# gOlogy: impact of -O* on -g

Alexandre Oliva

aoliva@redhat.com

`http://people.redhat.com/~aoliva/`

GNU Tools Cauldron, 2018

## Summary

- Project description
- Ground assumptions
- Main highlights

## gOlogy project

- Impact of optimization on debuggability
  - every pass enabled at -Og..-Ofast over -O0
- GCC 8, GNU/Linux, GNU binutils 2.29
  - files, lines, columns, ranges, blocks
  - VT: variable tracking from REGs and MEMs
  - VTA: debug binds at scalar assignments
  - SFN: debug markers between statements
  - LVu: multiple views per code address

## Setting expectations right

- Optimized out: sharing may cause early death
- Break at *0x1abe1add: misses code copies
- Setting vars in the debugger vs:
  - Shared/multiple locations
  - Removed conditional dead code
- Almost 150 flags, refer to full report

## Highlights

- Mostly seamless adjustment of debug binds
  - gimple more so than RTL
  - generic logic for dead pseudos and removals
  - improve moving stmts to dominating blocks
  - some passes need adjustments
  - improve no-longer-addressable variables?
- Surprise: -Wnull-dereference changes code

## Losing track of variable locations

- Disregard variable locations
  - delay slots (-Og), peepholes, autoinc
- View-related tracking of MEM stores
  - --tree-dse, --tree-sink at -Og, improvable?
  - --tree-{loop,slp}-vectorize at -O3, hopeless?
- Tracking of dismembered compound types
  - --split-wide-types, --tree-sra
  - --ipa-sra: drops SRAed parms altogether

## Conditional binds and markers

- Avoid discarding notes at CFG reorgs
  - jump threading
  - if conversion
  - phiopt
  - crossjumping/tail-merging
- DWARF extension: loclist for cond views?

## Loop optimizations

- Some are ok!
  - --split-loops, --unswitch-loops, --peel-loops
- Reordering the iteration space: confusing!
  - --loop-unroll-and-jam, --tree-loop-vectorize
- IV opts (--branch-count-reg) may lose bindings

## Subprogram transformations

- Partial inlining
  - Extension: link back to enclosing fragment
  - Combine with inlined enclosing scope
- Identical Code Folding
  - Conditional notes for combined functions?
  - Identify active variant from callers?
  - Separate debug info descriptions?

## Thank you!

Get the full (WIP) report

`http://people.redhat.com/~aoliva/`

Alexandre Oliva

aoliva@redhat.com