**Fast Implementations of WalkSAT and Resolution Proving**

In this assignment, I attempted to develop a fast implementation of WalkSAT using Java programming language. Concepts like recursion, random variable generation, file input/output, List and ArrayList abstratct data types. I used a python program to generate a random k-cnf file in minisat (DIMACS) format which accounts for all the rules as stated in the first paragraph of pg. 264 in the course textbook. While developing this assignment, it was found that the implementations of WalkSAT in logic.py is quite different from what I implemented. Furthermore, it was found that Java is limited by the depth upto which its stack can be filled. That number was found to be around 5000 and so the maximum recursion depth in this assignment has been set to 5000.

A screenshot of a cell phone

Description automatically generated

As seen from the data above, the python version of WalkSAT as given in logic.py turned out to be faster that the java version. This indicated that java is not efficient enough because of its limit while recursing. Furthermore, upon manually checking, it was found out that as the value of K increased, the computation time increased for the logic.py version. Whereas, it made the java version faster.

Since I wasn’t able to do the Resolution prover, the data regarding it has not been attached. Moreover, if done in C++, the WalkSAT solver that I have implemented is much more faster. Therefore I wonder if C++ is more efficient than Java when recursion is involved. However, I chose java because of the wide array of libraries that it offers.