README

P3 Theorem Prover Heuristic Strategy

# Wesley Gruenberg & Logan Barclay

# Executive Summary:

The theorem prover program is a Java program that takes in a knowledge base in conjunctive normal form (CNF) in the form of a text file using the command line. The program attempts to solve the theorem and displays the steps taken to reach a successful resolution or returns failure if the given proof is not solvable. We implemented two heuristic strategies to improve the number of steps needed to resolve a theorem when compared to a random resolution strategy.

# Unit Preference Heuristic Strategy

This heuristic strategy prefers to perform resolutions where one of the sentences is a single literal, or unit clause. The idea behind the strategy is that we are trying to produce an empty clause, so we want to use inferences that produce shorter clauses. Resolving a unit sentence with any other sentence always yields a clause that is shorter than the other clause. The unit preference strategy has shown dramatic speedups, making it possible to prove theorems that could not be handled without the strategy. True unit resolution is a restricted form of resolution in which every resolution step must involve a unit clause. The heuristic function we implemented adds a score to each clause, where lighter clauses are preferred. The score assigned to each clause is directly related to the number of predicates, and the number of parameters in each predicate. Unit clauses are treated as the lowest score and placed in a priority queue. This strategy is used as a generalization of the unit preference strategy. By adding heuristic scores to the sentences in the knowledge base, we were able to implement a random resolve by randomly assigning scores to the sentences and placing them in the priority queue. This random resolve strategy gave a us a base to test our heuristic strategies against.

# Subsumption Heuristic Strategy

# Testing

## Testing Unit Preference Strategy

We ran our tests on all of the provided test files. Most of the files were fairly simple and we weren’t able to see much of an improvement over random resolve. Tests test4.1, test4.2, test5.1 and test5.2 were complex enough that we had a noticeable improvement. While random would sometimes solve the theorem in less resolutions than our unit preference strategy, the majority of the time unit preference is the quicker solution.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **File** | **Random Res.** | **Random Time** | **Heuristic Res.** | **Heuristic Time** | **Result** |
| kb | 0 | 0.05ms | 0 | 0.23ms | Failure |
| kb2 | 100 | 503.41ms | 100 | 1066.51ms | Failure |
| kb3 | 3 | 1.26ms | 3 | 1.45ms | Success |
| test1.txt | 2 | 0.45ms | 2 | 1.67ms | Failure |
| test2.txt | 6 | 4.89ms | 5 | 2.89ms | Success |
| test3.txt | 2 | 0.65ms | 2 | 0.98ms | Success |
| test4.1 | 13 | 10.27ms | 8 | 4.79ms | Success |
| test4.2 | 21 | 40.44ms | 8 | 8.69ms | Success |
| test5.1 | 23 | 19.56ms | 13 | 13.32ms | Success |
| test5.2 | 55 | 89.98ms | 27 | 37.45ms | Success |
| test6.txt | 2 | 2.98ms | 2 | 2.83ms | Failure |

Figure 1.2 Table of results for Unit Preference