888 LLVM - Exercise Week 6
For 01. Mar 2011

1 Fix missed optimization in instcombine

(A & B) | (A ^ B) → (A | B)

The optimization above is currently missed in gcc, icc and LLVM. In this exercise we will create a patch that implements the optimization.

a) Write a file complexOr.ll that implements a function @$i64$ $complexOr(i64\ a, i64\ b)$ which calculates (A & B) | (A ^ B).

b) What do you expect will happen if you optimize it with `opt -instcombine`. What did actually happen?

c) Use complexOr.ll to create a new test case in `test/Transforms/InstCombine` that tests that the simplification you expected is performed.

d) Verify that this test case fails when running `llvm-lit complexOr.ll`.

e) Open `lib/Transforms/InstCombine/InstCombineAndOrXor.cpp` and implement the needed simplification in $visitOr()$.

f) Run `llvm-lit complexOr.ll` again and verify that the new test case now passes.

g) Add additional test cases to complexOr.ll that match similar patterns.

h) Generalize your simplification to optimize all those test cases.

i) Run `make check` and verify your generalized simplification works and does not break any other LLVM test cases.

j) Create a patch that contains all your changes.

Please have your patches electronically available or sent them to me. We will review one of them next week such that it can be submit to LLVM. Hence use a recent checkout of LLVM and verify that your patch is in a shape, such that it can be included in a production compiler. You can work in small teams to create this patch. If you need any help, pass by at my office (Dreese 572).