Automated Reasoning

- A Comprehensive Collection -

Ji, Yong-Hyeon

A document presented for
the Automated Reasoning

Department of Information Security, Cryptology, and Mathematics
College of Science and Technology
Kookmin University

April 2, 2024

Contents

Introduction

Welcome to the seminar on Automated Reasoning. This document is a compilation of various seminar materials designed to provide a comprehensive overview of the field. Here, we explore the fundamental concepts, methodologies, and applications of automated reasoning in computer science and logic.

This is an example of referencing Section

Chapter 1

Conflict Driven Clause Learning (CDCL)

1.1 The Boolean Satisfiability Problem

Propositional Variable

Definition 1.1. A **propositional variable** is an input variable (that can either be true or false) of a truth function. Propositional variables are the *basic building-blocks* of propositional formulas, used in propositional logic and higher-order logics.

Example 1.1. For a statement variable, a lowercase letter is usually used, for example: $p, q, r, \ldots,$ and so on or lowercase Greek letters, for example: ϕ, ψ, χ and so on.

Remark 1.1. The citing of a propositional variable can be interpreted as an assertion that the proposition represented by that symbol is true. That is:

"p" means "p is true".

Propositional Function (Formula)

Definition 1.2. A **propositional function** (or **formula**) $P(x_1, x_2, \ldots)$ is an operation which acts on the objects denoted by the object variables (here, propositional variables) $x_1, x_2, \ldots$ in a particular universe to return a truth value which depends on:

(1) The values of $x_1, x_2, \ldots$

(2) The nature of P.

The boolean satisfiability problem (SAT) is the following: given a formula F on propositional variables, does there exists an assignment $\mathcal{A}$ on theses variables, such that $\mathcal{A}(F) = 1$.

Given a formula F over a set of propositional variables $\{x_1, x_2, \ldots, x_n\}$,

$$\exists \mathcal{A} : \{x_1, x_2, \ldots, x_n\} \rightarrow \{0, 1\} : \mathcal{A}(F) = 1.$$

Boolean Satisfiability Problem (SAT)

Definition 1.3. Let $X = \{x_1, x_2, \cdots, x_n\}$ be a set of propositional variables.
Let L be a set of one or more propositional formulas constructed using only:

- $x_i \in X$ for $i = 1, \ldots, n$;
-

I present the CDCL algorithm and its implementation based on existing literature. This algorithm is used to solve SAT problems efficiently...

1.1.1 SAT solving basics

The CDCL algorithm is a mix of two older approaches to SAT solving: DPLL and Resolution...

Backtracking and unit propagation as in DPLL solvers

When we provide a SAT instance to a DPLL solver it builds up a search tree of assignments...

Resolution

If a formula F contains the clauses $\{\neg x\} \cup A$ and $\{x\} \cup A$...

1.2 Principles of CDCL

Now that we have seen how Backtracking and Resolution work we are ready to merge these approaches...

1.2.1 The trail

When applying CDCL rather than exploring a search tree of assignments...

1.2.2 Conflict clauses and backjumping

Consider our previous example again. When we want to continue building up our trail...

1.2.3 The implication graph

A nice way to illustrate the functionality of CDCL are implication graphs...

1.2.4 The algorithm

To get a clearer view Algorithm 4.1 shows the pseudocode for the CDCL algorithm...

1.3 Implementation

Let us now look at how the algorithm is implemented in real life...

1.3.1 Clauses

We use a monolithic array MEM to hold the original formula's clauses as well as the newly learned clauses...

1.3.2 Literals

Assume the variables are $x_1, x_2, \ldots, x_n$. We represent x_k by k...

1.4 Results

It is important to say that CDCL is a sound and complete algorithm for the propositional satisfiability problem...