Topic 5a

Partial Redundancy

Elimination and SSA Form
References


- Keith Cooper, Linda Torczon. Engineering a compiler. 2004
Outline

• Partial Redundancy Elimination (PRE)
• SSAPRE (a short summary)
KCC Compiler Infrastructure

Frontend
- Fortran
  - fe90
- C
  - gfec
- C++
  - gfecc

Middleend
- Very High WHIRL
- High WHIRL
- Middle WHIRL
- Low WHIRL
- Very Low WHIRL
- CGIR

Backend
- Asm file

Source to IR (Scanner → Parser → RTL → WHIRL)
- VHO (Very High WHIRL Optimizer)
- Standalone Inliner
- W2C/W2F

IPA (inter-procedural analysis & opt)
- LNO (Loop unrolling/fission/fusion/tiling/peeling etc)
- PREOPT (point-to analysis etc)

WOPT
- SSAPRE (Partial Redundancy Elimination)
- VNFRE (Value numbering based full redundancy elim.)
- RVI-1 (Register Variable Identification)

RVI-2

Some peephole opt

Cflow (control flow opt)
- EBO (extended block opt.)
- PQS (predicate query system)
- Loop Opt (Unrolling + SWP)
- GCM (global code motion), HB sched (hyperblk schedule)
- GRA/LRA (global/local register alloc)
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.
Partial Redundancy Elimination

Is the program in SSA form before and after PRE?

$B_1$

$t ← x*y$

$a ← t$

$a ← x*y$

$B_2$

$t ← x*y$

$B_3$

$b ← x*y$

$b ← t$

X*y is partially redundant here

X*y is made redundant here

2008-03-26
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.
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.
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.

- Safe
- Not safe
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])
Outline

• Partial redundancy elimination (PRE)

• SSAPRE (a short summary)
SSAPRE
SSAPRE: Motivation

- Traditional data flow analysis based on bit-vectors do not interface well with SSA representation
Traditional Partial Redundancy Elimination

Before PRE

\[
B_1: a \leftarrow x \times y \\
B_2: \text{Blank} \\
B_3: b \leftarrow x \times y
\]

After PRE

\[
B_1: t \leftarrow x \times y \\
B_2: t \leftarrow x \times y \\
B_3: b \leftarrow t
\]

SSA form

Not SSA form!

2008-03-26
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
FRG (Factored Redundancy Graph)

Without factoring

Factored

\[ E = \Phi(E, E, \perp) \]
FRG (Factored Redundancy Graph)

Assume $E = x \cdot y$

$E = \Phi(E, E, \bot)$

$t_0 \leftarrow x \cdot y$

$t_1 \leftarrow x \cdot y$

$t_2 \leftarrow \Phi(t_0, t_1, t_3)$

Note: This is in SSA form
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 $t \leftarrow E$
  - For a use, replace with $t \leftarrow t$
  - Sometimes need also insert an expression
An Example

\[ a_1 \leftarrow \]

\[ a_2 \leftarrow \phi(a_4, a_1) \]

\[ a_3 \leftarrow a_2 + b \]

\[ a_4 \leftarrow \phi(a_2, a_3) \]

\[ a_4 \leftarrow a_4 + b \]

\[ \text{exit} \]

\[ a_1 \leftarrow \]

\[ t_1 \leftarrow a_1 + b_1 \]

\[ t_2 \leftarrow \phi(t_4, t_1) \]

\[ a_2 \leftarrow \phi(a_4, a_1) \]

\[ t_3 \leftarrow a_3 + b_1 \]

\[ t_4 \leftarrow \phi(t_2, t_3) \]

\[ a_4 \leftarrow \phi(a_2, a_3) \]

\[ t_4 \leftarrow t_4 \]

\[ \text{exit} \]