Path Testing

CS 3750
Review

1. How are code coverage metrics used in black box testing?
2. What’s the difference between code coverage in black box and white box testing?
Create a control flow graph for the following method.

```java
public int basisExampleOne( int x, int y ) {
    int z = x + y;
    double w = z / 2.0 + 5;

    System.out.println( "Sum: " + z );

    for( int a = 1; a < z; a++, z-- ) {
        if( x > z ) {
            a = a + x;
        }
    }

    return z;
}
```
We still are interested in our other coverage metrics (e.g. statement, condition, decision, condition/decision, MCDC). We introduce another way of thinking about coverage:

- A set of paths is *linearly independent* if no path in the set is a linear combination of any other paths in the set (meaning there is at least one unique edge in each).
- A *basis set* of paths (also called *basis paths*) is a maximal linearly independent set of paths.
- The *cyclomatic complexity* is the size of the basis set and can be computed in several ways (number of regions + 1, decisions + 1, $e - n + 2$).
We can (usually) generate the basis set from a given program (or graph) by starting with a “normal” path through and flipping the decisions until we’ve covered all edges. McCabe calls this the “baseline method”.
Create a *control flow graph* for the following method.

```java
public int basisExampleTwo( int x, int y ) {
    int z = x + y;
    double w = z / 2.0 + 5;

    for( int a = 1; a < z; a++, z-- ) {
        System.out.println( w );
    }

    return z;
}
```
Finally, we’ll introduce *loop coverage*. A “typical” set of loop coverage criteria are:

- For each loop, the tested number of iterations is determined through boundary value testing (e.g. minimal, nominal, maximal)

- For nested loops, once an inner loop is tested, it is “collapsed” into a single node in our graph if possible (preventing path explosion)