Topic 5a

Partial Redundancy

Elimination and SSA Form
References


- Keith Cooper, Linda Torczon. *Engineering a compiler*. 2004
Important Observation

• An operand of a $\phi$ function is regarded as occurring at the end of its corresponding predecessor block.

• The result of a $\phi$ function is regarded as occurring at the beginning of the block that contains it.
Problem Formulation for PRE

- For an expression, identify its partially redundant occurrences
- Safely insert occurrences (to new temporaries) at some points
- Delete the original partially (now fully) redundant occurrences. Replace them by loads from a temporary variable.
PRE Basics

- **Availability**: An expression, say, \( x*y \), is available at a program point \( p \) if

  - EVERY path from entry to \( p \) evaluates the expression before reaching \( p \)
  
  - And there are no assignments to \( x \) or \( y \) after the (last) evaluation but before \( p \) (on all paths).

- **Availability implies full redundancy.**
• Partial availability: An expression, say, $x*y$, is partially available at a program point $p$ if
  - SOME paths from entry to $p$ evaluates the expression before reaching $p$
  - And there are no assignments to $x$ or $y$ after the (last) such evaluation but before $p$.

• Partial availability implies partial redundancy.
PRE Basics (Cont.)

- Anticipatability: An expression e is anticipatable ("down-safe") at a program point p if it appears (without redefinition) along ALL paths from p to an exit of the program.

\[ x \times y \text{ is anticipatable here} \]

\[ B_2 \quad a \leftarrow x \times y \]

\[ B_1 \]

\[ B_3 \quad b \leftarrow x \times y \]
Safe Placement

- Safe insertion: If an inserted computation occurs at a point where the computation is anticipated.
- Unsafe insertion: means that some original path in the program did not contain this computation.
  - May increase the execution time of the program
  - May cause exceptions
Safe Placement: Example

This is called an "critical edge". Allowable if we can insert computations on an edge. Otherwise, need split.
Optimal Placement (Cont’d)

• Computationally optimal
  ▪ If no other safe placement can result in fewer occurrences of the computation along any path from entry to exit in the program.
Optimal Placement (cont’d)

- Lifetime optimal
  - Minimize the lifetimes of the introduced temporaries.
  - Intuition: delay an expression to the latest point (Lazy Code Motion [KnoopEtal92, DrechslerStadel93])
**Traditional Partial Redundancy Elimination**

Before PRE

- $B_1: a \leftarrow x*y$
- $B_2$
- $B_3: b \leftarrow x*y$

After PRE

- $B_1$:
  - $t \leftarrow x*y$
  - $a \leftarrow t$
- $B_2$:
  - $t \leftarrow x*y$
- $B_3: b \leftarrow t$

SSA form

Not SSA form!
SSAPRE: Motivation (Cont.)

- Traditional PRE needs a postpass transform to turn the results into SSA form again.
- SSA is based on variables, not on expressions.
SSAPRE: Motivation (Cont.)

• Representative occurrence

  - Given a partially redundant occurrence $E_0$, we define the representative occurrence for $E_0$ as the nearest to $E_0$ among all occurrences that dominate $E_0$.

  - Unique and well-defined
Observation

- Every use-def edge of the temporary corresponds directly to a redundancy edge for the expression.
Intuition for SSAPRE

• Construct FRG for each expression E
• Refine redundant edges to form the use-def relation for each expression E’s temporary
• Transform the program
  ▪ For a definition, replace with t←E
  ▪ For a use, replace with ←t
  ▪ Sometimes need also insert an expression