Program Slicing:

Program Dependence Graphs

source

Interprocedural Slicing

Using Dependence Graphs,

Horwitz, Reps & Binkley

ACM TOPLAS, Jan., 1990.

available on course web site

http://www.dur.ac.uk/k.b.gallagher/local/Slicing/toplas.pdf pp1-9

The Program Dependence Graph 1

- Multigraph
- Vertices
  - Assignment
  - Control predicate, no side effects
  - Entry,
  - InitialState(v)
  - FinalUse(v) == End
- Control Dependence Edge: v -> w
  - v is entry; w is not in loop or conditional
  - v is control; w is immediately controlled by v

The Program Dependence Graph - 2

- Flow Dependent Edge: v -> w
  - v defines variable x
  - w uses (references) x
  - no intervening defs of x
- Loop Carried:
  - flow dependent, plus
  - edge back to controlling predicate
  - v, w in loop

The Program Dependence Graph - 3

- Loop Independent:
  - flow Dependent, plus
  - NO edge back to controlling predicate
- Def-Order Dependence: v -> w
  - v, w define the same variable
  - v, w are in same branch of conditional
  - there is z: v -> z and w -> z
  - v is “left” of w

Example *page 7*

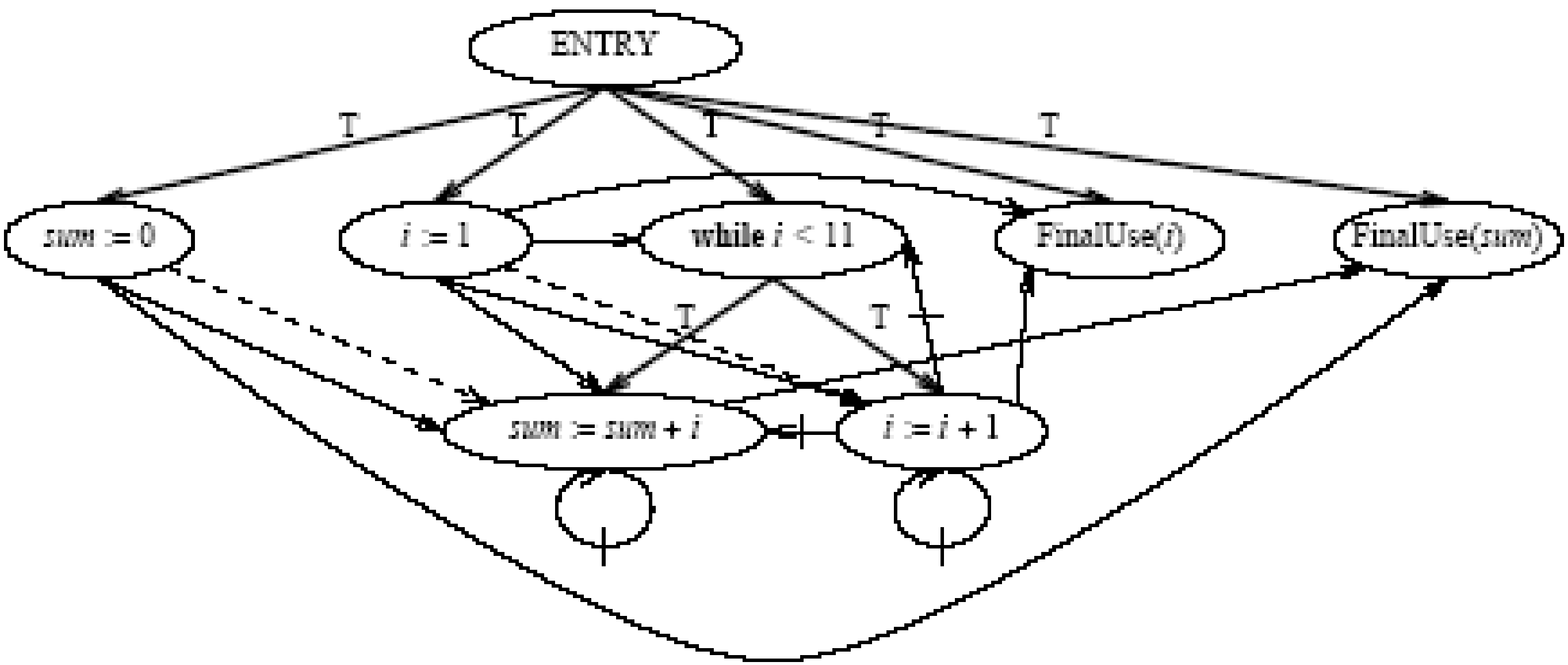
```
main
   sum := 0;
    i = 1;
    while ( i < 11) do
          sum := sum + I;
          i = I +1;
     od;
end( sum, i)
```

Example *page 7*

```
program Main
    sum := 0;
    i := 1;
    while i < 11 do
        sum := sum+i;
        i := i+1
    od
end(sum, i)
```

Edge Key

- control dependence
- loop-independent flow dependence
- loop-carried flow dependence
- def-order dependence



Slicing as Graph Reachability

Slice on v is the subgraph of all nodes and edges that can reach v (see page 8/9)

```
procedure MarkVerticesOfSlice(G, S)
declare
   G: a program dependence graph
   S: a set of vertices in G
   WorkList: a set of vertices in G
   v, w: vertices in G
begin
   WorkList := S
   while WorkList ≠ ∅ do
      Select and remove vertex v from WorkList
      Mark v
      for each unmarked vertex w such that edge w →_f v or edge w →_c v is in E(G) do
         Insert w into WorkList
      od
   od
end
```

Application of the algorithm

```
program Main
   i := 1;
   while i < 11 do
      i := i + 1
   od
end(i)
```

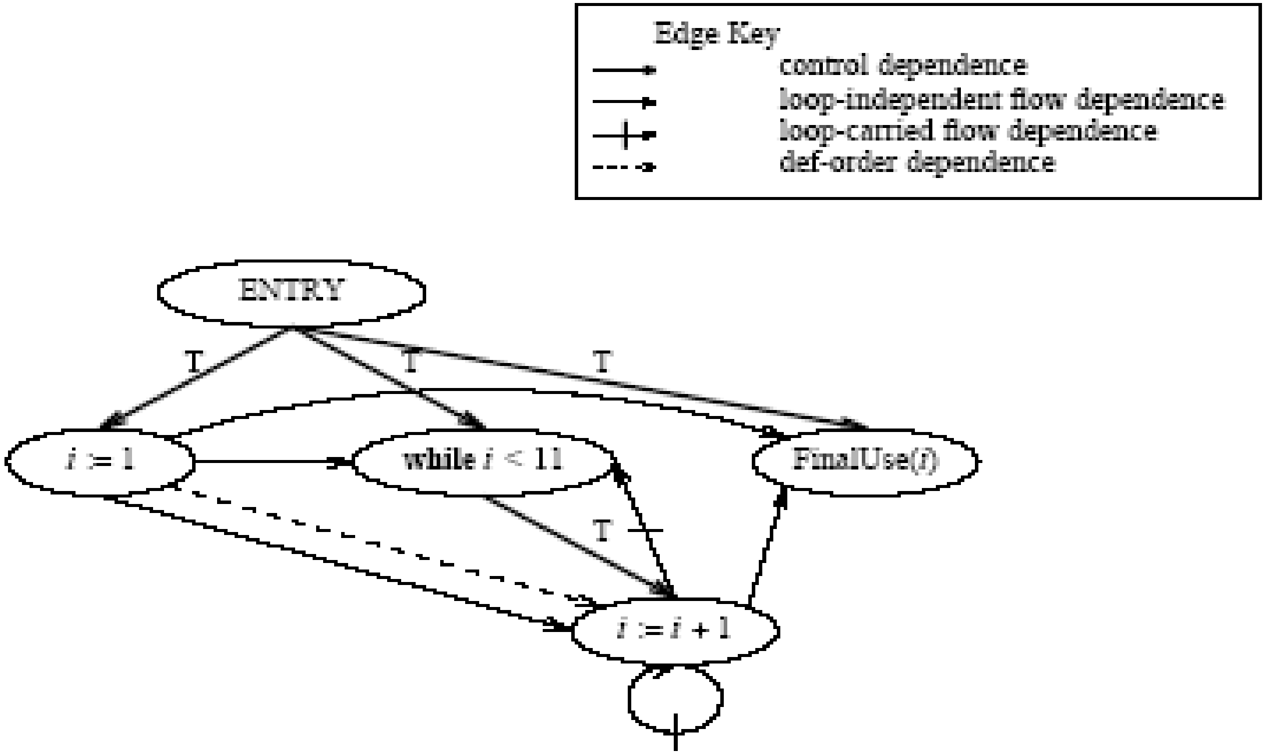


Figure 3. The graph and the corresponding program that result from slicing the program dependence graph from Figure 1 with respect to the final-use vertex for *i*.