Constant Propagation

66.648 Compiler Design Lecture (04/08/97)

Computer Science

Rensselaer Polytechnic

Lecture Outline

- Constant Propagation
- Examples
- Administration

Motivation

An expression that evaluates to a constant can be replaced by a direct load of the constant at compile-time, thus saving the overhead of evaluating the expression at run-time. The savings is proportional to the number of times the basic block (containing that expression) is getting executed.

Terminology

Constant propagation : “propagating” the constant value of a variable into its uses.

Constant Folding: “Folding” (simplifying) an expression with constant arguments at compile-time, so that expression can be replace by constant value.

E.g., x=3*9-7;

x=90;

y = x+7;

Example

a=1;

if (x < y) { b = 2; c=a +b;}

else { b = 4; c= b-1;}

d=a+c;

e=b+c;

Our constant propagation includes both of these cases.

Algorithm

Constant propagation is performed by an iterative algorithm similar to finding live variables.

Value mapping is a function which gives whether a variable is defined to be a constant or not a constant or not defined at all.

In[b] = value mapping at the entry of basic block.

Out[b] = value mapping at the exit of the basic block.

Rules for computing out[b] given in[b]

1. If a basic block does not update any variable, then the value mapping out[b] =in[b]
2. If a basic block consists of only assignment statement x=c; then out[b] is same as in[b] except out[b](x)=const (with value c)
3. If a basic block b consists only of the assignment statement x:= y op z, then out[b] is same as in[b] except for x, it is either const with a new value or nonconstant

Algorithm - Contd

4. If the basic block consists of a read statement read x, then out[b](x) nonconst
5. In all cases (when b contains multiple statements), out[b] is computed by using the above 4 rules.

Example

From One block to the Next

If a block has more than one predecessor, we need to compute in from the outs.

Let b1 and b2 be the predecessors of block b.

in[b](v) = nonconst if v is nonconst in either b1 or b2.

= undef if it is undefined in both

= const if it is const in both and have the same value.

More on one block to next

In[b](v) =nonconst if its const in b1 and b2 and different values.

= const, if it is const in one block and undefined in the other.

Optimistic versus pessimistic modes

Constant propagation Algorithm

For each basic block b do

{ for each variable v do

{ in[b](v) = undef;out[b](v) = Rules(undef);}}

change=true;

while change do

change=false;

for each basic block b, in dfs order do

Algorithm Contd

in[b](v) = meet(out[p][v]) for all predecessors of b.

old_out=out[b];

out[b]=Rules(in[b])

if out[b] !=old_out change = true;

// end for

// end while

Example

Convergence

Idea: For any basic block the in[b] and out[b] value mappings are monotonically nonincreasing functions

Comments and Feedback

Project 3 is out. Please start working. PLEASE do not wait for the due date to come.

We are in chapters 9 and 10. Please read that chapter and read the relevant portion of Java. Please keep studying this material and work exercises.