

Some Comments on “A Solution to a Problem with Morel and Renvoise’s ‘Global Optimization by Suppression of Partial Redundancies’”

Arthur Sorkin  
Tesuji, Inc.

The deficiency in Morel and Renvoise’s algorithm<sup>[1]</sup> cited by Drechsler and Stadel<sup>[2]</sup> in their Section 2 was solved previously by Joshi and Dhamdere<sup>[3]</sup> and, later, independently solved by Chow<sup>[4]</sup>. Joshi and Dhamdere completely drop the CONST factor in the computation of PPIN. Chow uses  $CONST = ANTIN * PAVIN$ , the same as Drechsler and Stadel.

Using the Joshi-Dhamdere’s solution for the example given in Drechsler and Stadel’s Figure 1, and representing the solutions to the equations as bitvectors with bit 1 representing  $a_1$  and bit 2 representing  $a_2$ , we get:

	PPIN	PPOUT	INSERT	DELETE
1	00	11	11	00
2	11	11	00	10
3	11	00	00	11

The Joshi-Dhamdere solution has the advantage hoisting redundant operations as far as possible, which can eliminate additional redundancies. In this example, both  $a_1$  and  $a_2$  are inserted in block 1,  $a_1$  is deleted from block 2, and both are deleted from blocks 2 and 3. The code motion produced in this example is the best possible in the sense that the most operations are removed from the loop. The disadvantage of the Joshi-Dhamdere solution is that it can unnecessarily increase the lifetime of hoisted expressions, possibly increasing register usage.

Using Chow’s solution, we get:

	PPIN	PPOUT	INSERT	DELETE
1	00	00	00	00
2	00	00	00	00
3	00	00	00	00

The Chow solution has the advantage of minimizing the lifetimes of hoisted expressions. The disadvantage is that some redundancies may not be eliminated. In order for PPOUT to be true for some expression in a particular block, PPIN must be true for that expression in all immediate successor blocks. But PPIN cannot be true unless PAVIN is true. Since PAVIN is false for both  $a_1$  and  $a_2$  in block 2, PPIN is also false in block 2 for both expressions. Since block 2 is a successor of block 1, PPOUT is false in block 1 for both  $a_1$  and  $a_2$ , and neither expression can be moved to block 1. Similar reasoning shows that nothing can be moved to block 2. The reason that nothing is moved is that block 1 is a predecessor of both block 2 and block 3. If that were not the case, then  $a_1$  and  $a_2$  would be hoisted out of block 3.

Both the Joshi-Dhamdere and Chow solutions allow the Morel and Renvoise algorithm to easily handle subexpressions. In addition, both a modified version of the Joshi-Dhamdere’s solution<sup>[5]</sup> and Chow’s solution allow Morel and Renvoise’s algorithm to be extended to perform strength reduction. Strength reduction is normally profitable even if the number of times an expression occurs isn’t reduced along any path.

The modification proposed by Drechsler and Stadel in their Section 4 is disadvantageous because it approximately doubles the number of basic blocks, and the running time of the algorithm is at least linear in the number of basic blocks. If explicit loops are generated in the way shown in Figure 1, below, then the number of basic blocks is increased, but only where needed. The extra blocks are worth while, because hoisting expressions out of loops generally results in the most savings in execution time. Either Joshi-Dhamdere's solution or Chow's solution will work well with this shape flow graph. With a compiler designed to support optimization, flow graphs of the sort shown in Drechsler and Stadel's Figure 1 hardly ever occur in practice.

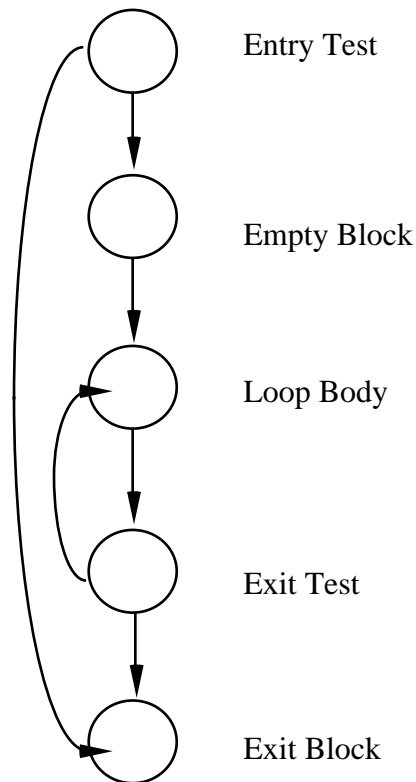


Figure 1

#### REFERENCE

- [1] Morel, E. and Renvoise, C. Global Optimization by Suppression of Partial Redundancies. *Commun. ACM* 22, 2 (1979), 96-103, 111-126.
- [2] Drechsler, K. and Stadel, M. A Solution to a Problem with Morel and Renvoise's "Global Optimization by Suppression of Partial Redundancies". *TOPLAS* 10, 4 (1988) 635-640.
- [3] Joshi, S. and Dhamdere, D. A Composite Hoisting-Strength Reduction Transformation for Global Program Optimization. *Int. J. Comp. Math* 11, 1-2 (1982) 22-41, 111-126.
- [4] Chow, F. A Portable Machine Independent Optimizer - Design and Measurements. Ph. D. Dissertation, 1983, Department of Electrical Engineering and Technical Report 83-254, Computer Systems Laboratory, Stanford University.

[5] Sorkin, A. Cydrome, Inc. FORTRAN Compiler Global Optimizer.