Comp Struct 2D Group 9
Christopher Wah Boon Chuan (1005214), Han Wei Guang (1005233), Lim Hong Jun Joshua (1005259),
Pala Tej Deep (1005282), Baarath S/O Sellathurai (1005433), Krittanat Kulsakdinun (1005609)

Approach

Using the given pseudocode in the handout, we employed the DPLL algorithm to solve the SAT problem. If given a formula without any clauses, that formula is satisfiable. If we have a clause in the formula, then we can find that smallest clause. If that clause is of unit length, we only take its literal to be TRUE if it is a positive literal and call substitute to simplify the formula. We then call solve on the simplified formula. Otherwise, we arbitrarily take a literal from the clause and set it to be TRUE and similarly call substitute and solve methods. We repeat this if there is a conflict detected for FALSE instead and repeat until there is a solution.

SATSolverTest

This program aims to do two tasks - to read the CNF file and feed it to the formula, and run SATSolver by feeding in the formula to the program. In order to do this, firstly, the pathway is fed to BufferedReader.[1] Then, each line starts to be read to find both “p” and “cnf” in the same line in the file. After that, each line is stored into an ArrayList “storedLines” and then the file is closed. The file is processed by splitting each line and adding them to the ArrayList “storedClauses”. The “0”s are then removed and a new line is added to the ArrayList “storedLines2”. Finally, each line is processed as a clause and added to the formula.

Next, the formula is fed to the SATSolver and the variables will be stored in the Environment instance “e”. If there are no variables, “e” is “null”. Otherwise, “e” is the list of variables and the “get” function is used to retrieve the boolean value of each variable. Lastly, this is all processed in a while loop such that the variable and values are printed into the file “BoolAssignment.txt”.[2]

Initially, our team used Filereader as a way to read the .cnf file.[3] The data was read character by character and then cleaned to remove the comments and get the number of variables. The variables were then sorted and placed into an array to be read and converted into strings, before storing them as clauses in formula. This increased the runtime of the program as there were a lot more loops and statements required to process the individual characters. Comparatively, after our team switched to BufferedReader, the program no longer needed to process so many different items and this shortened the runtime.

Time taken

The time taken to run the SATSolverTest using the largeSAT.cnf file is 1462.6786ms. The result was “Satisfiable” and this was done using a Zenbook UX325SA, processor with AMD Ryzen 7 5800U with Radeon Graphics at 1.9 GHz.

---

[1] JavaTPoint. “How to Read File Line by Line in Java - Javatpoint.” Accessed March 6, 2022. https://www.javatpoint.com/how-to-read-file-line-by-line-in-java.
[2] Home and Learn. “Write to a Text File in Java.” Accessed March 6, 2022. https://www.homeandlearn.co.uk/java/write_to_textfile.html.
[3] Techie Delight. “Read a Text File Using FileReader in Java.” Techie Delight, May 1, 2021. Accessed March 6, 2022.https://www.techiedelight.com/read-text-file-using-filereader-java/.