P5 report

Ruoyu Wu

Constant Branch Folding

We first go through all the instructions to check if they condition on a constant, and add to `removeSet` if they do. Then, for each such instruction, we remove it and replace with the `BranchInst` jumping directly to the target basic block. For the other basic block (that we determine to remove the edge), we remove the original instruction from its predecessor list.

Dead Block Removal

We first preform DFS (using built-in `dfs_ext_begin` and `dfs_ext_end`) and add every visited blocks to `visitedSet`. Then we collect all the basic blocks that are not in visited blocks as `unreachableSet`. At last, we remove them by telling their successor and detach them from Function.

Loop Invariant Code Motion

getAnalysisUsage

We refer to the code provided by the handout.

isSafeToHoistInstr

For an instruction, we first use `hasLoopInvariantOperands` to check if all of its operands are loop invariants. Thanks to the built-in loop analysis in LLVM, we can easily access these analysis results. Then we use `isSafeToSpeculativelyExecute` to check if the instruction has side effects or not. At the last, we make sure the instruction is one of the several instruction types we are targeting.

hoistInstr

We use `moveBefore` to move the instruction before the terminator of the loop header block.

hoistPreOrder

We follow the pseudocode and tips in handout to implement this function.