Isolation and Analysis of Optimization Errors

by

Mickey R. Boyd

David B. Whalley

Florida State University

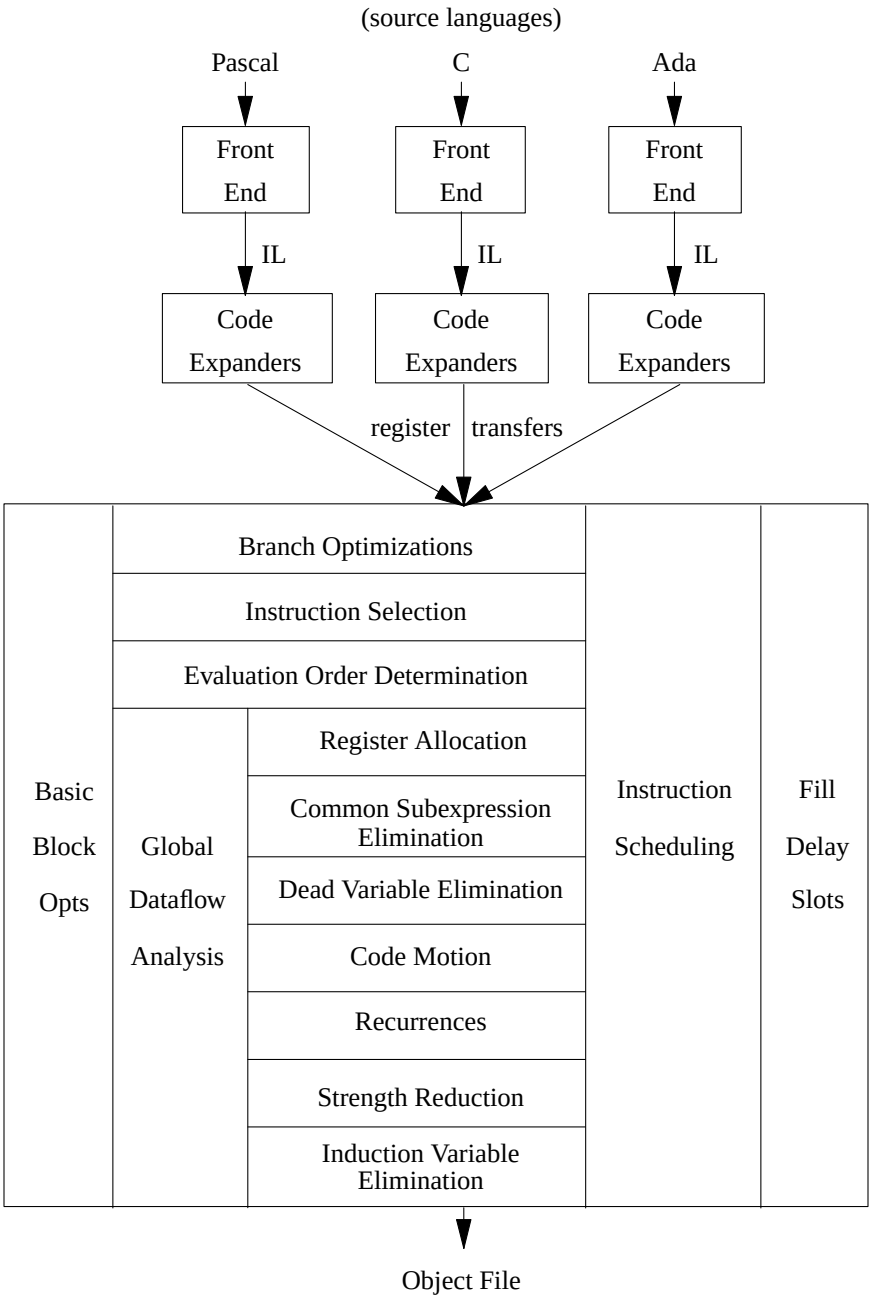
Motivation

- much time and effort spent during testing when
  - retargetting compilers to new machines
  - developing new optimizations

Overview

- developed an error isolator (VPOISO) that automatically
  - determines the first invalid transformation
  - identifies that point in the compiler
- developed a graphical optimization viewer (XVPODB) that
  - depicts the state of the instructions before and after each transformation
  - can be used to analyze the invalid transformation discovered by the error isolator

VPO Compiler System



VPO Summary

- Code generation performed before optimization.
- Each RTL corresponds to an assembly instruction.
- VPO only manipulates RTLs
  - eliminates many phase ordering problems
  - simplifies identication of changes to program representation
  - provides a simple and consistent form for program representation

Transformations

- VPO was modified to identify each change to RTL structure
  - insert RTL
  - delete RTL
  - modify RTL
  - ...
- VPO also identifies each serial sequence of changes, called a transformation, that preserves the meaning of the program.

Types of Transformations

- Necessary: Required to produce code that can be compiled and executed correctly. Examples include
  - assigning pseudo registers to hardware registers
  - fixing the entry and exit points of a function to manage the run-time stack
- Improving: Not required. Typically makes compiled program
  - faster
  - and/or smaller

Limiting the Number of Improving Transformations

- VPOISO invokes VPO with option to limit the number of improving transformations applied to a function.
- Limiting the number of improving transformations was accomplished using `setjmp()` and `longjmp()` library functions.
- Required very few modifications to VPO.

Modifying VPO with Setjmp and Longjmp

```
/*  Within a high level routine in VPO.  */
...
/*  Save current environment.  */
setjmp(my_env);

/*  If more optimizations allowed then
    perform register coloring.  */
if (moreopts)
   color();
...
/*  Within the routine that is invoked when
    the end of a transformation is identified.  */
...
/*  If reached limit, then set flag to not allow any
    more optimizations and restore environment.  */
if (maxtrans == opttransnum) {
   moreopts = FALSE;
   longjmp(my_env, 1);
   }
...
```

Information Required for Isolating an Error

- reads an input file of information to guide the isolation process

```
cexfiles: y1 y2 y3 y4 #
link command: cc -o yacc y1.o y2.o y3.o y4.o
execute command: yacc cgram.y
maximum time: 15
desired output file: yacc.out
actual output file: y.tab.c
compilation flags: LVGOCMSFA
disregard strings:
```

Algorithm for Isolating an Optimization Error

- checks if works with all optimizations
- checks if works with no optimizations
- performs binary search

```
lastmin = 0;
lastmax = total number of improving transformations
while (lastmax - lastmin > 0) {
    midnum = (lastmin + lastmax)/2;
    recompile program with only the first midnum
        transformations performed
    remove actual output file
    link and execute program
    if (actual output file == desired output file)
        lastmin = midnum+1;
    else
        lastmax = midnum;
    }
if (last result was incorrect)
    badtrans = midnum;
else
    badtrans = midnum+1;
```

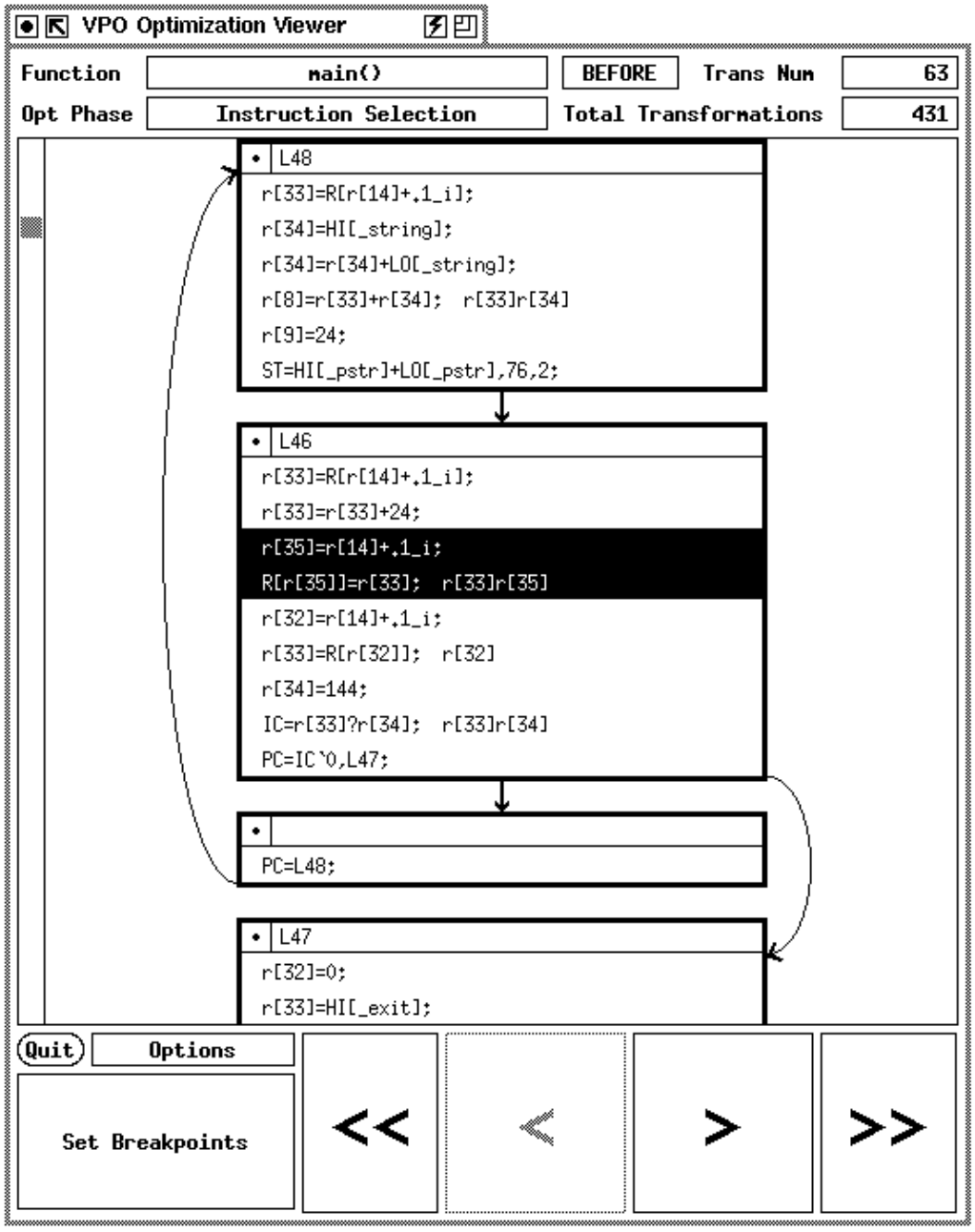
Performance of VPOISO

- total of 13,955 improving transformations when compiling Yacc
- isolating this error required
  - 16 compilations/executions
  - about 10 minutes
- a new enhanced technique
  - uses file merging instead of recompilation when possible
  - isolated same error in Yacc in about 6 minutes

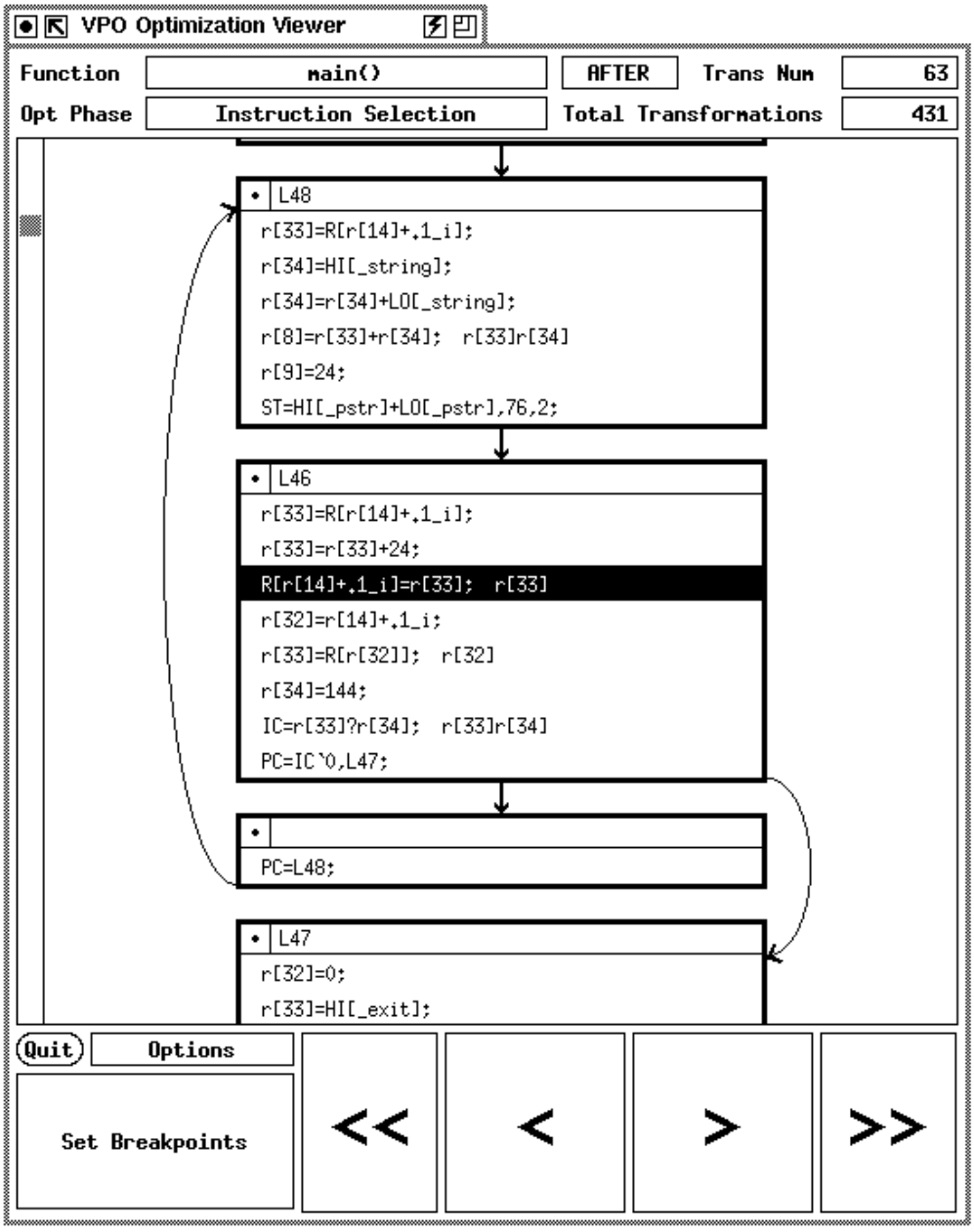
XVPODB

- Used to display the state of the generated instructions before and after any transformation.
- Implemented using X-Windows and UNIX sockets.
- Receives messages from VPO as the code is being optimized.
  - entry/exit of optimization phases
  - entry/exit of transformations
  - detail of changes

XVPODB Displaying Transformation (Before State)



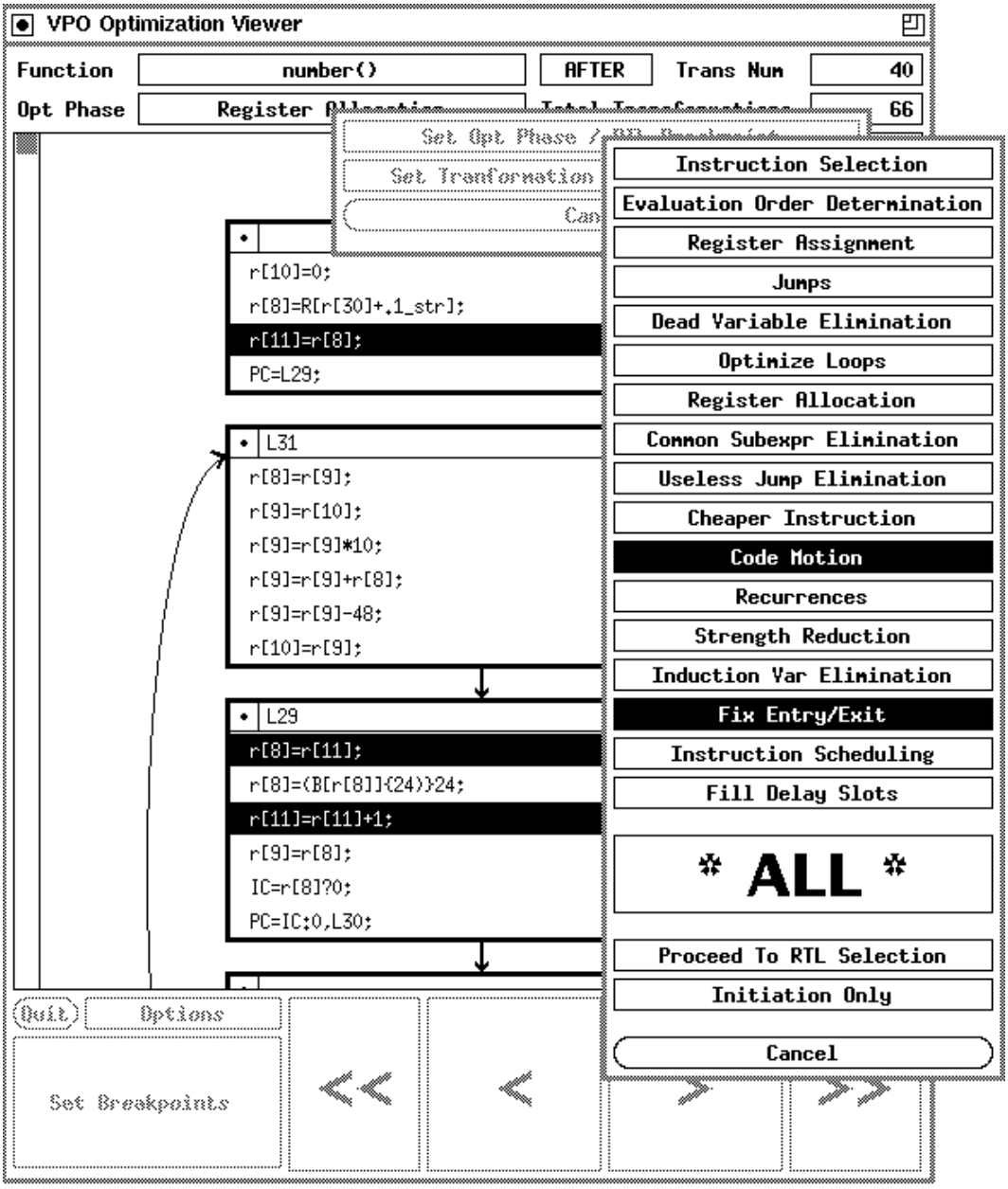
XVPODB Displaying Transformation (After State)



XVPODB Breakpoints

- Two main types of breakpoints.
  - — transformation number
  - — any combination of
    - optimization phases
    - RTLs

XVPODB Breakpoint Selections



Conclusions

- VPOISO and XVPODB are useful when
  - retargetting compilers to new machines
  - implementing new optimizations
  - maintaining a compiler
- In addition, XVPODB can be used a teaching tool.

VPOISO Log

starting binary search to isolate error within 13955 transformations

error within main to gtnm (transformation 1 to 13955)
compiling program: applying transformations 1 to 6978
compiling y1.cex
compiling y2.cex
stopped optimization of chfind after 11 improving transformations
linking program
executing program
execution was incorrect

error within main to chfind (transformation 1 to 6978)
compiling program: applying transformations 1 to 3489
compiling y1.cex
stopped optimization of closure after 197 improving transformations
compiling y2.cex
linking program
executing program
execution was correct

…

error within setup (transformation 4490 to 4491)
compiling program: applying transformations 4490 to 4490
compiling y2.cex
stopped optimization of setup after 500 improving transformations
linking program
executing program
execution was incorrect

incorrect transformation isolated to optimization 500 in function setup