Project - Resolution Theorem Proving Experiments

Catherine Block - Fall Semester 2004

Advisor: Dr. Chris Pollett

Overview

The goal for this project was to create a Java Applet that allows the user to enter a CNF or DNF sentence and choose an algorithm to check it for satisfiability. A circuit generator was planned to generate hard circuits.

Final Status

A Java Applet was built, allowing for user entry of CNF and DNF sentences. The Applet features:

- enter CNF sentences (figure 1)
- enter DNF sentences, which are negated and converted to CNFs (figure 1)
- entered CNFs are visible to the user (figure 1)
- CNF can be selected from the store of entered CNFs and conjuncted (figure 1)
- polynomial view shows entered CNFs as vectors (figure 2)
- CNFs can be checked for satisfiability using
    - DPLL algorithm
    - brute force algorithm
- CNF's can be checked for validity

A Tau Tautology Generator was partially constructed, but not finished. In the Applet, a dialog box allows the user to set the number of rows and the l-value for the generator grid, then generate a circuit. The user can then enter this circuit as a CNF. This feature is not finished, and the entry currently consists of placeholder code. (figure 3)

Implementation Details

The implementation is divided into four packages: cnfParser, gui, tauTautology, and algorithmsAndKbase. The applet init() is in gui.Gui. See the JavaDoc for useage details.

cnfParser: the classes used to parse input. This is a modified version of a recursive decent parser.

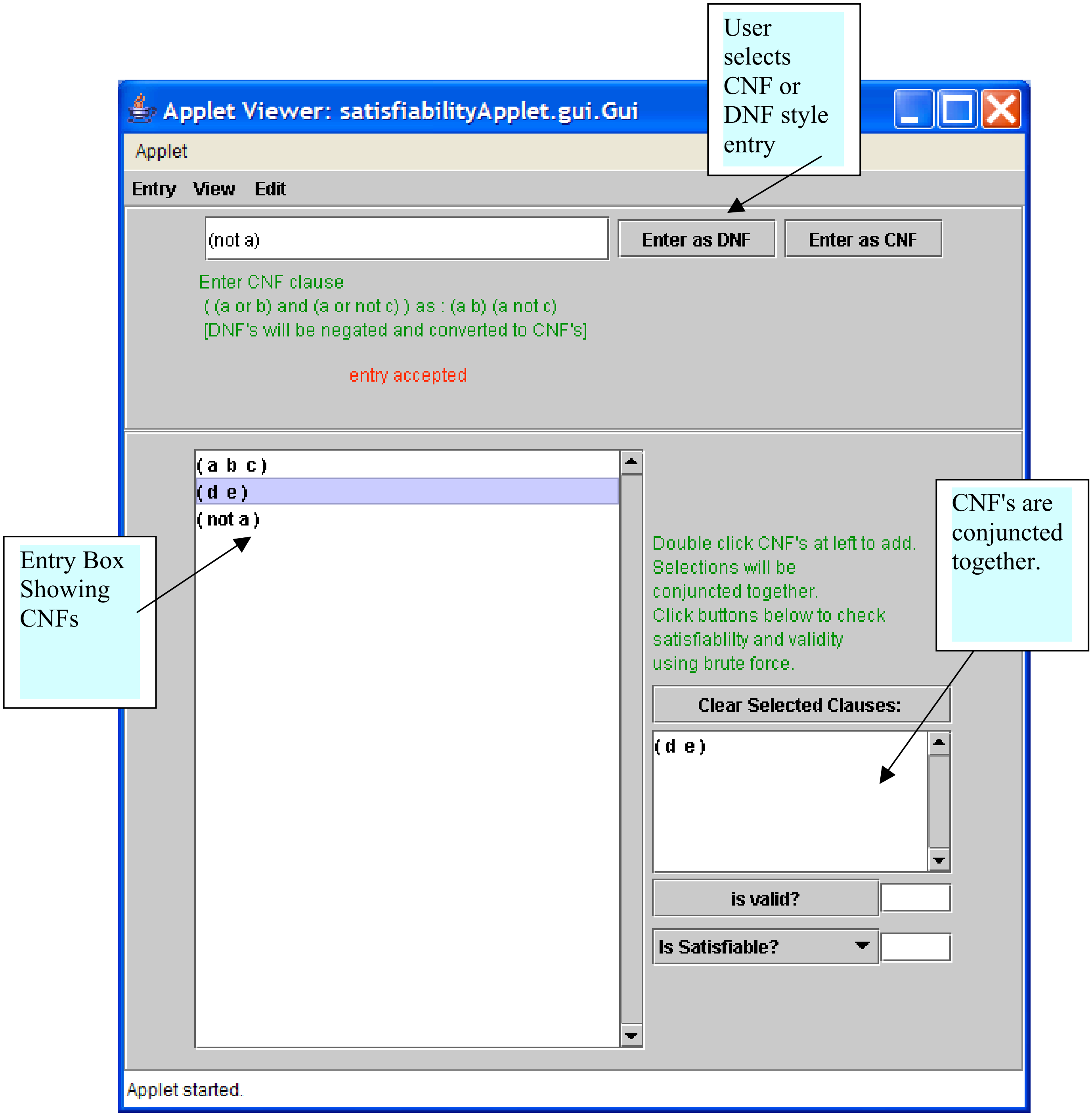
gui: user interface. Handles all user interaction. Program flow is user input driven.

algorithmsAndKbase: Satisfiability algorithms and store of entered CNFs. Uses stragegy pattern to substitute algorithms. Implemented satisfiablity algorithms are DPLL and brute force.

tauTautology: generates a circuit. Only partially implemented.

Next Steps (optional)

- finish implementation of tautology generator
- finish implementation of Groebner bases algorithm



(not a)

Enter CNF clause
((a or b) and (a or not c)) as : (a b) (a not c)
[DNF's will be negated and converted to CNF's]

entry accepted

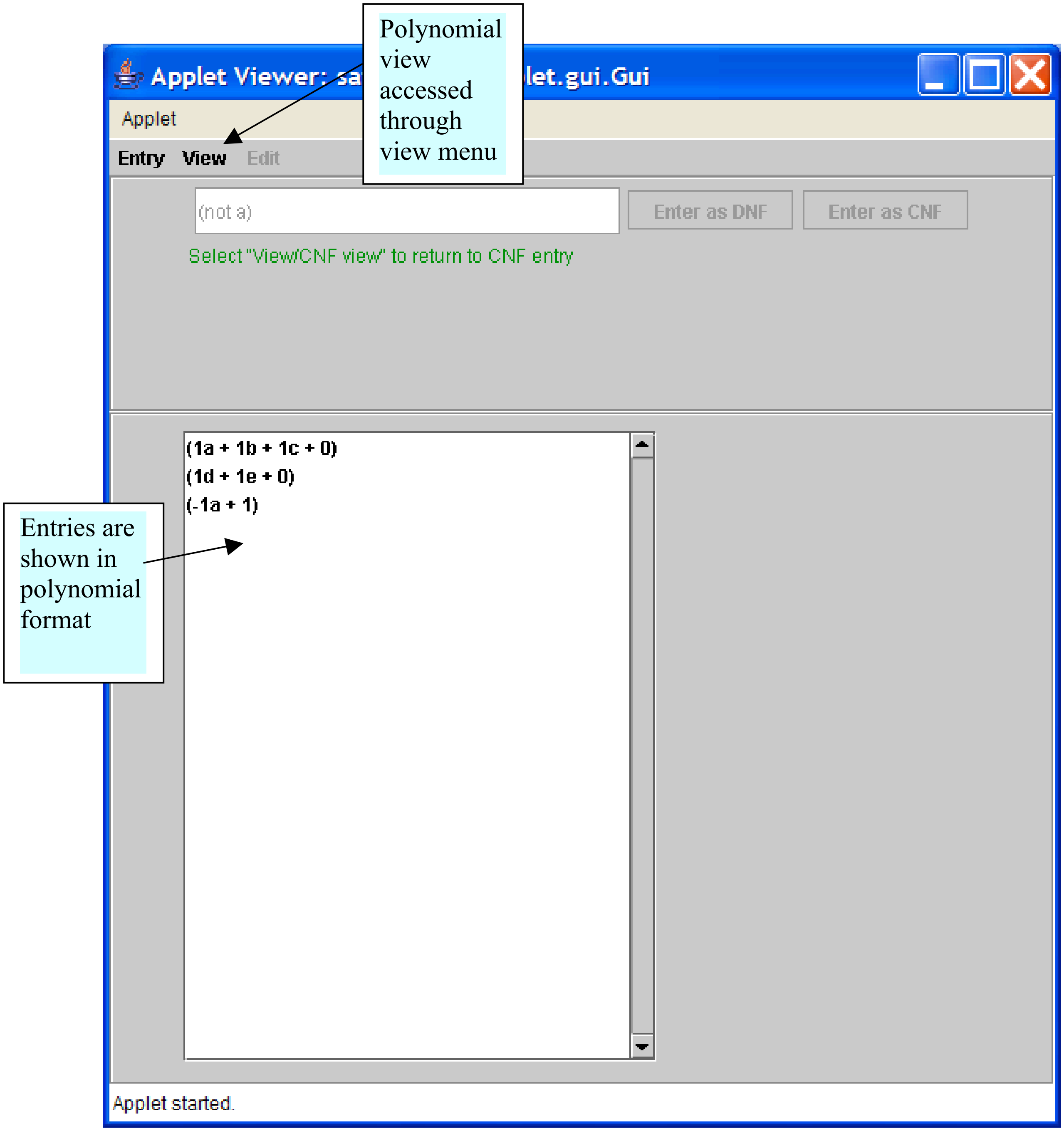
```
( a b c )
( d e )
( not a )
```

Double click CNF's at left to add. Selections will be conjuncted together.
Click buttons below to check satisfiablilty and validity using brute force.

```
( d e )
```

Applet started.

figure 1) Main Screen



(not a)

Select "View/CNF view" to return to CNF entry

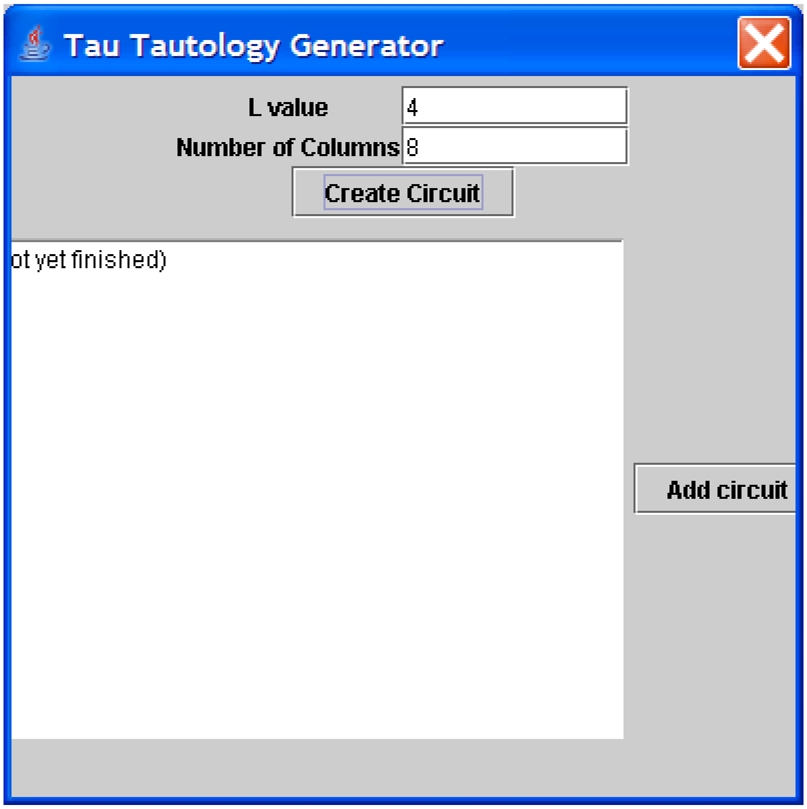
(1a + 1b + 1c + 0)
(1d + 1e + 0)
(-1a + 1)

Applet started.

Polynomial view accessed through view menu

Entries are shown in polynomial format

figure 2) Polynomial view



L value 4

Number of Columns 8

ot yet finished)

figure 3) Generator Dialog Box

References:

Clegg, M., Edmonds, J., & Impagliazzo, R. (1996, May). Using the Groebner basis algorithm to find proofs of unsatisfiability. *Proceedings of the Twenty-Eighth Annual ACM Symposium on the Theory of Computing*, 174-183, Philadelphia, Pennsylvania.

Russell, S. & Novig, P. (2003). *Artificial Intelligence: A Modern Approach* (2nd ed.). Upper Saddle River, NJ: Pearson Education Inc.

Wegener, Ingo, & Wolfgang, Johann (1987). *The Complexity of Boolean Functions*. Stuttgart: Wiley and Sons.