CSE 401 - Section 10 – Wrap-up!

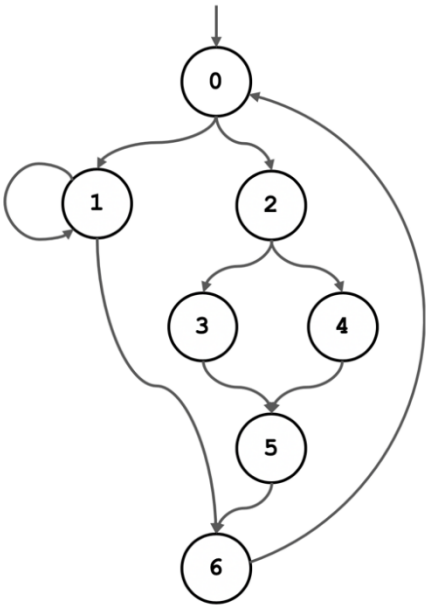
1. Dataflow Review For each of the following optimizations, list the dataflow analysis that would be most directly applicable. You may use a single dataflow analysis for multiple optimizations, or none. The possible dataflow analyses are reproduced here for reference:

Live Variable Analysis (Determining if there is any path from the use of a variable to its definition along which it is not redefined)
Available Expressions (For an expression, determining which other basic blocks are reached without redefining any of the variables in that expression)
Reaching Definitions (Determining which other basic blocks could potentially see the value of a given definition)
Very Busy Expressions (Determining if an expression is evaluated and used along every path that leaves a basic block, and if the value would be consistent in the parent basic block)

a) Constant Propagation - If a variable x is defined to be a constant in one part of the code, replace uses of the variable x with its defined constant.
b) Common Subexpression Elimination - If an expression is computed twice and will have the same value in both locations, compute it only once (Note: only applies to expressions without side effects).
c) Code Hoisting - Reducing the size of the code by factoring out duplicate code that appears in all possible paths in a part of the program.
d) Dead Store Elimination - Removing assignments to a variable if that assignment will never be used in the program.

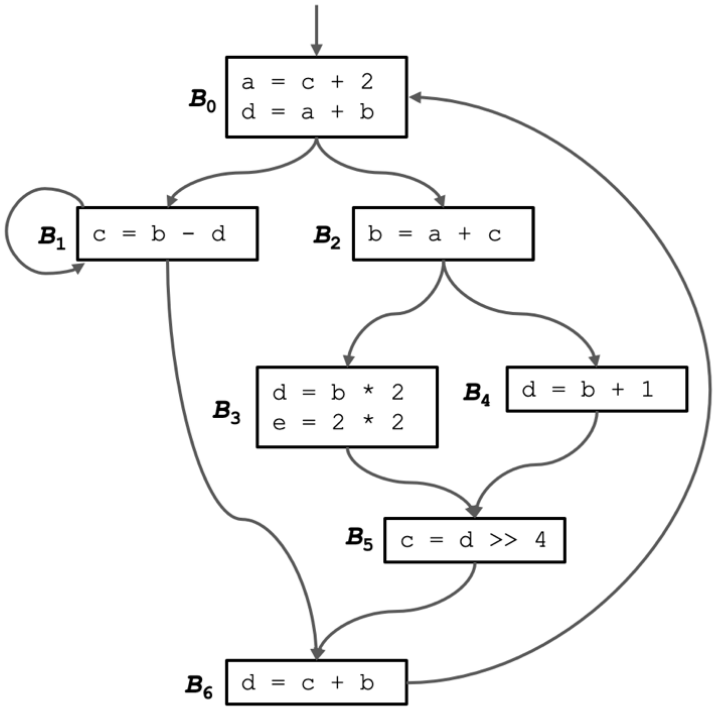
2. Single Static Assignment Conversion

a) Consider the following simplified control flow graph. For each node in the graph, fill in the table with the set of nodes that are strictly dominated by that node and the set of nodes in its dominance frontier. Recall: node **X** dominates **Y** iff every path from the CFG entry point to **Y** includes **X**. Node **X** strictly dominates **Y** iff **X** dominates **Y** and **X ≠ Y**. Finally, node **Y** is in the dominance frontier of node **X** if **X** dominates an immediate predecessor of **Y** but **X** does not strictly dominate **Y**.

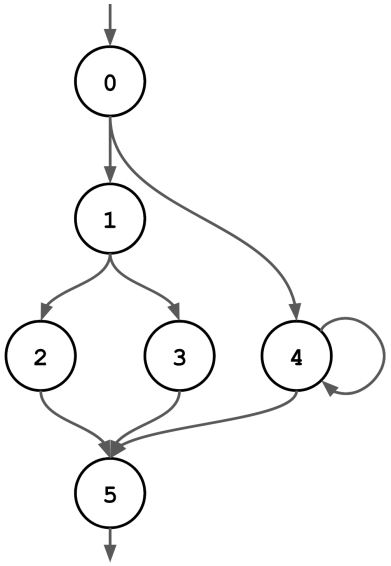


NODE	STRICTLY DOMINATES	DOMINANCE FRONTIER
0		
1		
2		
3		
4		
5		
6		

b) Now, you will complete the conversion to SSA. Suppose the control flow graph from part (a) contains the following code. Convert this code to Single Static Assignment form. Remember that you can use the dominance frontiers computed in part (a) to determine which variables need to be merged (using phi functions) in each block.



3. **Dominators and Dominance Frontiers** Consider the following simplified control flow graph. For each node in the graph, fill in the table with the set of nodes that are strictly dominated by that node and the set of nodes in its dominance frontier.



NODE	STRICTLY DOMINATES	DOMINANCE FRONTIER
0		
1		
2		
3		
4		
5		