CS 4390/5390 Fall 2025 **Franz Reyes**

Advanced Compilers

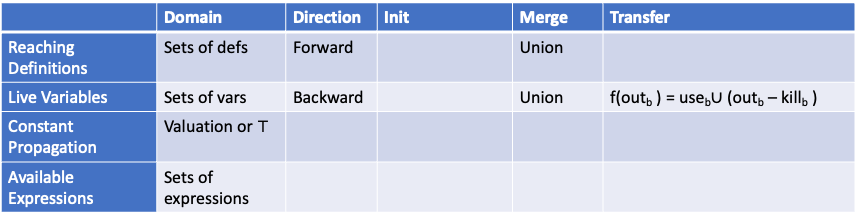
Shirley Moore, Instructor

25 points

## **Data Flow Analyses using the Worklist Algorithm**

For this assignment, you will answer a couple of questions about dataflow analysis and then implement two data flow analyses using the generic worklist algorithm. Please use the existing implementation of the worklist algorithm in examples/df.py from the Bril repository at <https://github.com/sampsyo/bril/> for tasks 3 and 4 below.

1. Fill in the table below with the necessary information for each type of analysis.



|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Domain | Direction | Init | Merge | Transfer |
| Reaching Definitions | Sets of Defs | Forward | In (entry) =∅, rest is in=∅ | Union | Outb=Genb U ( Inb -Killb) |
| Live Variables | Sets of vars | Backward | out (exit) = ∅, rest is out=∅ | Union | F(Outb)=UsebU(Outb-Killb) |
| Constant Propagation | Valuation of T | Forward | All vars start at T | Meet | Out=In, if x=const c then Out(X)=c, else if x=op(args), else Out(x)=T. |
| Available Expressions | Sets of Expressions | Forward | In(entry) =∅, rest is in=u | Intersection | Outb=Genb U ( Inb -Killb) |

1. Write a convincing argument that the worklist algorithm is guaranteed to converge to a solution, given a certain condition. Be sure to state that condition. You do not need to use lattice theory -- you may if you wish but you can also just give a convincing logical argument.

* Two conditions must be met for the worklist algorithm to complete: the set of facts must be finite, and the transfer and merge processes must be monotone. Therefore, updates will never reverse progress. Blocks with monotone updates only have one direction of information movement. The only way the sets expand in many analyses, such as attaining definitions and liveness, is by union. The sets only reduce in must analysis such as available expressions because we combine with intersection. Each variable's value only gets more precise in constant propagation, making the transition from unknown to constant or not constant. Eventually, nothing changes, the worklist empties, and you've hit a stuck point because there are only so many modifications you can make.

1. Implement reaching definitions analysis
2. Implement available expressions analysis.
3. Construct a test set and use it to thoroughly test your implementations using Turnt.

In addition to your answers to 1 and 2 above, please turn in your code files, your test cases, and a README file. The easiest way to do this is to create a github repository for the class and create a directory for this assignment with a subdirectory called test for your test cases. Your README file should include a description of the code along with usage instructions.

Grading: Each analysis question or implementation is worth 5 points. The README and test cases make up the remaining 5 points.