CMPT 310- Final Report

WalkSAT and Resolution Proving

* Introduction:

The project aims to implement walksat and resolution proving in both C++ and python, based on the implementation of the programs in AIMA python library. The project compares the efficiency of the implementations of both programs. The projects will try to implement an efficient version of walksat and resolution proving in C++.

* Comparing the Languages:

Pyhton is an interpreted language. Interpreted code is always slower than direct machine code. It takes a lot of time to implement an interpreted instruction machine code than to implement an actual machine code. Python is also dynamically typed language, so it makes that even slower.

* Comparing the algorithms:

C++ is much more efficient, so the implementations result in a much less runtime. The walSat in C++ uses “map” vector for clauses and “set” for unsatisfied clauses which is more efficient. The “map” function’s time complexity is O(logn), even the worst case will O(logn) as it stores the elements as binary search tree. The “set” function is also implemented as balanced tree structure, it also runs in the same time complexity.

The algorithm also stores the clauses after sorting them. It results in a faster runtime as the program knows where to insert the clause or where to delete from.

* Comparing the results:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Parameters | WalkSat(python)  (Run time) | WalkSat(C++)  (Run time) | Resolution  Prover(C++)  (Run Time) | MiniSat |
| Clauses 8000, symbol=4500 |  |  |  |  |
|  | 19.8509 | 1.364 | 1.193 | 0.0610 |
|  | 19.00 | 1.357 | 1.10 | 0.0611 |
|  | 19.2763 | 1.377 | 1.189 | 0.060 |
| Clauses  90000,  symbol=4500 |  |  |  |  |
|  | 29.94 | 3.487 | 1.453 | 0.073 |
|  | 29.2886 | 3.458 | 1.461 | 0.0723 |
|  | 29.334 | 3.51 | 1.495 | 0.069 |
|  |  |  |  |  |
|  |  |  |  |  |

Chart 1: Run time comparison for 8000 clauses and 4500 symbols in different algorithms.

It can be clearly seen from the above chart that; C++ runs much faster. It is due to the language type and the structure of the program. WalkSat is almost 10 times faster in C++ than python. Also, C++ resolution prover solves clauses very fast, whereas pyhton implementation is slow and sometimes run out of memory. MiniSat is much faster than C++, because it uses backtracking and clause learning.

* Instructions:

Python:

There are two text files named “CSP1.txt” and “CSP2.txt”. Run them by changing the names in the python program. In the Python version of the WalkSat change the “sat\_initializer” variable toward the end to try different file.

C++:

There are two text files named “CSP1.txt” and “CSP2.txt”. To run them:

1. Optimized walk SAT

* Replace file name (optional)
* Make all
* ./sat

1. Resolution proving

* Replace file name (optional)
* Make all
* ./resolution