A Simple and Effective Measure for Complex Low-Level Dependencies

Dirk Beyer, **Ashgan Fararooy**

Simon Fraser University
University of Passau

Example

```
int a, b;

void swap ()  {
      a += b;
      b = a – b;
      a – = b;
}
```

```
int a, b;

void swap ()  {
      int temp = a;
      a = b;
      b = temp;
}
```

Comparison

```
void swap ()  {
      a += b;
      b = a – b;
      a – = b;
}
```

```
void swap ()  {
      int temp = a;
      a = b;
      b = temp;
}
```

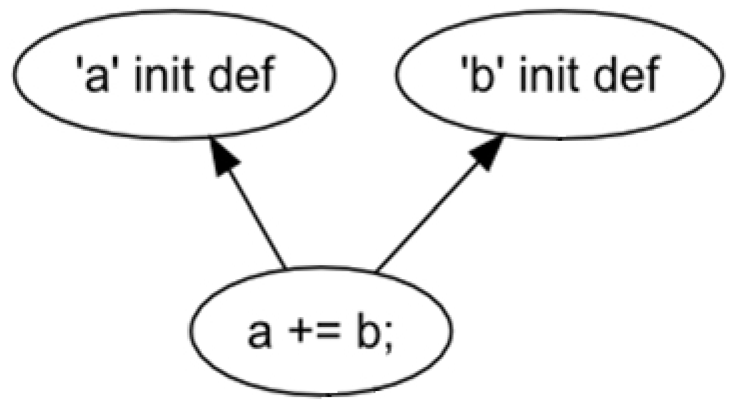
Measure / Function	swap (Left)	swap (Right)
Lines of Code	3	3
Cyclomatic Complexity	1	1
DepDegree	**6**	**3**

Dependencies

```
void swap ()  {
      a += b;
      b = a – b;
      a – = b;
}
```

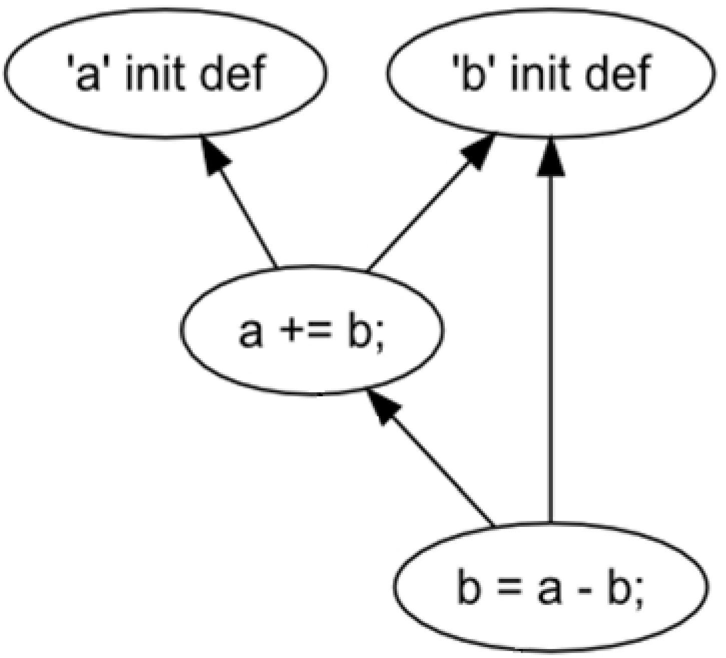
Dependencies

```
void swap ()  {
    a += b;
    b = a – b;
    a – = b;
}
```



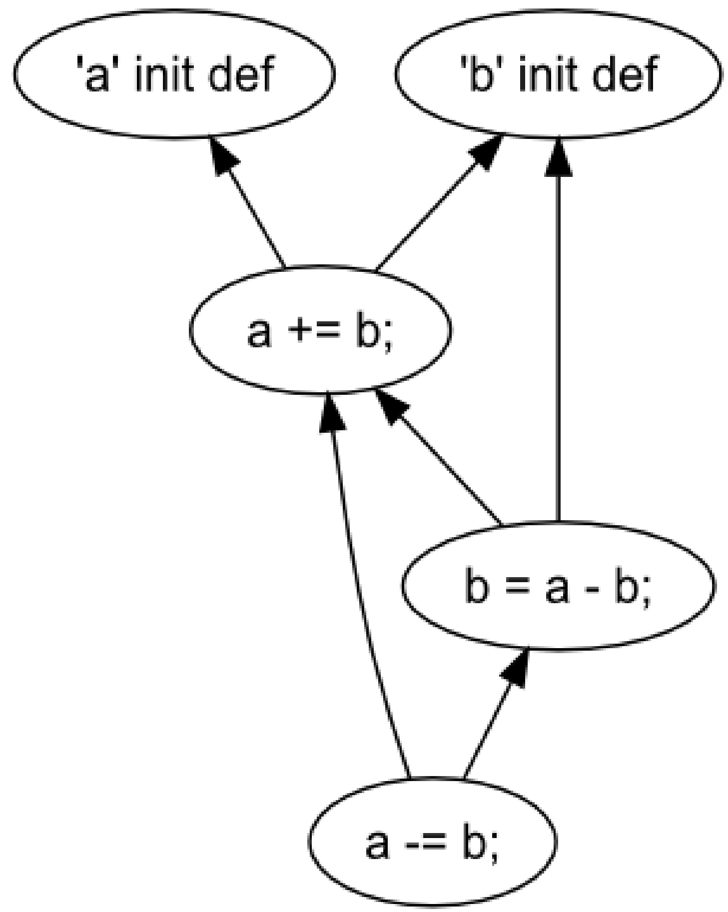
Dependencies

```
void swap ()  {
     a += b;
     b = a – b;
     a – = b;
}
```



Dependencies

```
void swap ()  {
      a += b;
      b = a – b;
      a – = b;
}
```



Definition

- Reaching Definitions

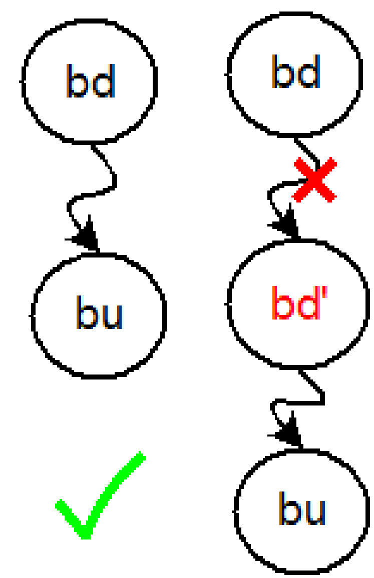
$$rd_G : B \times X \rightarrow 2^B$$

$$rd_G(b_u, x) = \{b_d \in B \mid b_u \text{ uses } x; b_d \text{ defines } x; b_d \rightsquigarrow b_u\}$$

$G = (B, F)$: Control-flow graph

B : Set of all program operations

X : Set of all program variables



Definition

- Use-Def Graph

$$S_G = (B, E)$$

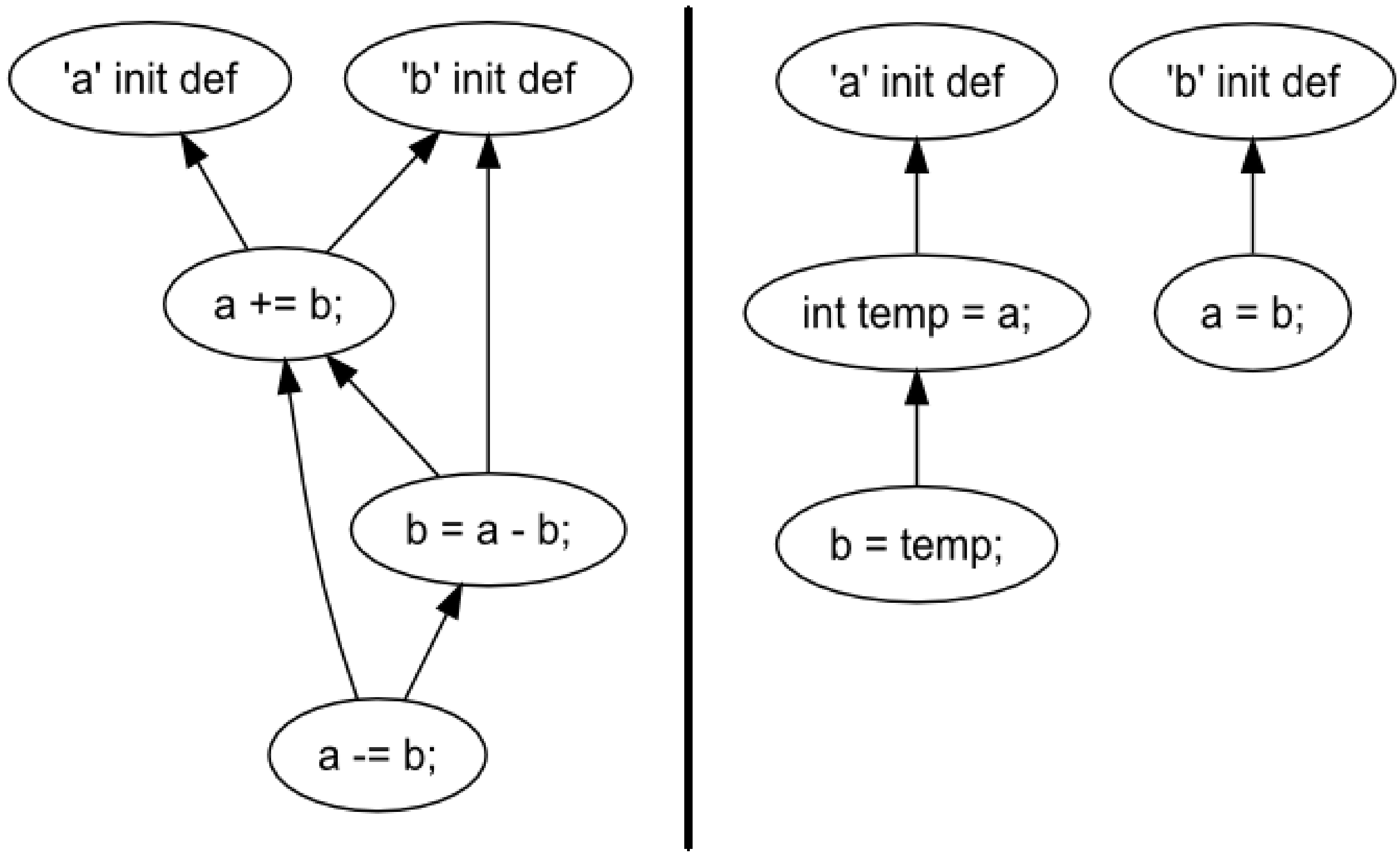
$$(b_u, b_d) \in E \Leftrightarrow \exists x \in X\colon\ b_d \in rd_G(b_u, x)$$

$G = (B, F)$: Control Flow Graph

B : Set of All Program Operations

X : Set of All Program Variables

Use-Def Graphs for Swap



Definition

- DepDegree
  - For **a single operation** of the program

$$dd_G : B \rightarrow \mathbb{N}$$

$$dd_G(b) = |\{b' \in B | (b, b') \in E\}|$$

$S_G = (B, E)$: Use-def graph

B : Set of all program operations

E : Set of edges in the use-def graph

Definition

- DepDegree
  - For **a function** of the program

$$dd_G : \mathbb{G} \rightarrow \mathbb{N}$$

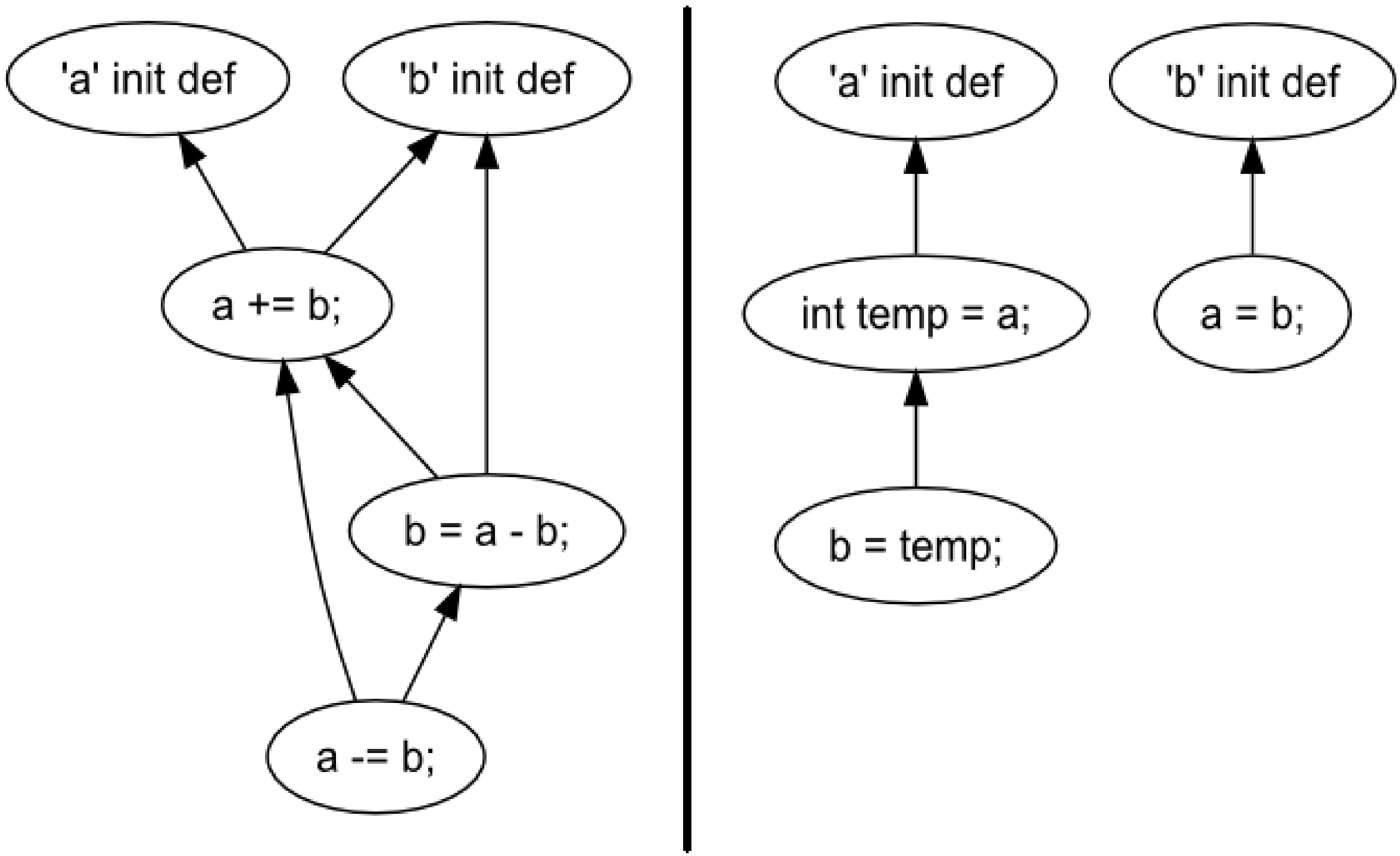
$$dd(G) = \sum_{b \in B} dd_G(b) = |E|$$

$\mathbb{G}$: Set of control-flow graphs (functions)

$S_G = (B, E)$: Use-def graph

$G = (B, F)$: Control-flow graph

Use-Def Graphs for Swap



Properties

Dep-Degree Indicator

- Simple
  - Based on well-known concepts
  - Straight-forward calculation
- Flexible
  - Applicable to all imperative programming languages
- Scalable
  - Applies to well-structured and unstructured code
- Independent
  - Complementing other indicators
- Automatic
  - Based on the facts present in the program code only

Evaluation

- Comparative Assessments
  - Compared alternative implementations
  - Compared DepDegree with other widely used indicators
  - Evaluated refactored code
  - Compared different revisions of open source software

Evaluation

- Eclipse Plug-in (DepDigger)
  - Automatic calculation of DepDegree
  - Highlighting operations based on DepDegree values

```java
public static int foo1(int n, int k)
{
    // Initialize 'arr' as an array
    // of length 'n+1' filled with ones
    int[]arr = arrayOfOnes(n + 1);

    for (int i = 0; i <= n; i++)
    {
        int temp = arr[0];
        for (int j = 1; j < i; j++)
        {
            arr[j] = arr[j] + temp;
            temp = arr[j] - temp;
        }
    }

    return arr[k];
}
```

Thank You

Questions?