Lectures Notes for Discrete Maths - Kenneth Rosen

Emulie Chhor

May 15, 2021

Introduction

This is my lecture notes from Discrete Maths by Kenneth Rosen accompanied by Kimberly Brehm's Discrete Maths Lectures.

1. Foundations of Logic and Proofs
2. Basic Structures: Set, Functions. Sequences, Sums
3. Algorithms
4. Number Theory and Cryptography
5. Induction and Recursion
6. Counting
7. Discrete Maths
8. Advanced Counting Techniques
9. Relations
10. Graphs
11. Trees
12. Boolean Algebra

1 Foundations of Logic and Proofs

1.1 Propositional Logic

1.1.1 Overview

1. What is a proposition: atomic vs compound statements
2. Logical Operators: and, or, not

3. Conditional Statement: implication, biconditional
4. Converse, Contrapositive, Inverse

1.1.2 What is a proposition: atomic vs compound statements

La première notion à comprendre est la notion de proposition. Une proposition est une phrase qui est associée à une valeur de vérité. Elle peut être soit vraie, soit fausse, mais par les 2.

Il existe 2 types de proposition:

1. Proposition atomique: Une seule idée
2. Compound Propostion: Plusieurs idées reliées par un connecteur logique

1.1.3 Logical Operators: and, or, not

Pour générer des des propositions atomiques, on utilise des connecteurs logiques. On distingue 3 principaux connecteurs logiques:

1. and
2. not
3. or

Celles-ci possèdent leur propre table de vérité, qui nous dit la valeur de vérité du compound proposition.

1.1.4 Conditional Statement: implication, biconditional

Il existe des connecteurs logiques qui montrent une relation entre 2 idées, qui sont aussi associées à leur table de vérité.

1. implication
2. biconditionnelle

1.1.5 Converse, Contrapositive, Inverse

Finalement, on voit qu'on peut construire différentes propositions à partir de l'implication:

1. converse: $q \to p$
2. inverse: $\not{p} \to \not{q}$
3. contrapositive: $\not{q} \to \not{p}$

Il faut retenir que la contraposée est équivalente à l'implication, un fait très utile à connaitre lorsqu'on lit des théorèmes.

1.1.6 Problèmes

1. Déterminer si la phrase est une proposition logique et dire si elle est atomique ou compound
2. Trouver la négation de la phrase
3. Déterminer la valeur propositionnelle du statement
4. Traduire la proposition en anglais
5. Traduire la proposition en symboles
6. Déterminer si les conditionnelles et biconditionnelles sont vraies ou non
7. Déterminer si le ou est includif ou exclusif
8. Traduire la proposition en "if p, then q"
9. Trouver l'inverse, la contraposée et la converse de la proposition
10. Construire la table de vérité pour les propositions
11. Find bitwise OR, AND, XOR of the binary strings

1.2 Applications of Propositional Logic

1.2.1 Overview

1. Translating English Sentence into Logical Sentences
2. Translating Logical Sentences into English Sentences
3. Puzzle: which tells the truth
4. Logical Circuits

1.2.2 Translating English Sentence into Logical Sentences

On veut traduire de sentence en logical statements pour simplifier la notation et mieux trouver sa contraposée, son inverse et sa converse. Ce processus est utile lorsqu'on lit des théorèmes

1.2.3 Translating Logical Sentences into English Sentences

Être capable de traduire des logical statements en phrases parlées peut nous donner une meilleure intuition sur les théorèmes, car ça nous permet de verbaliser les maths

1.2.4 Puzzle: which tells the truth

On a certaines situations et on doit déterminer qui ment ou pas:

1. Find Pirate Chess with treasure

2. Knights and Knaves: find which is which
3. Kids and Muds: find who has the mud

La stratégie est de partir en assumant que A dit la vérité et de voir si on a une contradiction. S'il y a contradiction, A ment nécéssairement. On peut répéter le processus.

1.2.5 Logical Circuits

Les tables de vérité nous permmettent de réduire les circuits logiques à l'aide de l'algèbre booléenne. On associe chaque opéarateur logiques à un logic gates, qu'on peut relier. Essentiellement, c'est une façon visuelle de représenter une fonction propositionnelle

1.3 Propositional Equivalences

Le but de cette section est de montrer que deux propositions sont logiquement équivalentes. Pour ce faire, on doit montrer que celles-ci ont la même table de vérité. Dans ce cas, on dit qu'il y a tautologie. Si toutes les valeurs propositionnelles mènent donnent F, on dit qu'il y a contradiction. Finalement, si les proposition peuvent être soit T, soit F, on dit que c'est une contingence. On note l'équivalence $A \equiv B$

1.3.1 Equivalence Propositionnelle

Notons qu'on peut voir l'équivalence comme une biconditionnelle. Ainsi, si on veut montrer que deux propositions sont équivalentes, on doit montrer que qu'elles ont la même table de valeur (on montre l'implication et l'inverse)

1.3.2 Logical Equivalence Table

De plus, il est important de noter les tableaux d'équivalence suivantes

1. Common Logical Equivalence Table
2. Conditional Statement
3. Biconditionnal Statement

Proposition 1.1 (Logical Equivalence)**.** *Let p,q,r be propositional variables. Here is a list of propositional equivalence*

1. *Law of double negation:* $p \equiv \neg(\neg p)$
2. *De Morgan's laws:*
   - $\neg(p \wedge q) \equiv \neg p \vee \neg q$
   - $\neg(p \vee q) \equiv \neg p \wedge \neg q$

3. *Commutative laws:*
   - $p \wedge q \equiv q \wedge p$
   - $p \vee q \equiv q \vee p$
4. *Associative laws:*
   - $(p \wedge q) \wedge r \equiv p \wedge (q \wedge r)$
   - $(p \vee q) \vee r \equiv p \vee (q \vee r)$
5. *Distributive laws:*
   - $p \wedge (q \vee r) \equiv (p \wedge q) \vee (p \wedge r)$
   - $p \vee (q \wedge r) \equiv (p \vee q) \wedge (p \vee r)$
6. *Identity laws:*
   - $p \wedge \mathrm{T} \equiv p$
   - $p \wedge \mathrm{F} \equiv \mathrm{F}$
   - $p \vee \mathrm{T} \equiv \mathrm{T}$
   - $p \vee \mathrm{F} \equiv p$
7. *Negation laws:*
   - $p \wedge \neg p \equiv \mathrm{F}$
   - $p \vee \neg p \equiv \mathrm{T}$
8. *Redundancy laws:*
   - $p \wedge p \equiv p$
   - $p \vee p \equiv p$
9. *Absorption laws:*
   - $(p \vee q) \wedge p \equiv p$
   - $(p \wedge q) \vee p \equiv p$
10. *Law of the conditional:* $p \to q \equiv \neg p \vee q$
11. *Law of contrapositive:* $p \to q = \neg q \to \neg p$
12. *Biconditional:* $p \leftrightarrow q \equiv (p \to q) \wedge (q \to p)$

Finalement, on peut utiliser les équivalence propositionnelle pour déterminer si un programme est satifsfiable ou non, c-à-d qu'il possède une solution ou non

1.3.3 Preconditions vs postconditions

La logique propositionnelle nous permet de décrire the correctness of a program:

1. preconditions: conditions que le input doit satisfier
2. postconditions: conditions que la sortie doit satisifier pour que la sortie soit valide

1.4 Predicates and Quantifiers

Malheureusement, on ne peut pas représenter toutes les expressions logiques qu'avec les connecteurs logiques. On doit donc introduire la notion de quantificateurs. On dénote plusieurs quantificateurs, mais on s'attardera au plus utilisés:

1. quantificateur universel: $\forall$
2. quantificateur existentiel: $\exists$
3. quantificateur unique: $\exists!$

Il est important de comprendre que les quantificateurs peuvent s'associer ou non à des variables. On dit qu'elles sont libres ou liées. Des variables libres peuvent prendre nimporte qu'elle valeur et quand même satisfier la proposition logique, mais les variables liés doivent respecter leur quantificateur.

Remarque. *Il est utile de voir que les quantificateurs ne sont que des while loop:*

1. *forall: on itère jusqu'à la fin, mais on arrête si on trouve un faux*
2. *exists: on itère jusqu'à la fin, et on arrête si on trouve un vrai*

1.4.1 Problems

1. Show Logical Equivalence of propositions with quantifiers: utiliser la logique pour montrer l'implication et l'inverse
2. Negating quantifiers using DeMorgan's Laws: forall becomes exists and vice-versa
3. Translating English to Logical Expression and vice versa

1.5 Nested Quantifiers

1. Translating Nested Quantifiers to English and vice-versa
2. Negating Nested Quantifiers

1.6 Rules of Inference

Dans cette section, on veut apprendre à utiliser la logique propositionnelle pour obtenir des conclusions, ce qu'on ne connait pas apriori, en utilisant les prémisses, ce qu'on connait apriori. On note plusieurs règles d'inférence:

Proposition 1.2 (Rules of Inference)**.**

1. *Modus Ponens:* $(p \to q) \wedge p \Longrightarrow q$
2. *Modus Tollens:* $(p \to q) \wedge \neg q \Longrightarrow \neg p$
3. *Disjunctive Syllogism:* $(p \vee q) \wedge \neg p \Longrightarrow q$
4. *Hypothetical Syllogism:* $(p \to q) \wedge (q \to r) \Longrightarrow (p \to r)$
5. *Conjunctive Introduction:* $(p \wedge q) \Longrightarrow p \wedge q$
6. *Disjunctive Introduction:* $p \Longrightarrow p \vee q$
7. *Biconditional Introduction:* $(p \to q) \wedge (q \to p) \Longrightarrow (p \leftrightarrow q)$
8. *Constructive Dilemma:* $(p \to q) \wedge (r \to s) \wedge (p \vee r) \Longrightarrow (q \vee s)$
9. *Destructive Dilemma:* $(p \to q) \wedge (r \to s) \wedge (\neg q \vee \neq p) \Longrightarrow (\neq p \vee \neg r)$
10. *Disjunctive Resolution:* $(p \vee q) \wedge (\neg p \vee r) \Longrightarrow (q \vee r)$
11. *Conjunctive Elimination:* $(p \wedge q) \Longrightarrow p$
12. *Biconditional Elimination:* $(p \leftrightarrow q) \Longrightarrow (p \to q)$

1.7 Introduction to Proofs

We saw three types of proofs

1. Direct Proof: $p \to q$
2. Contrapositive: $\not{q} \to \not{p} \Longrightarrow p \to q$
3. Contrary: $p \to \not{q} \to false \Longrightarrow p \to q$

More:

1. Vacuous Proof: if p is false, then $p \to q$ is always true
2. Trivial Proof

1.8 Proof Methods and Strategy

1. Proof by exhaustion: proves all cases (ex: for integers, $n = 0, n \leq 1, n \geq 1$)

TODO