

<b>1.1 Proofs</b>	
Double Negative	$\text{not}(\text{not}(P)) \Leftrightarrow P$
Associative Law	$(X \text{ and } (Y \text{ or } Z)) \Leftrightarrow ((X \text{ and } Y) \text{ or } Z)$
Distributive Law	$(X \text{ and } (Y \text{ or } Z)) \Leftrightarrow ((X \text{ and } Y) \text{ or } (X \text{ and } Z))$
Contrapositive	$X \rightarrow Y \Leftrightarrow \text{not}(Y) \rightarrow \text{not}(X)$
Equivalence	$X \leftrightarrow Y \Leftrightarrow (X \rightarrow Y) \text{ and } (Y \rightarrow X)$
<b>Proof Techniques</b>	
Exhaustive checking	
Conditional proof	
Proof by contradiction	
<b>1.2 Sets</b>	
Natural Numbers (N)	$\{0\}, 1, 2, 3, \dots$
Integers (Z)	$\dots, -3, -2, -1, 0, 1, 2, 3, \dots$
Rational numbers (R)	$A/B, A \text{ and } B \text{ in } Z$

Two characteristics of sets:

- There are no repeated occurrences of elements
- There is no particular order of elements

#### Power sets

$S = \{a, b, \{a, c\}\}$   
 $\text{power}(S) = \{\text{NULL}, \{a\}, \{b\}, \{a, c\}, \{a, b\}, \{a, \{a, c\}\}, \{b, \{a, c\}\}, \{a, b, \{a, c\}\}\}$   
 $\text{Cardinality}(S) = |S| = 3$

#### Subsets:

A is a subset of B if A contains all elements in B  
A is a proper subset of B if A is a subset of B and  $A \neq B$

Two sets are equal if they have the same elements,  
so A is a subset of B and B is a subset of A

#### Set Operations

$A \cup B = \{x \mid x \in A \text{ or } x \in B\}$   
 $A \cap B = \{x \mid x \in A \text{ and } x \in B\}$

#### Set properties

$A \cup \text{NULL} = A$   
 $A \cup (B \cup C) = (A \cup B) \cup C$   
 $A - B = \{x \mid x \in A \text{ and } x \text{ not in } B\}$   
 $A' = U - A$   
 $A \cup (A \cap B) = A, A \cap (A \cup B) = A$   
 $|A \cup B| = |A| + |B| - |A \cap B|$   
 $(A \cup B \cup C \cup D)' = A' \cap B' \cap C' \cap D'$

#### 1.3 Ordered Structures

Tuple: An ordered collection of elements

Two characteristics of tuples:

- There may be repeated occurrences of elements
- There is an order or arrangement of the elements

$A = \{A, B, C\}, B = \{1, 2\}$   
 $A^0 = \{\emptyset\}$   
 $A^1 = \{(a), (b), (c)\}$   
 $A^2 = \{(a, a), (a, b), (a, c), (b, a), (b, b), (b, c), (c, a), (c, b), (c, c)\}$   
 $A^* B = \{(a, 1), (a, 2), (b, 1), (b, 2), (c, 1), (c, 2)\}$

List: a finite ordered sequence of zero or more elements.

Two characteristics of lists:

- Elements can be repeated
- Only two accessible elements: element at the head, and the tail which is the list of elements

following the head

Cartesian Products

$\text{cons}(h, L)$  creates a list whose head is h and whose tail is L

#### 1.4 Graphs

Graph: a set of nodes that are interconnected by edges  
Adjacent: two nodes share an edge  
n-colorable: if a graph is n-colorable then its edges can be assigned n colors without any adjacent nodes sharing two same colored edges  
Chromatic number: the minimum n-color for a graph  
Complete graph: every node has an edge to every other node

Path: the series of edges that link one node to another

Connected graph: all nodes have a path to every other node

Cycle: a path that starts and ends at the same node

Graph traversal

Breadth first: visit all unvisited adjacent nodes of a given node, then visit all unvisited nodes adjacent to the adjacent nodes

Depth first: for some node, visit unvisited node, visit rest of unvisited nodes.

Tree: a connected graph without cycles  
Rooted tree: a tree with a node designated as root  
Height: the number of edges from root to farthest child  
Leaf: a childless node

#### 2.1 Functions

Function: an association between two sets, A and B, that map exactly one element from set A to set B.

Given sets A and B and function  $f: A \rightarrow B$

f maps elements from A to B

Domain: the set A

Codomain: the set B

Range: the subset of B that is mapped to by A

$\text{range}(f) = \{f(a) \mid a \in A\}$

Image: For any set S that is a subset of A, the image is the elements in A that are actually mapped to B

Injection, one-to-one: if f maps distinct elements of A onto distinct elements of B. AKA Differentiable.

Surjection: if f has a value from A mapped to every value in B. Also,  $\text{range}(f) = \text{codomain}(f)$

Bijection: f is injective and surjective

EG

$A = \{a, b, c\}, B = \{1, 2, 3\}$

$f: A \rightarrow B$

$f(a) = 1$

$f(b) = 1$

$f(c) = 2$

$\text{domain}(f) = \{a, b, c\}$   
 $\text{codomain}(f) = \{1, 2, 3\}$   
 $\text{range}(f) = \{1, 2\}$

Images:

$f(\{a\}) = \{1\}$

$f(\{a, b\}) = \{1\}$

$f(\{a, b, c\}) = \{1, 2\}$

Preimages:

$f^{-1}(\{1, 3\}) = \{a, b\}$

$f^{-1}(\{3\}) = \text{NULL}$

Partial functions: Functions that are undefined for some values

#### 2.1 Various functions

$\text{Floor}(x) \Rightarrow f: R \rightarrow Z$

$\text{Floor}(R) = Z \text{ iff } R - 1 < Z < R + 1 \text{ iff } Z - 1 < R \leq Z$

Properties:

$\text{Floor}(R + Z) = \text{Floor}(R) + Z$

$\text{Floor}(R) < Z \text{ iff } R < Z$

$Z < \text{Floor}(R) \text{ iff } Z < R$

$\text{Ceiling}(x) \Rightarrow f: R \rightarrow Z$

$\text{Ceiling}(R) = Z \text{ iff } R - 1 < Z \leq R \text{ iff } Z < R < Z + 1$

Properties:

$\text{Ceiling}(R + Z) = \text{Ceiling}(R) + Z$

$Z < \text{Ceiling}(R) \text{ iff } Z < R$

$\text{Ceiling}(R) \leq Z \text{ iff } R \leq Z$

$\text{gcd}(a, b) = \text{gcd}(b, a)$

$\text{gcd}(a, b) = \text{gcd}(b, a - bq) \text{ q} \in Z$

if  $g = \text{gcd}(a, b)$ , then  $g = ax + by, x, y \in Z$

algorithm:

$a = bq + r, r \in Z, b \neq 0$

#### 2.4 Countability

Given sets A and B, if A biject B, then  $|A| = |B|$

Informally, a set is countable if its elements can be counted in a step by step manner.  
Formally, a set is countable if it is finite or there is a bijection between it and N.

Countable properties:

Every subset of N is countable

S is countable iff  $|S| \leq |N|$

If  $S_0 \dots S_N$  is a sequence of countable set,  $S_0 \cup \dots \cup S_N$  is countable.

Diagonalization:

Let A be an alphabet with two or more symbols and let  $S_0 \dots S_N$  be a countable listing of sequences. The sequences are listed as the rows of an infinite matrix

$S_0| a_0, a_1, a_2, a_3, a_4, a_5, a_6,$

$S_1| a_{10}, a_{11}, a_{12}, a_{13}, a_{14}, a_{15}, a_{16},$

$S_{11}| a_{20}, a_{21}, a_{22}, a_{23}, a_{24}, a_{25}, a_{26}$

Then there is a sequence  $S = (a_0, a_1, a_2, a_3, a_4, \dots)$  over A that is not in the original list. S can be constructed from a diagonal list of elements,  $(a_{00}, a_{11}, a_{22}, a_{33}, \dots)$ .

#### 3.1 Inductively defined sets

An inductive definition of a S set consists of three steps:

Basis: Specify one or more elements of S.

Induction: Give one or more rules to construct new elements of S from existing elements of S.

Closure: State that S consists exactly of the elements obtained by the basis and induction steps (assumed).

All Strings over A

Basis:  $A \in A^*$

Induction: if  $s \in A$  and  $a \in A$ , then  $as \in A^*$ .

#### 3.2 Recursive functions and procedures

If S is an inductively defined set, then we can construct a function f with domain S as follows:

- For each basis element  $x \in S$ , specify a value for  $f(x)$ .
- Give rules that, for any inductively defined element  $x \in S$ , will define  $f(x)$  in terms of previously defined values of f.

#### 3.3 Grammars

A grammar is a set of rules used to define the structure of the strings in a language.

If L is a language over an alphabet A, then a grammar for L consists of a set of grammar rules of the form  $a \rightarrow b$ , where a and b denote strings of symbols taken from A and a set of grammar symbols disjoint from A.

The  $a \rightarrow b$  notation is also known as a production.

The four parts of a grammar

- An alphabet N of grammar symbols called nonterminals
- An alphabet T of symbols called terminals. Distinct from N.
- A specific nonterminal S, called the start symbol.
- A finite set of productions of the form  $a \rightarrow b$ , where a and b are strings over the alphabet  $N \cup T$  with the restriction that a is not the empty string.

EG:

$S \rightarrow A \mid aS \mid bS \mid cS$

$P = \{S \rightarrow A, S \rightarrow aS, S \rightarrow bS, S \rightarrow cS\}$

4 tuple =  $(\{S\}, \{a, b, c\}, S, P)$

#### 4.1 Properties of binary relations

For a binary relation R on a set A, we have the following definition

- R is reflexive if  $x R x$  for all  $x \in A$
- R is symmetric if  $x R y$  implies  $y R x$  for all  $x, y \in A$
- R is transitive if  $x R y$  and  $y R z$  implies  $x R z$  for all  $x, y, z \in A$
- R is irreflexive if  $(x, x) \notin R$  for all  $x \in A$
- R is antisymmetric if  $x R y$  and  $y R x$  implies  $x = y$  for all  $x, y \in A$

If R and S are binary relations, then the composition of R and S is the following relation:

$R$  composition  $S = \{(a, c) \mid (a, b) \in R \text{ and } (b, c) \in S\}$

#### 4.2 Equivalence relations

Any binary relation that is reflexive, symmetric, and transitive is called an equivalence relation.

Intersection Property of Equivalence

If E and F are equivalence relations on the set A, then  $E \cap F$  is an equivalence relation on A.

Kernel relations

If f is a function with domain A, then relation ~ defined by

$x \sim y \text{ iff } f(x) = f(y)$

is an equivalence relation on A, and is called the kernel relation of f.

Equivalence class

Let R be an equivalence relation on a set S. if  $a \in S$ , then the equivalence class of a, denoted by  $[a]$ , is the

subset of S consisting of all elements that are

equivalent to a. In other words, we have

$[a] = \{x \in S \mid x R a\}$

#### 4.3 Order Relations

Definition of a partial order

A binary relation is called a partial order if it is antisymmetric, transitive, and either reflexive or irreflexive.

Definition of a partially ordered set

The set over which a partial order is defined is called a partially ordered set, or poset. IF we want to emphasize the fact that R is the partial order that makes S a poset, we can write  $\langle S, R \rangle$ .

Descending chains and minimality

If A is a well-founded set, then every nonempty subset of A has a minimal element. Conversely, if

## 1 Inductively Defined Sets

$A = \{3, 5, 7, 9, \dots\}$  can be represented as  $A = \{2k+3 \mid k \in \mathbb{N}\}$   
But we can also describe A by saying  $3 \in A \Rightarrow x+2 \in A$  and nothing else is in A.

**Sets specified as unions of inductively defined sets:**  
 $A = \{2, 3, 4, 7, 8, 11, 15, 16, \dots\}$  can be expressed as  $B \cup C$ ,  
 $B = \{2, 4, 8, 16, \dots\}$  and  $C = \{3, 7, 11, 15\}$   
*Basis:*  $2, 3 \in A$

*Induction:* If  $x \in A$  and  $x$  is odd, then  $x+4 \in A$   
If  $x \in A$  and  $x$  is even, then  $2x \in A$

### 1.1 Strings

**All Strings over A:** *Basis:*  $\Lambda \in A$ , *Induction:* If  $s \in A^*$  and  $a \in A$ , then  $as \in A^*$

**Inductive Definition of Languages:**  $S = \{a, ab, abb, abbb\} = \{ab^n \mid n \in \mathbb{N}\}$

*Basis:*  $a \in S$ , *Induction:* If  $x \in S$ , then  $xb \in S$

### 1.2 Lists

$\langle x, y, z \rangle = \text{cons}(x, \langle y, z \rangle) =$   
 $\text{cons}(\text{head}(\langle x, y, z \rangle), \text{tail}(\langle x, y, z \rangle))$

Don't forget,  $\text{cons}(x, \langle y, z \rangle) = x :: \langle y, z \rangle$

**All lists over A:** *Basis:*  $\langle \rangle \in \text{lists}(A)$ , *Induction:* If  $x \in A$  and  $L \in \text{lists}(A)$ , then  $\text{cons}(x, L) \in \text{lists}(A)$

### 1.3 Binary Trees

**All Binary Trees over A:** *Basis:*  $\langle \rangle \in B$ , *Induction:* If  $x \in A$  and  $L, R \in B$ , then  $\text{tree}(L, x, R) \in B$

### 1.4 Cartesian Products of Sets

**Cartesian Product:** *Basis:*  $(0, a) \in \mathbb{N} \times A \ \forall a \in A$ , *Induction:* If  $(x, y) \in \mathbb{N} \times A$ , then  $(x+1, y) \in \mathbb{N} \times A$

**Part of a plane:** Let  $S = \{(x, y) \mid x, y \in \mathbb{N} \text{ and } x \leq y\}$ . S is set of points in first quadrant, on or above the main diagonal.

*Basis:*  $(0, 0) \in S$ , *Induction:* If  $(x, y) \in S$ , then  $(x, y+1), (x+1, y+1) \in S$

## 2 Recursive Functions

A function or procedure is recursively defined if defined in terms of itself. Constructing a recursively defined function: if S is inductively defined set, then construct function  $f$  with domain S as follows: 1. for each basis element  $x \in S$ , specify value for  $f(x)$ ; 2. give rules that for any inductively defined element  $x \in S$ , specify a value for  $f(x)$ .

### 2.1 Numbers

Let  $f: \mathbb{N} \rightarrow \mathbb{N}$  be defined in terms of floor as follows:  
 $f(0) = 0$ ,  $f(n) = f(\text{floor}(n/2)) + n$  for  $n > 0$ , then:

$f(25) = f(12) + 25 = f(6) + 12 + 25 = f(3) + 6 + 12 + 25 =$   
 $f(1) + 3 + 6 + 12 + 25 = f(0) + 1 + 3 + 6 + 12 + 25 = 0 + 1 +$   
 $3 + 6 + 12 + 25 = 47$

### 2.2 Lists

Consider  $f: \mathbb{N} \rightarrow \text{lists}(\mathbb{N})$  which computes the backward sequence:  $f(n) = \langle n, n-1, \dots, 1, 0 \rangle$ . We can define this recursively as:  $f(0) = \langle 0 \rangle$ ,  $f(n) = n :: f(n-1)$  for  $n > 0$ .

**The pairs function:**

$\text{pairs}(\langle a, b, c \rangle, \langle 1, 2, 3 \rangle) = \langle \langle a, 1 \rangle, \langle b, 2 \rangle, \langle c, 3 \rangle \rangle =$   
 $\langle a, 1 \rangle :: \langle \langle b, 2 \rangle, \langle c, 3 \rangle \rangle = \langle a, 1 \rangle :: \text{pairs}(\langle b, c \rangle, \langle 2, 3 \rangle)$ . So pairs can be defined recursively as:

$\text{pairs}(\langle \rangle, \langle \rangle) = \langle \rangle$ ,  $\text{pairs}(x :: T, y :: U) = (x, y) :: \text{pairs}(T, U)$

### 2.3 Binary Trees

**Preorder traversal:** visit(root), preorder(L), preorder(R).

**Inorder traversal:** inorder(L), visit(root), inorder(R).

**Postorder traversal:** postorder(L), postorder(R), visit(root).

## 3 Grammars

Example:  $A = \{a, b, c\}$ , the grammar for the language  $A^*$  has 4 productions:  $\{S \rightarrow \Lambda, S \rightarrow aS, S \rightarrow bS, S \rightarrow cS\}$ . A Grammar is a 4-tuple:

1. alphabet  $N$  of nonterminals, 2. alphabet  $T$  of *terminals*, distinct from nonterminals, 3. specific nonterminal  $S$  called start symbol, 4. finite set of products of form  $\alpha \rightarrow \beta$ , where  $\alpha$  and  $\beta$  are strings over  $N \cup T$  and  $\alpha \neq \Lambda$ .

### 3.1 Derivations

If  $x$  and  $y$  are sentential forms and  $\alpha \rightarrow \beta$  is a production, then replacement of  $\alpha$  by  $\beta$  in  $x\alpha y$  is called a derivation step, written:  $x\alpha y \Rightarrow x\beta y$ .

$\Rightarrow$  derives in one step;  $\Rightarrow^+$  derives in one or more steps;  $\Rightarrow^*$  derives in zero or more steps

**The language of a grammar:** if  $G$  is a grammar with start symbol  $S$  and set of terminals  $T$ , then language of  $G$  is the set  $L(G) = \{s \mid s \in T^+ \text{ and } S \Rightarrow^+ s\}$

A grammar is **recursive** if it contains a recursive production or indirectly recursive production.  $S \rightarrow b \mid aA$ ,  $A \rightarrow c \mid bS$  is indirectly recursive because  $S \Rightarrow aA \Rightarrow abS$ , and  $A \Rightarrow bS \Rightarrow baA$ .

**Constructing an inductive definition for  $L(G)$ :**  $G: S \rightarrow \Lambda \mid aB, B \rightarrow b \mid bB$ . 2 derivatives don't contain recursive productions:  $S \Rightarrow \Lambda$ , and  $S \Rightarrow aB \Rightarrow ab$ . This is basis:  $\Lambda, ab \in L(G)$ . Only recursive production of  $G$  is  $B \rightarrow bB$ . Any element of  $L(G)$  whose derivation contains  $B$  must have form  $S \Rightarrow aB \Rightarrow^+ ay$  for some string  $y$ . Then we can say  $S \Rightarrow aB \Rightarrow abB \Rightarrow^+ aby$ . *Induction:* If  $ay \in L(G)$ , then put  $aby \in L(G)$ .

**Constructing grammars:**  $L = \{\Lambda, ab, aabb, \dots, a^n b^n, \dots\} = \{a^n b^n \mid n \in \mathbb{N}\}$ . Grammar:  $S \rightarrow \Lambda \mid aSb$ .

## 1 Proofs, Sets, Graphs, and Trees

conjunction  $\Rightarrow A \cup B$ , disjunction  $\Rightarrow A \cap B$ , converse if  $x > 0$  and  $y > 0 \Rightarrow x+y > 0$  or if  $x+y > 0 \Rightarrow x > 0 \cup y > 0$ ,  
' $(A \cup B) = (A \cap B)$ , ' $(A \cap B) = (A) \cup (B)$

## 2 Functions

## 3 Construction Techniques

## 4 Equivalence, Order, Induction

### Binary Relation Relation properties

R is *reflexive* if  $xRy$  for all  $x, y \in A$ . R is *symmetric* if  $xRy$  implies  $yRx$  for all  $x, y \in A$ . R is *transitive* if  $xRy$  and  $yRz$  implies  $xRz$  for all  $x, y, z \in A$ . R is *irreflexive* if  $(x, y) \notin R$  for all  $x \in A$ . R is *antisymmetric* if  $xRy$  and  $yRx$  implies  $x = y$  for all  $x, y \in A$ . The  $<$  on  $\mathbb{R}$  is transitive, symmetric, reflexive, and antisymmetric. The  $\leq$  relation on  $\mathbb{R}$  is reflexive, transitive and antisymmetric. The "is parent of" relation is irreflexive and antisymmetric.

### Composition of Relations

If  $R \circ S = \{(a, c) \mid (a, b) \in R \text{ and } (b, c) \in S \text{ for some element } b\}$ . To construct the "isGrandparentOf" relation we can compose "isParentOf".  $\text{isGrandparentOf} = \text{isParentOf} \circ \text{isParentOf}$ . If we try an combine relations, i.e:  $x(> <)y$  iff there is some number  $z$  such that  $x > y$  and  $z < y$ . With  $\mathbb{R}$  we know there is always another number  $z$  that is less than both. So the whole composition must be  $\mathbb{R} \times \mathbb{R}$ .

### Properties of combining Relations

$R \circ (S \circ T) = (R \circ S) \circ T$

$R \circ (S \cup T) = R \circ S \cup R \circ T$

$R \circ (S \cap T) = R \circ S \cap R \circ T$

### Equivalence Relations

Any binary relation that is reflexive, symmetric and transitive is called an *equivalence relation*. The intersection property of equivalence follows: if E and F are equivalence relations on the set A, then  $E \cap F$  is an eq. rel. on A. We can also say  $x \sim y$  iff  $xEy$  and  $xFy$  which can also be said,  $x \sim y$  iff  $(x, y) \in E \cap F$

### Kernel Relations: eq rel on functions

Notice that we can show  $x \sim y$  iff  $f(x) = f(y)$ . This is called the kernel relation of  $f$ . Mod is one function that can be defined  $x \sim y$  iff  $x \bmod n = y \bmod n$ .

### Equivalence Classes

Let R be an eq rel on a set S. If  $a \in S$ , then the eq class of  $a$  by [a] is the subset of S consisting of all elements that are eq to  $a$ . In other words we have  $[a] = \{x \in S \mid xRa\}$ . The property of equivalences is as follows: Let S be a set with

an equivalence relation R. If  $a, b \in S$ , then either  $[a] = [b]$  or  $[a] \cap [b] = \emptyset$ . Partitions are the collection of nonempty subsets that are disjoint whose union is the whole set. For example  $S = \{1, 2, 3, 4, 5, 6, 7, 8, 9\}$  can be partitioned in many ways, one of which consists of  $\{0, 1, 4, 9\}, \{2, 5, 8\}, \{3, 6, 7\}$ . We could also say  $[0] = \{0, 1, 4, 9\}$ ,  $[2] = \{2, 5, 8\}$ ,  $[3] = \{3, 6, 7\}$ . This is generalized into the rule: If R is an eq. rel on the set S, then the eq classes form a partition of S. Conversely, if P is a partition of a set S, then there is an eq rel on S whose eq classes are sets of P.

### Partial Orders

A binary relation is called a *partial order* if it is antisymmetric, transitive, and either reflexive or irreflexive. The set over which a partial order is defined is called a *partially ordered set* or *poset* for short. If we want to emphasize that R is the partial order that makes S a poset, we'll write  $(S, R)$ . Symbols used: *irreflexive partial order* or  $<$  and *reflexive partial order* or  $\leq$ . These are read as  $a < b$  or  $a$  is less than  $b$  and  $a \leq b$  or  $a$  is less than or equal to  $b$ . We can define them with each other too:  $\leq = < \cup \{(x, x) \mid x \in A\}$  and  $< = \leq - \{(x, x) \mid x \in B\}$ . We talk of *predecessors* like so:  $\{z \in A \mid x < z < y\} = \emptyset$ . We can also say that  $y$  is an *immediate successor* of  $x$  here. An element in a poset is considered a *minimal* element of S if it has no predecessors. An element is the *least* if  $x \leq y$  for all  $y \in S$ . The *maximal* element and *greatest* elements are simply the reverse.

### Inductive Proof

The important thing to remember about applying inductive proof techniques is to *make an assumption* then *use the assumption* just made. The **Principle of Mathematical Induction** follows: Let  $m \in \mathbb{Z}$ . To prove that  $P(n)$  is true for all integers  $n \geq m$ , perform the following two steps:

1. Prove that  $P(n)$  is true.
2. Assume that  $P(k)$  is true for an arbitrary  $k \geq m$ . Prove that  $P(k+1)$  is true.

**Second Principle of Induction:** Let  $m \in \mathbb{Z}$ . To prove that  $P(n)$  is true for all integers  $n \geq m$ , perform the following two steps:

1. Prove that  $P(m)$  is true.
2. Assume that  $n$  is an arbitrary integer  $n > m$ , and assume that  $P(k)$  is true for all  $k$  in the interval  $m \leq k < n$ . Prove that  $P(n)$  is true.

**Analyzing Algorithms** The Optimal Algorithm Problem: Suppose algorithm  $A$  solves problem  $P$ . Is  $A$  the best solution to  $P$ ?

**Definition of Optimal in the Worst Case** An algorithm  $A$  is *optimal in the worst case* for problem  $P$ , if for any algorithm  $B$  that exists, or ever will exist, the following relationship holds:

$$W_A(n) \leq W_B(n), \forall n > 0.$$

**Decision Trees** Lower bound on a  $k$ -ary decision tree of depth  $n$ :  $\lceil \log_k n \rceil$

**Summations Facts**

- a.  $\sum_{k=m}^n c = (n - m + 1)c$
- b.  $\sum_{k=m}^n ca_k = c \sum_{k=m}^n a_k$
- c.  $\sum_{k=1}^n (a_k - a_{k-1}) = a_n - a_0$  and  $\sum_{k=1}^n (a_{k-1} - a_k) = a_0 - a_n$
- d.  $\sum_{k=m}^n (a_k + b_k) = \sum_{k=m}^n a_k + \sum_{k=m}^n b_k$
- e.  $\sum_{k=m}^n a_k = \sum_{k=m}^i a_k + \sum_{k=i+1}^n a_k$
- f.  $\sum_{k=m}^n a_k x^{k+i} = x^i \sum_{k=m}^n a_k x^k$
- g.  $\sum_{k=m}^n a_{k+i} = \sum_{k=m+i}^{n+1} a_k$

**Sum of Powers** When doing powers of sums we can use the form of  $k^m$  can also be represented as  $k^{m+1} - (k + 1)^{m+1}$ .

**Closed Forms of Elementary Finite Sums**

- a.  $\sum_{k=1}^n k = \frac{n(n+1)}{2}$
- b.  $\sum_{k=1}^n k^2 = \frac{n(n+1)(2n+1)}{6}$
- c.  $\sum_{k=0}^n a^k = \frac{a^{n+1} - 1}{a - 1} (a \neq 1)$
- d.  $\sum_{k=1}^n ka^k = \frac{a - (n+1)a^{n+1} + na^{n+2}}{(a - 1)^2} (a \neq 1)$

**Abel's Summation Tranformation**

$$\sum_{k=0}^n a_k b_k = A_n b_n + \sum_{k=0}^{n-1} A_k (b_k - b_{k+1}),$$

where  $A_k = \sum_{i=0}^k a_i$

**Counting Rules** If there are  $m$  choices for some event and  $n$  choices for another event to occur and events are disjoint, then there are  $m + n$  choices for either event to occur. If there are  $m$  choices for some event and  $n$  choices for another event, then there are  $mn$  choices for both events.

**Permutations** An arrangement (or ordering) of distinct objects without replacement. The number of permutations of  $n$  distinct objects is  $n!$ . An  $r$ -permutation of  $n$  objects is a permutation of  $r$  of the objects. The number of  $r$ -permutations of  $n$  distinct objects is

$$P(n, r) = \frac{n!}{(n - r)!}$$

$B$  is an  $n$ -element bag with  $k$  distinct elements, where each of the numbers  $m_1, \dots, m_k$  denotes number of occurrences of each element. The number of permutations of the  $n$  elements of  $B$  is

$$\frac{n!}{m_1! \dots m_k!}$$

**Combinations** Chosing some objects from set of objects without order. An  $r$ -combination of  $n$  distinct objects is a combination of  $r$  of the objects. The number of  $r$ -combinations chosen from  $n$  distinct objects is

$$C(n, r) = \frac{n!}{r!(n-r)!} \text{ or } (x + y)^n = \sum_{k=0}^n C(n, r) x^{n-k} y^k$$

The number of  $k$ -element bags whose distinct elements are chosen from an  $n$ -element set, where  $k$  and  $n$  are positive, is given by  $C(n + k - 1, k)$

**Probability Distribution** A *probability distribution* on a sample space  $S$  is an assignment of probabilities to the points of  $S$  such that the sum of all the probabilities is 1. **Probability of an Event** The *probability* of an event  $E$  is denoted by  $P(E)$  and is defined by

$$P(E) = \sum_{x \in E} P(x)$$

For instance,  $P(S) = 1$  and  $P(\emptyset) = 0$ ;  $P(A \cup B) = P(A) + P(B) - P(A \cap B)$ ;  $P(E') = 1 - P(E)$

**Conditional Probability** If  $A$  and  $B$  are events and  $P(B) \neq 0$ , then the *conditional probability of  $A$  given  $B$*  is denoted by  $P(A | B)$  and defined by

- $P(A | B) = \frac{P(A \cap B)}{P(B)}$
- $P(A \cap B) = P(B)P(A | B)$
- $P(A \cap B) = P(A)P(B | A)$

**Bayes' Theorem**

$$P(H_i | E) = \frac{P(H_i \cap E)}{P(H_1 \cup E) + \dots + P(H_n \cap E)}$$

$$P(H_i | E) = \frac{P(H_i \cap E)}{P(H_1)P(E | H_1) + \dots + P(H_n)P(E | H_n)}$$

**5.4.3 Independent Events** Two events  $A$  and  $B$  are *independent* if the following equation holds:  $P(A \cap B) = P(A)P(B)$ .

**Binomial Distribution**

$$\binom{n}{k} p^k (1 - p)^{n-k}$$

**Solving Recurrences by Substitution** Substitute occurrences  $f_n$  on right side of equation until pattern emerges.

$$f_0 = b_0 \text{ and } f_n = 2f_{n-1} + n \text{ and } f_n = 2^2 r_{n-2} + 2(n - 1) + n$$

**Solving Recurrences by Cancellation** Write succeeding equations for  $f_n$  such that the term on the left side is the same as the term that contains  $f$  on the right side of previous equation.

$$r_n = 2r_{n-1} + n \text{ and } 2r_{n-1} = 2^2 r_{n-2} + 2(n - 1) \dots$$
$$2^{n-1} r_1 = 2^n r_0 + 2^{n-1}$$

Add equations, cancel like terms, replace  $r_0$  by its value.

**5.5.2 Divide-and-Conquer recurrences**

Consider algorithm that splits a problem of size  $n$  into a smaller problems, where each subproblem has size  $n/b$ . Let  $T(n)$  be total operations to solve problem of size  $n$ .

$$T(n) = aT(n/b) + f(n)$$

**5.5.3 Generating Functions**

The generating function for the sequence  $a_0, a_1, \dots, a_n$  is

$$\sum_{n=0}^{\infty} a_n x^n$$

**More Generating Functions**

- a.  $\frac{1}{(1-x)^{k+1}} = \sum_{n=0}^{\infty} \binom{k+n}{n} x^n$ , for  $k \in \mathbb{N}$
- b.  $(1+x)^r = \sum_{n=0}^{\infty} \left( \frac{r(r-1)\dots(r-n+1)}{n!} \right) x^n$ , for  $r \in \mathbb{R}$

**Geometric Series**

$$\frac{1}{1-x} = \sum_{n=0}^{\infty} x^n$$

**5.6 Big-O Face Notation**

**Comparing Rates of Growth** Big Oh (Big-O) can be defined as the *growth rate of  $f$  is bounded above by the growth rate of  $g$*  if there exists positive numbers  $c$  and  $m$  such that  $|f(n)| \leq c|g(n)|$  for  $n \geq m$ . In this case we write  $f(n) = O(g(n))$  and we say the  $f(n)$  is *big oh of  $g(n)$*  **Properties of Big Oh**

- a.  $f(n) = O(f(n))$
- b. If  $f(n) = O(g(n))$  and  $g(n) = O(h(n))$ , then  $f(n) = O(h(n))$

- c. If  $0 \leq f(n) \leq g(n)$  for all  $n \geq m$ , then  $f(n) = O(g(n))$
- d. If  $f(n) = O(g(n))$  and  $a$  is any real number, then  $af(n) = O(g(n))$
- e. If  $f_1(n) = O(g(n))$  and  $f_2(n) = O(g(n))$ , then  $f_1(n) + f_2(n) = O(g(n))$
- f. If  $f_1$  and  $f_2$  have nonnegative values and  $f_1(n) = O(g_1(n))$  and  $f_2(n) = O(g_2(n))$ , then  $f_1(n) + f_2(n) = O(g_1(n) + g_2(n))$

**Bonus: Partial Infractions**

Consider expressions of following form, where  $p(x)$  and  $q(x)$  are two polynomials and the degree of  $p(x)$  is less than the degree of  $q(x)$ :

$$\frac{p(x)}{q(x)}$$

For example, if we have

$$\frac{x+1}{(2x-1)(3x-1)}$$

Then we can write

$$\frac{x+1}{(2x-1)(3x-1)} = \frac{A}{2x-1} + \frac{B}{3x-1}$$

We need two equations in  $A$  and  $B$ . One way is to pick values for  $x$  such that none of the denominators are made zero. For instance, picking  $x = 0$  and  $x = 1$  yields:

$$1 = -A - B$$

$$1 = A + B/2$$

Solving for  $A$  and  $B$  yields  $A = 3$  and  $B = -4$ , so the partial fraction representation of the expression is:

$$\frac{x+1}{(2x-1)(3x-1)} = \frac{3}{2x-1} - \frac{4}{3x-1}$$