Assignment 04: Data Flow Analysis

Angelo Porcella, Emery Call, Jasmine Vang

December 4th, 2024

1 Introduction

Our data-flow analysis tool meets the assignment requirements, which includes both an integer sign analysis and reaching definition analysis for the Where3addr language. We successfully implemented a command-line interface, and a corresponding output table.

2 Architecture

The tool starts by receiving the analysis mode (signed, reaching) and the path of the program to be analyzed. The tool then parses the program. Depending on the analysis mode, the appropriate analysis is executed for the program, and the results are output in a table.

2.1 Parse Source Code (parser.py)

The parsing is completed line by line, with specific cases determining the location of the line split. The cases are: assignments, goto, conditional goto, and halt. Prior to parsing each line, the line number is removed, then the program goes through parsing for use in the tool.

2.2 Lattice (domain.py)

This creates a partial order graph for domain elements. We represented the partial order rules as directed edges in a networkx directed graph. There is a join operator that computes the least upper bound and state joining

that merges two abstract states. First, the relationships of the elements are defined with the partial-order rules, then the graph of the lattice is created with the rules, any values or states that need to be merged are then joined.

In the lattice structure, the nodes represent types, and edges denote relationships where one type is a subset of another. The join computes the least upper bound of two elements, which is used for merging information. Figure 1 is an example of the Integer Sign Analysis Lattice.

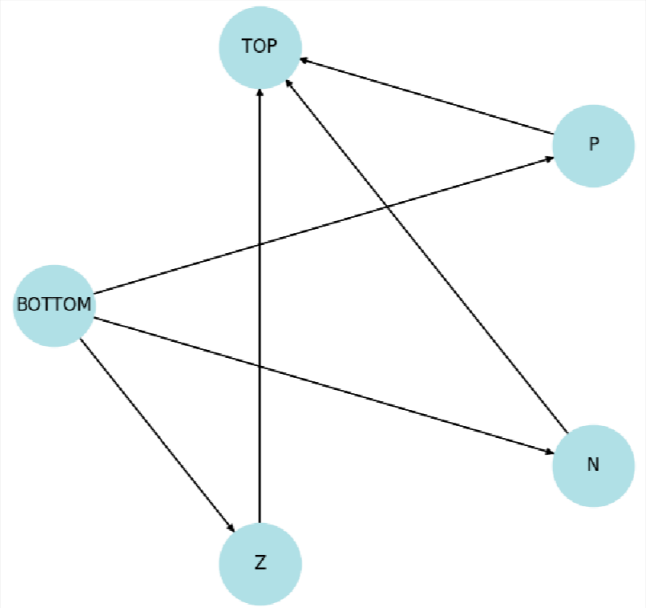

Figure 1: Integer Sign Analysis Lattice Example

2.3 Integer Sign Analysis (integer_sign_analysis.py)

Integer sign analysis reads instructions that have been parsed from the input program, then assigns a value for zero, positive, or negative based upon the assignment line in the file. The rules for the analysis are a long list of IF statements containing the operator, and the current values for each variable. It then outputs based upon the IF statement entered. There are statements for each operation (addition, subtraction, multiplication, and division). Each block of rules handles every scenario possible to ensure no errors. For handling true/false statements (i.e. '=' or '<'), values not equal to zero are assigned 'TOP'. Values after the '<' operator resolves false, are as follows:

if the value is equal to zero, the value 'Z' is assigned, if not, the value 'P' is assigned.

2.4 Reaching Definition Analysis (reach_analysis.py)

A Reaching definition analysis tracks the active variable assignments. To implement the domain for this we first found all variable assignments, generated all subsets of them, and then inserted the subset relationships between those subsets into our lattice. Figure 1 is an example of this lattice construction on prog_1.w3a. The flow function implementation involves both adding some variable assignments and removing others. We join the current variable assignment in the case of assign_var and assign_op because they involve variables and their assignments flowing into this one. However, in the case of assign_num, there is no flow, so it is just set to the current variable assignment. We would then generate a KILL set which is every single assignment to the variable except the current assignment. This KILL set is subtracted from the abstract state of every variable, not just the one being currently being assigned. It represents these other assignments to this variable becoming inactive in the analysis.

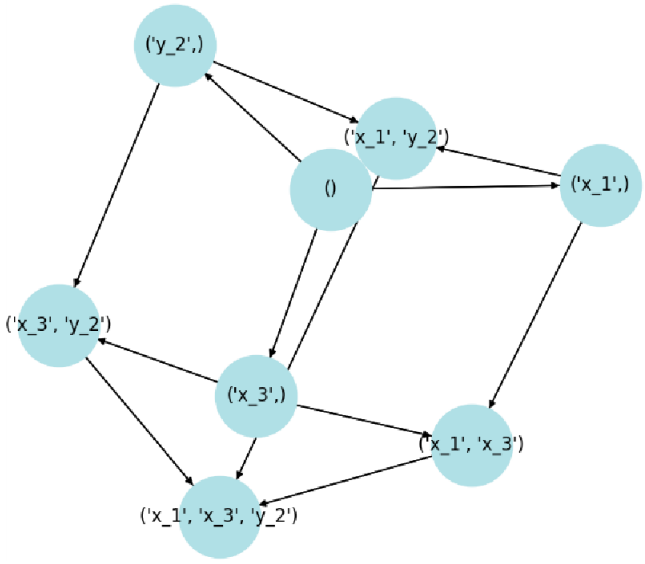

Figure 2: Reaching Definition Lattice For prog_1.w3a

2.5 Worklist Algorithm (worklist.py)

After an analysis is selected (Reaching or Signed), the analysis's domain and flow function is passed into the worklist algorithm to be computed upon. We implemented the worklist algorithm following the pseudo code provided in class and in the provided pdf textbook. It adds line nodes back to the worklist if the input to that line node was changed by its predecessor. It also uses the passed domain to join the input and output states allowing for states to be joined following the specified partial order rules.

3 Setup and Execution

The packages necessary for running the program is matplotlib and networkx (the latter is only needed if printing of the domain graph is necessary).

To execute integer sign analysis the following command is used:

python run.py signed put/file/path/here.w3a

To execute reach analysis the following command is used:

python run.py reaching put/your/file/path/here/you/silly/goose.w3a

4 Team Contributions

Angelo: int_sign_analysis.py, Guide
Emery: domain.py, reach_analysis.py, run.py, worklist.py, Guide
Jasmine: parser.py, Guide