CSE 2353 SPRING 2008
DISCRETE COMPUTATIONAL STRUCTURES

Professor Margaret H. Dunham
Department of Computer Science and Engineering
Southern Methodist University
Dallas, Texas 75275
(214) 768-3087
fax: (214) 768-3085
email: mhd@seas.smu.edu
www: http://www.seas.smu.edu/∼mhd

January 15, 2008

CSE2353 SPRING 2008 OUTLINE

- PART ONE
  - I. Introduction
  - II. Sets
  - III. Logic
- PART TWO
  - IV Proof Techniquess
  - V. Counting Principles & Combinatorics
  - VI. Relations & Functions
- PART THREE
  - VII. Graphs & Trees
  - VIII. Boolean Algebra

Note: All book references are to *Discrete Mathematics with Proof* by Eric Gossett

CSE2353 - INTRODUCTION

- What is Discrete?
- "Consisting of distinct or unconnected elements." Webster's Seventh New Collegiate Dictionary
- Course objectives: Study discrete math concepts which are the basis of programming and many Computer Science topics
- Will relate each topic to Computer Science

Why discrete?

CSE2353 - SETS OUTLINE

- Basic Definitions
- Venn Diagrams
- Set Operations
- Power Set
- Relationship to Computer Science

SET DEFINITIONS

- Well defined collection of distinct objects.
- Set $\{ ... \}$
- Element of $\in$
- Cardinality $| ... |$
- Subset $\subset, \subseteq$
- Equality p16

DEFINING A SET

- Enumerate (Roster Method)
- Example: A={1,2,3}
- Identify set membership (Set Builder)
- Example: A=$\{x \mid x \in Z$ and $x \geq 1$ and $x \leq 3\}$

SPECIAL SETS

- Null (Empty) Set - $\emptyset$
- Universal Set - U
- Natural Numbers - **N**
- Integers - **Z**
- Positive Integers - Z^+
- Negative Integers - Z^-
- Rational - **Q**
- Real - **R**
- Complex - **C**
- Power - **P**(A)

If $|A|$=n, what is $|\mathbf{P(A)}|$?

SET OPERATIONS

- Intersection $\cap$
- Union $\cup$
- Difference -
- Complement $\overline{A}$
- We can represent these using Venn Diagrams
- Cartesian Product $\times$
- Laws on p 27

Venn Diagram Worksheet

SETS IN COMPUTER SCIENCE

- Representation:
  - Linked List
  - Array
  - Bit Map
- Multiset (Removing duplicates is sometimes difficult)
- Ordered Set
- File
  - Subset of Cartesian Product
  - Operations can be viewed as set operations

CSE2353 - LOGIC OUTLINE

- Propositions
- Truth Tables
- Predicates
- Use in CS

WHAT IS A PROPOSITION?

- Proposition - Statement which is true or false.
- Examples
  - There are planets circling Apha Centauri.
  - 3 is an even
- Propositional Calculus - Notation and rules for combining and manipulating propositions.
- Notation:
  - Proposition: p,q,r,s, ... (Small or Capital)
  - Operations:
    - and (conjunction) $\wedge$
    - or (inclusive disjunction) $\vee$
    - not (negation) $\neg$ or $\sim$ or $\overline{p}$
    - if ... then (implication) $\rightarrow$ or $\supset$
    - if and only if (biconditional) $\leftrightarrow$
  - Precedence: $\neg \wedge \vee \rightarrow \leftrightarrow$

TRUTH TABLES

- Truth Tables - Summarizes truth values of propositions.
- Column for each proposition (simple and compound)
- Row for each possible combination of truth values for simple propositions
- See tables on p34, p35

Examples: $\overline{p}$**,** $p \wedge q$**,** $p \vee q$**, ...**

IMPLICATION

- Examples
  - If 3 is even then 4 is odd.
  - If I get an A in this course I'll celebrate by going to Europe
- p→q
- p - antecedent (hypothesis)
- q - consequent (conclusion)
- Converse of p→q is q→p
- Inverse of p→q is $\overline{p} \rightarrow \overline{q}$
- Contrapositive of p→q is $\overline{q} \rightarrow \overline{p}$

Truth Table for Implication

LOGICAL EXPRESSIONS

- Well Formed Formula:
  - $\langle prop \rangle$ ::= p | q | ... | z
  - $\langle binop \rangle$ ::= $\rightarrow \mid \wedge \mid \vee \mid \leftrightarrow$
  - $\langle wff \rangle ::= \langle prop \rangle \mid (\overline{\langle wff \rangle}) \mid (\langle wff \rangle \langle binop \rangle \langle wff \rangle)$
- Note: Parentheses may be dropped if order of execution follows precedence order

How many different logical expressions are there on two propositions?

OTHER OPERATIONS

- xor $\oplus$ ($\veebar$)
- nand $|$
- nor $\downarrow$
- tautology (denoted by t or T)
- contradiction (denoted by f or F)

LOGIC WORKSHEET

EQUIVALENCE and INFERENCE

- Two propositions are logically equivalent (p$\equiv$q) or (p$\Leftrightarrow$q)if they have the same truth values (p$\leftrightarrow$q is T).
- A proposition logically implies (infers) another proposition (p$\vdash$q) or (p $\Rightarrow$ q) if whenever p is true then q is also true (Note that p$\rightarrow$q is T.)

How to show equivalence?

How to show inferences?

INFERENCE RULES FOR PROPOSITIONAL CALCULUS

- Inference 1: (Modus Ponens)
  - From p→q and p, infer q.
- Inference 2: (Substitution)
  - A propositional variable may be substituted by another variable.
- With substitution must be careful to substitute all occurrences of a propositional variable.
- Substitution p45
- Tables 2.21, 2.22, 2.23 p47-48

Equivalence/Logical Inference Worksheet

ARGUMENT

- Argument (Claim)
  - Set of propositions (premises) and another proposition (conclusion).
  - The argument is valid if the conjunction of the premises logically implies the conclusion. Else it is invalid.
- Notation
  - $A_1, A_2, ..., A_{n-1} \Rightarrow A_n$
  - $A_1, A_2, ..., A_{n-1}$ are the premises
  - A_n is the conclusion
  - This is a valid argument if $A_1 \wedge A_2 \wedge ... \wedge A_{n-1} \rightarrow A_n$ is T
- Section 2.7.1 p69-72

SYLLOGISMS

- Techniques to infer new true statements from existing statements
- Major premise, Minor premise, Conclusions
- Syllogisms p14
- Valid arguments depend on the application of syllogisms
- http://en.wikipedia.org/wiki/Syllogisms
- Disjunctive Syllogism
- Hypothetical Syllogism

Valid Argument Worksheet

WHAT IS A PREDICATE?

- Not all statements are propositions
  - 2x is even
  - x+y=4
- However these can be expanded to become propositions
- Predicate - property describing an object.
- Predicates are denoted by capital letters: E, F
- Predicate Form (Propositional Function) - Predicate with variables: E(x), F(x,y)
- A predicate form becomes a proposition when variables are replaced with elements from a specific domain: E(2), F(2,3)

Can you determine the truth value of a predicate. Why?

What do you have to do to convert a predicate to a proposition?

QUANTIFIERS

- Can also convert a propositional function into a proposition by using quantifiers, $\forall$, $\exists$:
  - $\forall x$, E(x)
  - $\exists x, \exists y$, F(x,y)
- Examples
  - Computer Science students are brilliant: B(x)
  - All computer science students are brilliant: $\forall$x, B(x)
  - Some computer science students are brilliant: $\exists$x, B(x)

Quantifier Worksheet

ARGUMENTS IN PREDICATE LOGIC

- Claims with quantifiers, p73
- Universal Specification: If $\forall$x, F(x) is true then F(a) is true for all a in the universe of discourse
- Universal Generalization: If F(a) is true for all a in the universe of discourse then $\forall$x, F(x) is true.
- Existential Specification: If $\exists$x, F(x) is true then there is at least one a in the universe of discourse such that F(a) is true
- Existential Generalization: If F(a) is true for some a in the universe of discourse then $\exists$x, F(x) is true.

Inference Rules Handout

Examples

RELATIONSHIP OF LOGIC TO PROGRAMMING

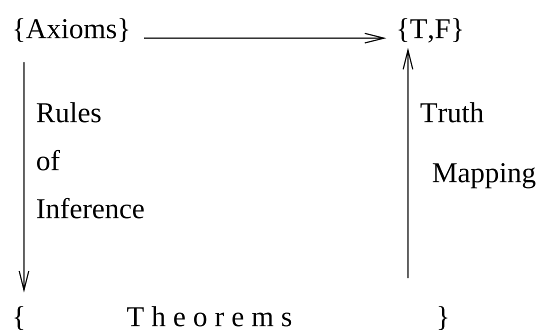
- Conditional statements
- Variables
- Looping
- Database query languages

CSE2353 - PROOF TECHNIQUES OUTLINE

- What is a proof?
- Proof Techniques
- Examples - practice
- Relationship to Computer Science

AXIOM SYSTEM

- Axiom - Statement which is assumed. Given as a starting point.
- Inference Rules - Rules describing how to generate new statements from existing ones.
- Theorem - Statement derivable from a set of axioms using specified rules of inference.
- Proof - Finite sequence of statments where each is either an axiom or theorem previously derived.



PROVING A THEOREM

- A theorem is a statement which is logically implied by the conjunction of axioms.
- A theorem is a proposition T, such that
  - $(A1 \wedge A2 \wedge ... \wedge An) \vdash$ T
- Assume the truth of the axioms and show this guarantees the truth of T
- $P \Rightarrow Q$ or P implies Q. This means that Q follows from P. Or, the conjunction of P and all axioms logically implies Q.

METHODS OF PROOF

- Vacuous
- Trivial
- Direct
- Contrapositive
- Contradiction
- Induction
- YOU CAN NOT PROVE BY EXAMPLE
- Counter Example
- Divide into Cases
- Construction

IMPLICATION "PROOFS"

p	q	p→q
F	F	T
F	T	T
T	F	F
T	T	T

- Vacuous - Only rows 1 and 2 occur. Show p is false
- Trivial - Only rows 2 and 4 occur. Show q is true
- Direct - Only rows 1,2,4 occur
- Indirect/Contrapositive - Prove $\neg q \rightarrow \neg p$.
- Contradiction - Show $\neg(p \wedge \neg q)$ is a tautology.
- These show the existence of a proof but are not technically a proof.

Proof Handout

EXAMPLE PROOFS

Prove: If $x \in Z$ is even then x+1 is odd.

Prove: There is no largest prime number.

Prove: If A and B are finite sets, then $A \cup B$ and $A \cap B$ are also finite.

Direct Proofs Worksheet

Indirect Proofs Worksheet

Contradiction Proofs Worksheet

PROOF BY INDUCTION

- Induction allows us to prove a property about an infinite (or a large) set of objects without having to look at all the objects.
- Ex: I can climb to the top of the Empire State Building.
- Technique:
  - Prove conjecture holds for n=1. (Basis/Base)
  - Prove that for all k $\geq$ 1, if the result holds for n=k, then it must also hold for n=k+1 (Inductive step: Hypothesis and Induction).

Prove

$$\sum_{i=1}^{n} i = \frac{n(n+1)}{2}$$

Proof by Induction Worksheet

General Proofs Worksheet

CSE2353 COUNTING PRINCIPLES & COMBINATORICS OUTLINE

- Basic Counting Principles Addition and Multiplication
- Pigeonhole Principle
- Permutations and Combinations

Addition & Multiplication Principles

- Suppose tasks $T_1, T_2, \cdots, T_n$ can be done in $x_1, X_2, \cdots, x_n$ ways.
- Addition Principle Number of ways to do one
  - The number of ways any one of the tasks can be done is $x_1 + x_2 + \cdots + x_n$.
  - Example: Suppose a menu has three different desserts: cake, ice cream and cookies. If there are 5 types of ice cream, 3 types of cakes, and 10 types of cookies, how many actual choices does one have to choose a dessert?
- Multiplication Principle Number of ways to do all
  - The number of ways any all of the tasks can be done is $x_1 \cdot x_2 \cdot \cdots \cdot x_n$.
  - Example: How many ways could one order one of each type of dessert?

Counting Principles Worksheet

Pigeonhole Principle

- Suppose there are n pigeons and k pigeonholes where $n > k$. If all these pigeons try to fit into the pigeonholes, then some pigeonholes will have to contain at least two pigeons.
- Generalized Pigeonhole Principle: n pigeons, k pigeonholes, where m = $\lceil \frac{n}{k} \rceil$. Some pigeonhole must contain at least m pigeons.
- Example: Suppose a room has 25 people in it? Then at least 3 must have a birthday in the same month.

Pigeonhole Principle Worksheet

Permutaions and Combinations

- Permutations: Number of ways to arrange r objects from a set of n objects in order (without repetition is: $P(n, r) = \frac{n!}{(n-r)!}$
- Combinations Number of ways to select r objects from a set of n objects (order independent, no repetition): $C(n, r) = \frac{n!}{(n-r)! \cdot r!}$
- How many different combinations of desserts could a table of 4 have? (Use the dessert choices mentioned earlier.)

Permutations and Combinations Worksheet

CSE2353 - RELATIONS & FUNCTIONS OUTLINE

- What is a relation?
- Equivalence relation
- Partial order
- Relations in Computer Science
- What is a function?
- Function operations
- Functions in Computer Science

RELATION DEFINITIONS

- Relation
  - bf R⊆A×B.
  - x**R**y iff (x,y)∈**R**.
  - A is the domaine of the relation
  - B is the range of the relation
  - NOTE: Some definitions of range/domain only refer to the subset of B/A actually involved in the relation
- Tuple - element of a relation.
- Relation on a set A, is a subset of A×A.
- Inverse Relations - x$\mathbf{R}^{-1}$y iff y**R**x.

REPRESENTING BINARY RELATIONS

- Set - ordered pairs
- Set definition - membership values
- Arrow Diagram
  - Write elements of set A in one column and elements of B is another
  - Draw an arrow from each element in A column to each element in B column with which it is related
- Digraph (Directed Graph) for relation on set A
  - Create a vertex for each element in A and label
  - Draw an arrow from each vertex in A to each vertex with which it is related

Relation Worksheet

PROPERTIES OF RELATIONS

- Let **R** be a relation on a set A:
  - Reflexive:$\forall$ a$\in$A a**R**a.
  - Symmetric: $\forall$ a$\in$A if a**R**b then b**R**a.
  - Antisymmetric: $\forall$ a$\in$A if a**R**b and b**R**a then a=b.
  - Transitive: $\forall$ a$\in$A if a**R**b and b**R**c then a**R**c.
  - Irreflexive: $\neg\exists a \in A$ where a**R**a.
- Examples:
  - Reflexive: $\subseteq$, $\leq$, $=$.
  - Symmetric: $=$.
  - Antisymmetric: $\subseteq$, $\leq$, $=$, $<$
  - Transitive: $\subseteq$, $\leq$, $=$.
  - Irreflexive: $<$, $>$.
- The closure of a relation **R**, is the smallest superset of R which satisfies a particular property.

Relation Properties Worksheet

EQUIVALENCE RELATIONS

- Equivalence - reflexive, symmetric, transitive
  - Examples: =, mod n
  - Partition of a set S is a collection of subsets $A_1, A_2, ..., A_n, ...$ such that $A_1 \cup A_2 \cup ... \cup A_n \cup ... =$ S and if i$\neq$j then $A_i \cap A_j = \emptyset$.
  - An equivalence relation on a set S defines a set of equivalence classes which partition S.
  - The equivalence class of x$\in$A is the set of elements of A to which x is related: $[x]_R$=$\{$y$\in$A$|$x**R**y$\}$.

PARTIAL ORDER RELATIONS

- Partial Order - reflexive, antisymmetric, transitive
  - Examples: $\leq$, $\subseteq$
  - Linear Order (Total Order) - partial order where any two sets are related
  - Poset - set with a partial order.
  - Hasse Diagram - technique of representing a finite poset.

MORE ON POSETS

- Given A,**R** which is a partial order:
  - Greatest element of A: $\alpha \in A$ such that ∀ a∈A a**R**α
  - Least element of A: $\beta \in A$ such that ∀ a∈A β**R**a
  - Poset need not have a greatest or least element
  - Maximal element of A: x∈ A if ∀ a∈A x**R**a implies x=a
  - Minimal element of A: y∈ A if ∀ a∈A a**R**y implies a=y

Poset and Equivalence Relations Worksheet

RELATIONS IN COMPUTER SCIENCE

- Relational Databases (Sometimes called a table)
- Relational Algebra
- Relational Calculus

Relational database worksheet

FUNCTION DEFINITIONS

- Function - Function f from A to B, f:A$\rightarrow$B, is a relation that associates to each a$\in$A a unique element f(a)$\in$B. (Also called mapping or transformation)
- A is the domain
- B is the codomain
- Vertical line test
- Range of f (or image) - im(f) = $\{$f(a):a$\in$A$\}$
- Identity function - id_A:A$\rightarrow$A where id_A=$\{$(x,x):x$\in$A$\}$

COMPOSITE FUNCTIONS

- Let $f:A\rightarrow B$ and $g:B\rightarrow C$ be functions. The composite function $g\circ f:A\rightarrow C$ is

  $g\circ f=\{(x,z)\in A\times C:(x,y)\in f \text{ and } (y,z)\in g \text{ for some } y\in B\}$

- $g\circ f(x)=g(f(x))$

- $im(g\circ f)\subseteq im(g)$

PROPERTIES OF FUNCTIONS

- Let f:A→B be a function:
  - Injection(1-1) - $\forall a, a^{'} \in A, if a \neq a^{'}$ then $f(a) \neq f(a^{'})$
  - Surjection(Onto) - $\forall b \in B$, $\exists a \in A$ such that $(a, b) \in f$
  - Bijection - both 1-1 and onto
  - If f is a bijection then g:B→A defined by g(b)=a iff f(a)=b is the inverse of f and is denoted by f^{-1}
- The composite of injective (surjective) functions is injective (surjective)
- If a composite g∘f is injective (surjective) then so is f(g)

Functions Worksheet

CARDINALITY

- Two sets have the same cardinality if there is a bijection between them.
- Countably Infinite sets have a bijection from the positive integers

APPLICATIONS OF FUNCTIONS IN COMPUTER SCIENCE

- Functions/Procedures in Programming Languages
- Functional Dependency
- Complexity Analysis

CSE2353 - GRAPHS & TREES OUTLINE

- What is a Graph?
- Special Types of Graphs
- Trees
- Fun Graph Problems
- Graphs in Computer Science

WHAT IS A GRAPH?

- An (Undirected) Graph G consists of:
  - A finite non-empty set V of Vertices
  - A finite set E of Edges
  - A function δ:E$\longrightarrow$P(V) such that, for every edge e, σ(e) is a one- or two-element subset of V.

GRAPH TERMS

- Adjacent - Vertices which are adjacent if they have an edge in common. Edges are adjacent if they have at least on vertex in common.
- Incident - Two vertices are incident to an edge if they are connected via that edge. In this case the edge is incident to the two vertices.
- Degree - The degree of a vertex is the number of edges which are incident to it.
- Subgraph - H is a subgraph of G if its vertices and edges are subsets of those in G.
- Edge Sequence - Sequence of edges such that any two consecutive ones are adjacent.
- Path - Edge sequence in which all edges are distinct
- Circuit - Closed (starting and ending vertex the same) simple (all vertices distinct) path with at least one edge.

GRAPH REPRESENTATIONS

- Suppose G={V,E} where |V|=n and |E|=m
- Adjacency Matrix n×n matrix A where $A_{i,j}$ is the number of edges joining v_i and v_j. (Note if only one edge exists, the value may be a weight associated to that edged.)
- Incidence Matrix m×n binary matrix B where $B_{i,j}$ is 1 iff e_i is incident with v_j.

DIRECTED GRAPHS

- Graph where edges have a direction
- Edges are sometimes called arcs

Graphs Worksheet

SPECIAL GRAPHS

- Simple - No loops or multiple edges
- Null - No edges.
- Complete K_n - Simple graph where every pair of distinct vertices is joined by an edge.
- Bipartite - Vertices can be partitioned such that every edge joins a vertex from one partition to that in another partition.
- Connected - Any pair of distinct vertices are connected via a path.

TREES

- Tree - Connected graph with no circuits
- Rooted Tree
  - Root
  - Leaves
  - Parent
  - Child
- Binary Tree - Every vertex has cardinality 2 or 3 (0 or 1 children)

Trees Worksheet

FUN GRAPH PROBLEMS

- Königsberg Bridge Problem p8
- Traveling Salesman Problem
- Stable Marriage Problem p3
- Four Color Problem

GRAPHS IN COMPUTER SCIENCE

- Graph Representation
  - Array
  - Linked List
  - Tree - One dimensional array
- Algorithms
- Applications
  - Indexing Data Structures (B^+ Tree)
  - Spelling Checkers (Trie)
  - Game Trees
  - Efficient Routing
  - Parsing
  - Sorting
  - Scheduling programs - Heap
  - Hierarchical and Network Databases
- NOTE - Graph proofs often use Induction

Finite State Machine

CSE2352 - BOOLEAN ALGEBRA OUTLINE

- Algebraic Structures
- What is a Boolean Algebra?
- Relationship to Sets and Propositions
- Boolean Functions
- Minterms/Maxterms
- Boolean Algebra in Computer Science

WHAT IS AN ALGEBRAIC STRUCTURE?

- Algebraic Structure - One or more sets with one or more operations
- Example: **Z** with +,-,$*$
- Example: Given S, $\cap$, $\cup$, - on P(S)
- Algebraic Structures are classified based on the properties of the operations
- The primary Algebraic Structure we will look at is a Boolean Algebra

BOOLEAN ALGEBRA INTRODUCTION

$(\{0,1\},+,*,^{-},0,1)$

$+$	0	1
0	0	1
1	1	1

$*$	0	1
0	0	0
1	0	1

b	$\overline{\text{b}}$
0	1
1	0

$(P(\{x\}),\cup,\cap,^{-},\emptyset,\{x\})$

$\cup$	$\emptyset$	$\{x\}$
$\emptyset$	$\emptyset$	$\{x\}$
$\{x\}$	$\{x\}$	$\{x\}$

$\cap$	$\emptyset$	$\{x\}$
$\emptyset$	$\emptyset$	$\emptyset$
$\{x\}$	$\emptyset$	$\{x\}$

b	$\overline{\text{b}}$
$\emptyset$	$\{x\}$
$\{x\}$	$\emptyset$

$(\{f,t\},\vee,\wedge,^{-},f,t)$

$\vee$	f	t
f	f	t
t	t	t

$\wedge$	f	t
f	f	f
t	f	t

b	$\overline{\text{b}}$
f	t
t	f

WHAT IS A BOOLEAN ALGEBRA?

- $(B,+,*,^{-},\mathbf{0},\mathbf{1})$
- Binary Operator $+$ (Sum)
- Binary Operator $*$ (Product)
- Unary Operator $^{-}$ (Complement)
- $\mathbf{0}$ (Identity): $\forall b \in B, b + \mathbf{0} = \mathbf{0} + b = b$
- $\mathbf{1}$ (Identity): $\forall b \in B, b * \mathbf{1} = \mathbf{1} * b = b$
- $+$ and $*$ are Associative
- $+$ and $*$ are Commutative
- $+$ Distributes over $*$
- $*$ Distributes over $+$
- $\forall b \in B, b + \overline{b} = \mathbf{1}$
- $\forall b \in B, b * \overline{b} = \mathbf{0}$

BOOLEAN ALGEBRA PROPERTIES

- Dual - Change Binary operators and Identities
- The dual of a Boolean algebra theorem is a Boolean algebra theorem
- The identity elements are unique
- The complement of an element is unique
- $\forall b \in B, b + b = b$ and $b * b = b$
- $\forall b \in B, \mathbf{1} + b = \mathbf{1}$ and
  $\mathbf{0} * b = \mathbf{0}$
- Absorption: $\forall b_1, b_2 \in B, b_1 + (b_1 * b_2) = b_1$
  and $b_1 * (b_1 + b_2) = b_1$
- Involution: $\forall b \in B, \overline{\overline{b}} = b$
- DeMorgan's Laws: $\forall b_1, b_2 \in B, (\overline{b_1 + b_2}) = \overline{b_1} * \overline{b_2}$
  and $(\overline{b_1 * b_2}) = \overline{b_1} + \overline{b_2}$

BOOLEAN EXPRESSIONS

- Given $(B, +, *, ^{-}, \mathbf{0}, \mathbf{1})$, a Boolean Variable is a variable over the set B
- A Literal is a Boolean variable, x, or its complement $\bar{x}$
- Boolean Expressions
  - Identity Elements **0, 1**
  - Boolean Variables $x_1, x_2, ..., x_n$
  - $(X + Y), (X * Y), \overline{X}$ where X and Y are Boolean Expressions
- Two Boolean expressions are equivalent (equal) if one can be obtained from the other by a finite sequence of applications of the Boolean algebra axioms

GENERAL BOOLEAN ALGEBRAS

- The Power Set for any finite set can be used to define a Boolean Algebra
- Example: Look at A={x,y}
- The cardinality of a finite Boolean algebra is a power of 2
- In Computer Science we are primarily interested in the Boolean algebra of cardinality 2

Boolean Algebra on Two Variables

Boolean Algebra on Four Variables

BOOLEAN FUNCTIONS

- A Boolean Function f:$B^n \rightarrow B$ such that f$(x_1, x_2, ..., x_n)$ is a Boolean expression
- Examples: $f_1(x_1, x_2) = x_1 + x_2$; $f_2(x_1, x_2, x_3) = \overline{x_1} * (x_2 + \overline{x_3})$
- Notation: Use $x_1 x_2$ to mean $x_1 * x_2$
- Precedence Order: Complement, Product, Sum, Left to Right

BOOLEAN FUNCTION ON ONE VARIABLE

- f(x)=**1**, g(x)=x
- Examine f(x)+f(x), f(x)+g(x), f(x)g(x), g(x)+g(x), g(x)g(x), $\overline{f(x)}$,$\overline{g(x)}$,$\overline{(}f(x))$
- Any function f(x) can be written as f(x)=f(**0**)x+f(**1**)$\overline{x}$.
- $\mathbf{0}x+\mathbf{0}\overline{x} = \mathbf{0}$
- $\mathbf{0}x+\mathbf{1}\overline{x} = \overline{x}$
- $\mathbf{1}x+\mathbf{0}\overline{x} = x$
- $\mathbf{1}x+\mathbf{1}\overline{x} = x+\overline{x} = \mathbf{1}$

MINTERMS & MAXTERMS

- One variable: x,$\overline{\text{x}}$
- Two variables: x_1x_2, $x_1\overline{x_2}$, $\overline{x_1}x_2$, $\overline{x_1x_2}$
- Minterm on n variables $x_1, x_2, ..., x_n$ is a Boolean expression which has the form of the product of each Boolean variable or its complement.
- Two minterms on one variable, four on two, eight on 3, and 2^n on n.
- Notation: $x^0 = \overline{x}$ and $x^1 = x$.
- $m_{e_1e_2...e_n} = x_1^{e_1}x_2^{e_2}...x_n^{e_n}$
- Examples: $m_{11} = x_1x_2$, $m_{10} = x_1\overline{x_2}$, $m_{01} = \overline{x_1}x_2$, $m_{00} = \overline{x_1x_2}$
- Maxterm on n variables $x_1, x_2, ..., x_n$ is a Boolean expression which has the form of the sum of each Boolean variable or its complement.
- Examples: $M_{11} = x_1 + x_2$, $M_{10} = x_1 + \overline{x_2}$, $m_{01} = \overline{x_1} + x_2$, $m_{00} = \overline{x_1 + x_2}$

CANONICAL FORMS OF BOOLEAN EXPRESSIONS

- Any Boolean expression can be written as the sum of minterms (or product of maxterms)
- Disjunctive Normal Form - Boolean expression written as sum of minterms
- Conjuctive Normal Form - Boolean expression written as product of maxterms
- We will normally use DNF
- To convert a Boolean expression, $f(x_1, x_2, ..., x_n)$ into DNF, we need to determine the values for the constant prefixes (That is the $f(e_1, e_2, ..., e_n)$).

Convert $x_1 + x_2$ into DNF

CONVERTING TO CANONICAL FORM

- Algorithm:

  1. Create table showing values of $f(e_1, ..., e_n)$
  2. Rewrite f using the output values shown as the prefixes for the corresponding terms

- Example:

e_1	e_2	$f(e_1, e_2)$
0	**0**	**0**
0	**1**	**0**
1	**0**	**1**
1	**1**	**1**

BOOLEAN ALGEBRA IN COMPUTER SCIENCE

- Switching Circuits
- Logic Networks
- Karnaugh Maps

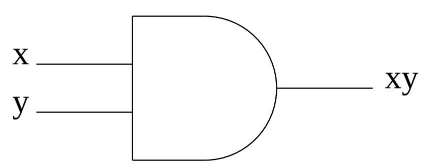
Design an adder

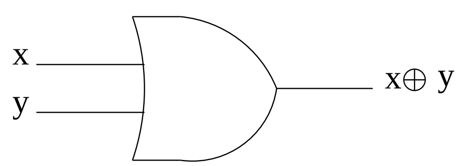
SWITCHING CIRCUITS

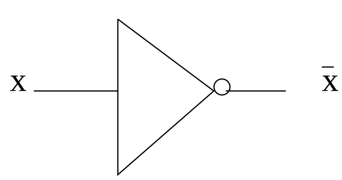
- Two-state Device: On or Off (1 or 0; T or F)
- Switch - Open (No current) or Closed (Current)
- Switches Can be combined:
  - Parallel - Boolean algebra +
  - Series - Boolean algebra $*$
  - If S denotes a switch then $\overline{S}$ denotes a switch which always has the opposite state
- Switching functions can be created to represent any Boolean function
- Switches can be mechanical or electronic

LOGIC NETWORKS

- Switching circuits either conduct (switch closed) or do not conduct (switch open) electricity. For example voltage-operated circuits could define **1** as a signal with ≥ 3 volts and **0** with 0 volts and an acceptable tolerance
- Gate - hardware that implements logic operations of AND, OR, NOT (inverter)
- Linking gates together creates a logic network







Logic Network Worksheet