

# **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 4, 2024

# Contents

- 1 Preliminaries . . . . . 4**
- 2 Conflict Driven Clause Learning (CDCL) . . . . . 8**
  - 2.0.1 SAT solving basics . . . . . 8
  - 2.1 Principles of CDCL . . . . . 9
    - 2.1.1 The trail . . . . . 9
    - 2.1.2 Conflict clauses and backjumping . . . . . 9
    - 2.1.3 The implication graph . . . . . 9
    - 2.1.4 The algorithm . . . . . 9
  - 2.2 Implementation . . . . . 9
    - 2.2.1 Clauses . . . . . 9
    - 2.2.2 Literals . . . . . 9
  - 2.3 Results . . . . . 9

## 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

## Preliminaries

### Truth Function

**Definition 1.1.** Let  $\mathbb{B} = \{T, F\}$  be the *boolean domain*. Let  $k \in \mathbb{N}$ . A mapping

$$f : \mathbb{B}^k \rightarrow \mathbb{B}$$

is called a **truth function**.

**Remark 1.1** (Truth Functions of Connectives). The logical connectives are assumed to be **truth-functional**. Hence, they are represented by certain **truth functions**.

**Logical Negation** The *logical not connective* defines the truth function  $f^\neg$  as follows:

$$\begin{aligned} f^\neg(F) &= T \\ f^\neg(T) &= F \end{aligned}$$

**Logical Conjunction** The *conjunction connective* defines the truth function  $f^\wedge$  as follows:

$$\begin{aligned} f^\wedge(T, T) &= T \\ f^\wedge(T, F) &= F \\ f^\wedge(F, T) &= F \\ f^\wedge(F, F) &= F \end{aligned}$$

**Logical Disjunction** The *disjunction connective* defines the truth function  $f^\vee$  as follows:

$$\begin{aligned} f^\vee(T, T) &= T \\ f^\vee(T, F) &= T \\ f^\vee(F, T) &= T \\ f^\vee(F, F) &= F \end{aligned}$$

### Count of Truth Functions

**Proposition 1.1.** There are  $2^{(2^k)}$  distinct truth functions on  $k$  variables.

*Proof.* Let  $f : \mathbb{B}^k \rightarrow \mathbb{B}$  be a truth function for  $k \in \mathbb{N}$ . Then

(Cardinality of Cartesian Product of Finite Sets)

$$\begin{aligned}
 \#(\mathbb{B}^k) &= \#(\overbrace{\mathbb{B} \times \cdots \times \mathbb{B}}^{k \text{ times}}) \\
 &= \overbrace{\#\mathbb{B} \# \mathbb{B} \cdots \# \mathbb{B}}^{k \text{ times}} \\
 &= \overbrace{2 \cdot 2 \cdots 2}^{k \text{ times}} \\
 &= 2^k.
 \end{aligned}$$

(Cardinality of Set of All Mappings.)

$$\#(T^S) := \{f \subseteq S \times T : f \text{ is a mapping}\} = (\#T)^{(\#S)} \implies \#(\mathbb{B}^{\mathbb{B}^k}) = 2^{(2^k)}$$

□

### Unary Truth Functions

**Corollary 1.1.1.** *There are 4 distinct unary truth functions:*

- The constant function  $f(p) = \text{F}$
- The constant function  $f(p) = \text{T}$
- The identity function  $f(p) = p$
- The logical not function  $f(p) = \neg p$

*Proof.* From Count of Truth Functions there are  $2^{(2^1)} = 4$  distinct truth functions on 1 variable. These can be depicted in a truth table as follows:

$p$	$\circ_1$	$\circ_2$	$\circ_3$	$\circ_4$
T	T	T	F	F
F	F	T	F	F

$\circ_1$ : Whether  $p = \text{T}$  or  $p = \text{F}$ ,  $\circ_1(p) = \text{T}$ . Thus  $\circ_1$  is the *constant function*  $\circ_1(p) = \text{T}$ .

$\circ_2$ : We have

$$(1) \ p = \text{T} \implies \circ_2(p) = \text{T}$$

$$(2) \ p = \text{F} \implies \circ_2(p) = \text{F}$$

Thus  $\circ_2$  is the *identity function*  $\circ_2(p) = p$ .

$\circ_3$ : We have

$$(1) \ p = \text{T} \implies \circ_3(p) = \text{F}$$

$$(2) \ p = \text{F} \implies \circ_3(p) = \text{T}$$

Thus  $\circ_3$  is the *logical not function*  $\circ_3(p) = \neg p$ .

$\circ_4$ : Whether  $p = \text{T}$  or  $p = \text{F}$ ,  $\circ_4(p) = \text{F}$ . Thus  $\circ_1$  is the *constant function*  $\circ_4(p) = \text{F}$ .

□

### Binary Truth Functions

**Corollary 1.1.2.** *There are 16 distinct unary truth functions:*

- *Two constant operations:*
  - $f_{\text{T}}(p, q) = \text{T}$
  - $f_{\text{F}}(p, q) = \text{F}$
- *Two projections:*
  - $\text{Proj}_1(p, q) = p$
  - $\text{Proj}_2(p, q) = q$
- *Two negated projections:*
  - $\overline{\text{Proj}_1}(p, q) = \neg p$
  - $\overline{\text{Proj}_2}(p, q) = \neg q$
- *The conjunction:*  $p \wedge q$
- *The disjunction:*  $p \vee q$
- *Two conditionals:*
  - $p \implies q$
  - $q \implies p$
- *The biconditional (iff):*  $p \iff q$
- *The exclusive or (xor):*  $\neg(p \iff q)$
- *Two negated conditionals:*
  - $\neg(p \implies q)$
  - $\neg(q \implies p)$
- *The NAND*  $p \uparrow q$
- *The NOR*  $p \downarrow q$

*Proof.* From Count of Truth Functions there are  $2^{(2^2)} = 16$  distinct truth functions on 2 variable. These can be depicted in a truth table as follows: □

$p$	T	T	F	F
$q$	T	F	T	F
$f_F(p, q)$	F	F	F	F
$p \downarrow q$	F	F	F	T
$\neg(p \Leftarrow q)$	F	F	T	F
$\overline{\text{Proj}_1(p, q)}$	F	F	T	T
$\neg(p \Rightarrow q)$	F	T	F	F
$\overline{\text{Proj}_2(p, q)}$	F	T	F	T
$\neg(p \iff q)$	F	T	T	F
$p \uparrow q$	F	T	T	T
$p \wedge q$	T	F	F	F
$p \iff q$	T	F	F	T
$\text{Proj}_2(p, q)$	T	F	T	F
$p \Rightarrow q$	T	F	T	T
$\text{Proj}_1(p, q)$	T	T	F	F
$p \Leftarrow q$	T	T	F	T
$p \vee q$	T	T	T	F
$f_T(p, q)$	T	T	T	T

## Chapter 2

# Conflict Driven Clause Learning (CDCL)

### Propositional Function (Formula)

**Definition 2.1.** A **propositional function** (or **formula**) is defined inductively as follows:

(Basic Step) Any propositional variable  $x \in V$  is a formula.

(Inductive Step) If  $F$  and  $G$  are formulas, then the following are also formulas:

- $\neg F$
- $F \wedge G$
- $F \vee G$
- $F \rightarrow G$
- $F \leftarrow G$

Formally, the set of all  $\Phi$  is the smallest set satisfying:

$$\Phi = V \cup \{\neg F : F \in \Phi\} \cup \{\circ(F, G) : F, G \in \Phi \text{ and } \circ \in \{\wedge, \vee, \rightarrow, \leftarrow\}\}$$

### Boolean Satisfiability Problem (SAT)

**Definition 2.2.** Let  $X = \{x_1, x_2, \dots, x_n\}$  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  $i = 1, \dots, n$ ;
- the  $2^{(2^1)} = 4$  unary logical connectives;
- the  $2^{(2^2)} = 16$  binary logical connectives;

The problem is to find truth values for all  $x \in X$  such that all the formulas in  $L$  are true.  
Such a problem is a boolean satisfiability problem.

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

### 2.0.1 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  $\{x\} \cup A...$

## 2.1 Principles of CDCL

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

### 2.1.1 The trail

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

### 2.1.2 Conflict clauses and backjumping

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

### 2.1.3 The implication graph

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

### 2.1.4 The algorithm

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

## 2.2 Implementation

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

### 2.2.1 Clauses

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

### 2.2.2 Literals

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

## 2.3 Results

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

# Bibliography

- [1] Schlenga, Alexander T. (2020). "Conflict Driven Clause Learning". June 8, 2020.
- [2] ProofWiki. "Definition: Truth Function." Accessed on [April 3, 2024]. [https://proofwiki.org/wiki/Definition:Truth\\_Function](https://proofwiki.org/wiki/Definition:Truth_Function).
- [3] ProofWiki. "Unary Truth Functions" Accessed on [April 3, 2024]. [https://proofwiki.org/wiki/Unary\\_Truth\\_Functions](https://proofwiki.org/wiki/Unary_Truth_Functions).
- [4] ProofWiki. "Binary Truth Functions" Accessed on [April 3, 2024]. [https://proofwiki.org/wiki/Binary\\_Truth\\_Functions](https://proofwiki.org/wiki/Binary_Truth_Functions).