Juan Bancamper
Formal Languages and Computability
31 January 2016

# Java For Loop DFA

When creating a DFA for Java's For Loop, I decided to use the divide and conquer approach. I noticed that the For Loop had a print statement embedded within its blocks. Taking into account different automatas, I made two different DFAs. The first DFA is the For Loop, and somewhere after the block of code starts, I point the viewer to another DFA for printing in Java.

When creating the DFA for the For Loop, I ran into a lot of problems with whitespace. It is very hard to determine how someone will write code, including whitespace every possible chance, or using them only when necessary. Whitespace is sometimes irrelevant to some operations, for example, using an equality operator. One programmer may write **i < 2**, whereas someone may write **i<2**. Both of these statements are correct, but using a DFA to determine that requires many different states. I solved this problem by using regular expressions to allow spaces, but sho that they are not necessary. This way, the DFA can encompass all types of programming styles, while following a certain automata.

Another problem I ran into was representing a boolean expression. In the termination section of the for loop, boolean expression can be respected by a single number, an equality expression, a method call, and many other statements that hold the value of true or false. This made me think of the DFA for a boolean expression, and how huge it must be. I overlooked this DFA, as the most common boolean expression in a for loop an equality expression of a variable coming to an upper bound. I assumed that this will always be the case and continued to another state as long as the boolean expression followed **(variable)(equality operator)(number)**.

When creating the second DFA, I realized that robust languages, not only programming languages, depend on each other for different things. In the example of the For Loop, I realized that a DFA for a print statement, or any statement in Java must be defined first for the for loop to repeat instruction. Although the for loop is dependent on the statements, the statements are dependent on being in between blocks of a function, namespace, loop, conditional, etc… Defining a new language, or even understanding the automata of it, can be very confusing.