Basic Block Optimizer

Henry Gordon Dietz; Department of Electrical and Computer Engineering, University of Kentucky; Lexington, Kentucky

Abstract

Here are three versions of the project code: the original, the solution, and an improved solution. They differ only in a single file and build different executables. The original is built using `bb5orig.c` and is named `bb5orig`. The sufficient solution is built using `bb5.c` and is named `bb5`; it does as much as I expect you to have done. The improved solution is built using `bb5plus.c` and is named `bb5plus`. It differs primarily in that it understands that subscript expressions that differ by addition, subtraction, or exclusive-OR of a non-zero constant are not aliased (i.e., cannot refer to the same memory cell).

Make

The `Makefile` is very straightforward, and contains the usual stuff, including the ability to `make clean`, `make notes`, `make test`, and `make tar`. However, simply typing `make` will do everything important.

Solution

Not much to say about this. The only trick is that `ttoi()` is used to obtain numbers by which the operand order is normalized. The only auxiliary data structures are two tuples that get pre-installed in each block: `zero` for `const(0)` and `negone` for `const(-1)`, which simplifies the checks for local optimizations.

Plus Version

There are several tweaks applied here:

- The indexed alias check is moved into a separate function called `sameval()`, which can recurse to check if two subscript expressions differ by a constant
- The normalized order is opposite to that of the solution
- A variety of minor rearrangements have been made, including that all loads and stores are treated as indexed up until they are printed

Testing

There is just one test input, `test.c`. It should generate output with the properties listed in `oracle`... but not in that notation. The test outputs are generated into `orig`, `output`, and `plus`.

It appears that all three versions behave as they were supposed to.

Author Biography

Henry (Hank) Dietz earned his PhD at Polytechnic University and joined the faculty at Purdue University's School of Electrical and Computer Engineering in 1986. Since 1999, he has been a Professor and Hardymon Chair at the University of Kentucky. For some reason, he still seems to write many compilers....