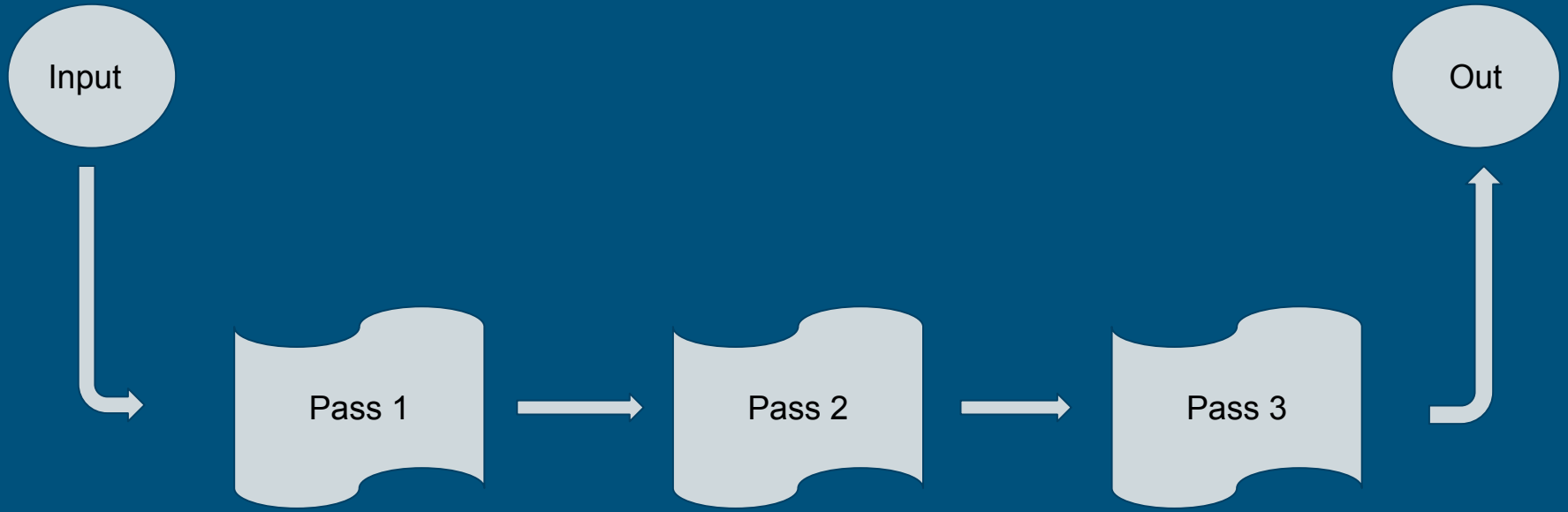
Toward Fixed-Point Optimization in LLVM

Nathan Wilson, KCG
Hal Finkel, Argonne

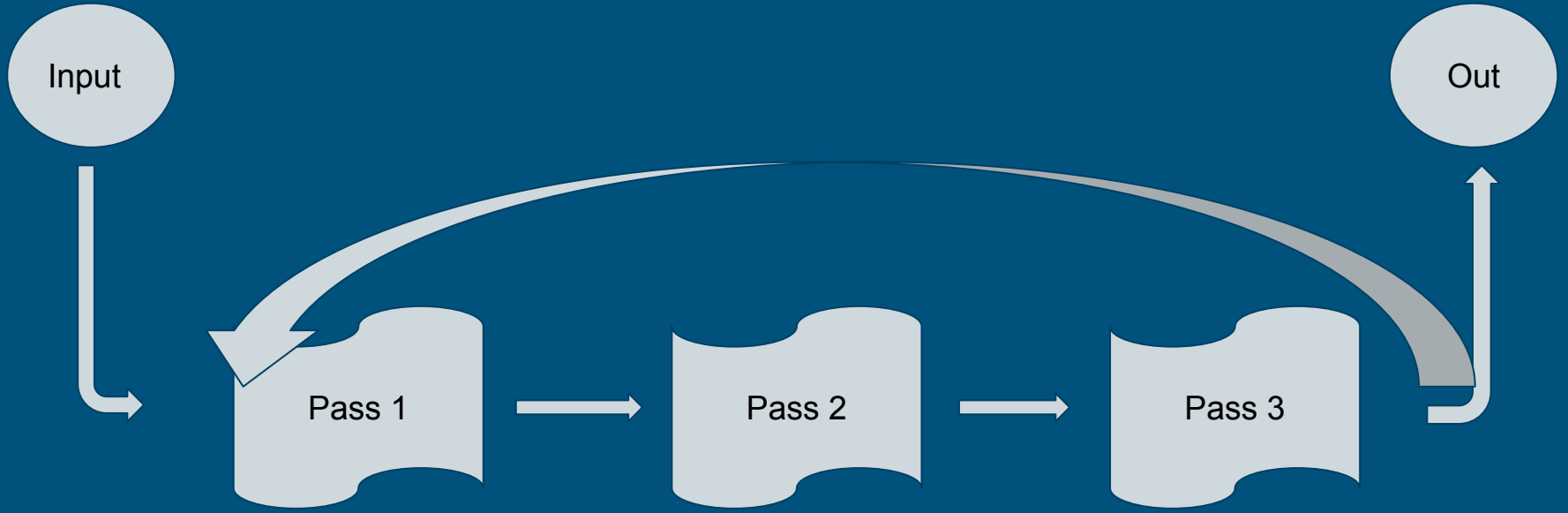
Normal Optimization Pipeline



Observations and Problems of the Optimization Pipeline

- Anecdotally, Generated Code may be more performant when the whole optimization pipeline is run multiple times
  - Is it true generally?
- Optimizations may fight with each other by undoing previous changes and creating new changes which a particular optimization thinks is better (cycle)
- When forcing multiple pipeline runs, optimizations may be unnecessarily run
- An optimization may produce a false positive (bug) telling the pass manager that code has changed when it actually did not - causes unnecessary analysis invalidation

Fixed-Point Optimization Pipeline



Motivation and Benefits of Fixed-Point

- Generated Faster Code!
  - We would like to have code generated via fixed-point to be faster than the normal pipeline
  - Run optimization pipeline until there are not any changes in IR
- Smaller Code Size
- Faster Compilation Times!
  - Could run the Normal Pipeline N number of times, but not all optimizations need to be run
- Find out if Optimizations fight with each other

Actual findings of the investigation

- Performance: +/- a few percent on the test suite
- Code size: Some things get a few percent better, overall things get 30% bigger
- A limited number of iterations needed in practice (no more than 4 on the test suite - most things took only 2 iterations)

Come see the Poster!