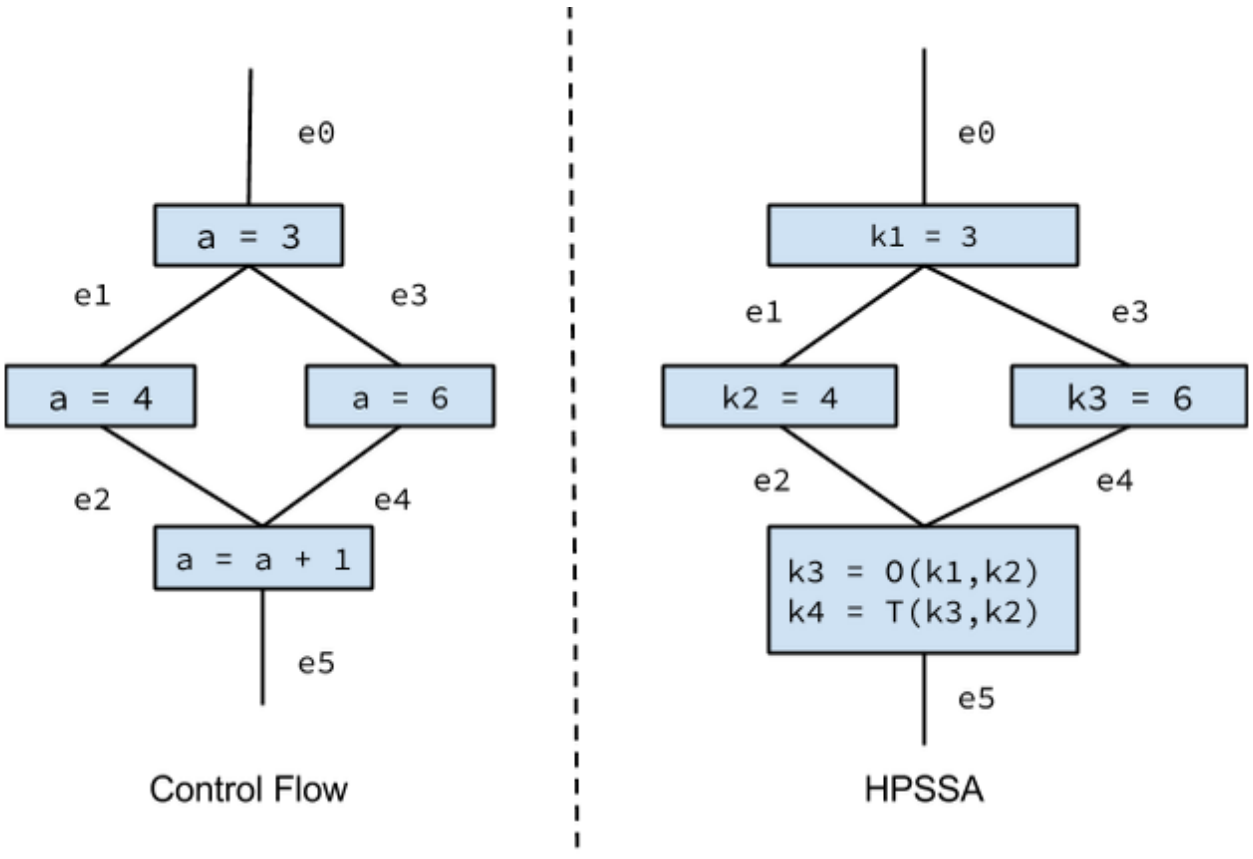


HotPath SSA

HotPath SSA [4] merges dataflow information with the control flow graph and enables a variety of optimizations.

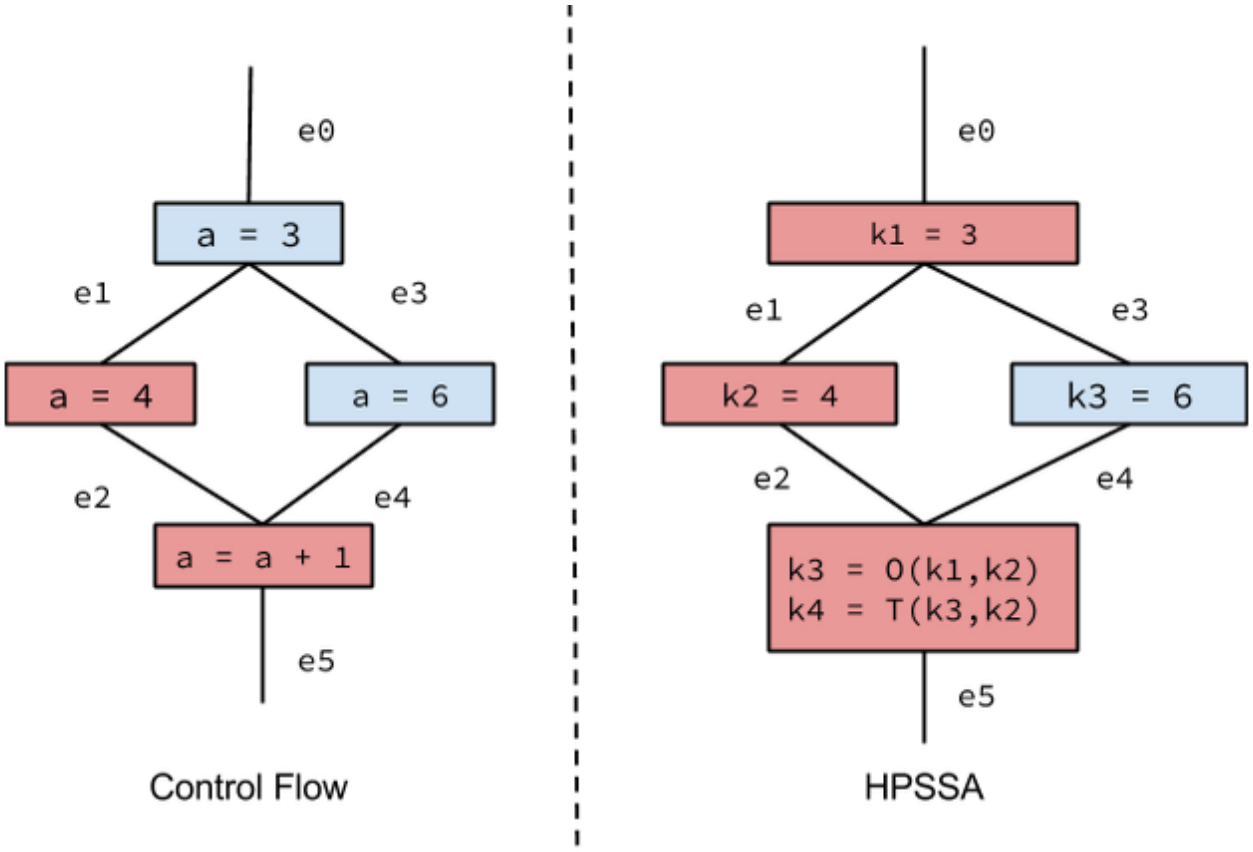
HotPath SSA

τ functions.



HotPath SSA

Hot and Cold Edges, Hot Blocks and Cold Blocks.



HotPath SSA

In Figure, e_0, e_1, e_2, e_5 are hot edges and e_3, e_4 are cold edges. Hot blocks have been indicated by a red tone and cold blocks have been indicated by a blue tone. Also, e_0 and e_5 are the starting and terminating edges of the program and we call them e_{start} and e_{end} for convinience.

Aside

Why care about DU coverage?

Section 3

Automatic Test Case Generation

Problem Statement

Given an HPSSA Graph G = (V, E), find a path $e_l \ldots e_m$ which maximizes du coverage.

where :

$$e_i = edges, \quad \forall i \in \{1 \ldots m\}$$

$$v_j = basic\ blocks, \quad \forall j \in \{1 \ldots n\}$$

Objective

Maximize the number of cold variables on any path.

Aside

How does this increase DU coverage?

Formulation 1

Intuition: Maximize the number cold blocks on any path.

Formulation 1

$$max \sum_{i=1}^{n} \omega_i \times e_i$$

where :

$$\omega_i = \begin{cases} 1 & hot\ \ block \\ 100 & cold\ \ block \end{cases}$$

Formulation 1

$$
\begin{aligned}
&subj\ to: \\
&\qquad e_i \leq 1, \quad \forall i \in \{1 \dots m\} \\
&\qquad e_{start} = 1 \\
&\qquad e_{end} = 1 \\
&\qquad \sum v_{e_{incoming}} = \sum v_{e_{outgoing}} \quad \forall v \in \{1 \dots n\}
\end{aligned}
$$

Formulation 2

Intuition: Maximize the number of cold edges on any path.

Formulation 2

$$max \sum_{i=1}^{n} \tau_i \times e_i$$

where :

$$\tau_i = \begin{cases} 1 & hot\ \ edge \\ 100 & cold\ \ edge \end{cases}$$

Formulation 2

$$
\begin{aligned}
&subj\ to: \\
&\qquad e_i \leq 1, \quad \forall i \in \{1 \dots m\} \\
&\qquad e_{start} = 1 \\
&\qquad e_{end} = 1 \\
&\qquad \sum v_{e_{incoming}} = \sum v_{e_{outgoing}} \quad \forall v \in \{1 \dots n\} \\
&\qquad \sum v_{e_{coldpair\_incoming}} = \sum v_{e_{coldpair\_outgoing}} \quad \forall v \in \{1 \dots n\}
\end{aligned}
$$

Bigger Picture

KLEE, SMT Solvers, Test Case Generation.

Unwanted Paths

Remove two kinds of paths:

- Explored Paths
- Infeasible Paths

Explored Paths

For an explored path $e_1 \dots e_n$, we add the constraint:

$$\sum_{i=1}^{n} e_i \leq n - 1$$

Infeasible Paths

For an infeasible path on edges $e_1 \ldots e_n$, we compute the *UNSAT Core* $e_l \ldots e_m$ and add constraint:

$$\sum_{i=l}^{m} e_i \leq l - m + 1$$

Section 4

Future

Present Work

A first implementation has been tested for intra-procedural unit test cases.

Future Work

- Compare performance on standard test suites.
- Complete the implementation of UNSAT Core on KLEE.
- Generalize the implementation to inter-procedural test cases.

References

Nikolai Tillmann, Jonathan de Halleux (2008)

PexWhite Box Test Generation for ṄET

Tests and Proofs Lecture Notes in Computer Science Volume 4966, 2008 pp 134-153

Cadar, Cristian and Dunbar, Daniel and Engler, Dawson (2008)

KLEE: Unassisted and Automatic Generation of High-coverage Tests for Complex Systems Programs

Proceedings of the 8th USENIX Conference on Operating Systems Design and Implementation pp 209 - 224

Paul Ammann, Jeff Offutt

Introduction to Software Testing

S Roy, Y. N. Srikant, 2010 (2010)

The Hot Path SSA Form: Extending the Static Single Assignment Form for Speculative Optimizations

Compiler Construction, Springer pp 304-323

The End

Section 5

Insight

Thermal Frontiers

Merging of Blocks.

Thermal Frontiers

Splitting of Blocks.