CIS 160 - Fall 2012 (instructor Val Tannen)

Lecture Notes Skeleton

1 Introduction

1. This is a course about the kind *mathematical modeling* that has become essential in today's technologies and sciences, and even in some arts. Such modeling uses *discrete mathematics* and it's essential ingredient kind in applying the power of computers to solve problems. It is forced upon us because computers apply their power through *software*.

   In inventing the models that enable us to write software (i.e., to *program*) people face two classes of problems:

   - How to represent *information*;
   - How to enable *computation*, i.e., to figure out *algorithms* that solve our problems.

2. Model, abstraction. Example: vending machine.

2 Proofs and Counting with Sets, Functions and Relations

1. **Review:** set, element of a set, set inclusion, subset, proper subset, empty (sub)set, union, intersection, disjoint sets, difference.

2. **Review:** , cartesian product (cross-product), pairs, triples, tuples. (Finite) sequences. Arrays. Lists.

3. "Measuring" the size of sets, i.e., counting the number of elements. The Product Rule, the Sum Rule. Example: counting menu orders.

4. Example: multiple representations of the same (ideal) number. Representing numbers in computers. Bits, bytes, and gigabytes.

5. **Review:** function, domain, codomain. Representing functions with tables, with arrow diagrams.

6. Positions in a sequence, sequences as functions. The number of sequences of n bits is (2^n).

7. Injective (one-to-one) function (injection), surjective (onto) function (surjection), range, bijective function (bijection, one-to-one correspondence).

8. Comparing the size of sets with injections or surjections, the Bijection Rule.

9. The number of subsets of a set by bijection with the number of sequences of bits. Discussion on level of detail in proofs.

10. Proof pattern: proving set inclusions and set equalities. Proof pattern: by cases. Example: $(A \cap B) \cup C \subseteq A \cup C$.

11. Direct image of a subset of the domain through a function. $(f : X \to Y, A \subseteq X), f(A) \subseteq Y$. Range, $f(X)$.

12. Proof pattern: proving inclusions involving direct images. Example: $f(A \cup B) = f(A) \cup f(B)$.

13. The number of functions from X to Y is $|Y|^{|X|}$).

14. The Generalized Product Rule. Example: watchers in a grid. Permutations ($n!$), permutations of k out of n.

15. The Division Rule. Example: another way to count permutations of k out of n.

16. Proof pattern: proving implications by contrapositive. Example: injectivity.

17. Negation of conjunction/disjunction leads to disjunction/conjunction. Example: contrapositive of "$ab =$ implies $a = 0$ or $b = 0$".

18. Proof pattern: falsity implies anything, Example: cases that cannot happen.

19. Proof pattern: proof by contradiction, relationship with the contrapositive. Example: r irrational implies $\sqrt{r}$ irrational. Example: $\sqrt{2}$ is irrational. Example sum of catheti is bigger than hypothenuse.

20. Combinations $\binom{n}{k}$, number of subsets of size k. Example: counting handshakes.

21. Counting one thing by counting another: more of the Bijection Rule. Example: $\binom{\binom{n}{k}=n}{n-k}$. Example: counting the ways of placing indistinguishable objects into distinguishable bins (choosing a dozen donuts of five flavors).

22. The binomial theorem, Pascal's triangle, Pascal's identity.

23. Function composition, it's not commutative. The identity function. The inverse of a bijection.

24. Predicates, "for all" ($\forall$), "there exists" ($\exists$), quantification. Negation of quantifiers. Example: a function that has an inverse must be a bijection.

25. Binary relations, graph of a relation. Example: professors to courses.

26. Total binary relations, functional binary relations, functions are particular cases of binary relations.

27. Direct and inverse images of subsets of the domain and codomain under a binary relation. Example: proof of properties.

28. The inverse of a relation. Example: proof of properties.

29. The Principle of Inclusion-Exclusion. Example: students by courses.

30. The Pigeonhole Principle. The Generalized PH Principle. Example: people and hair. Example: robots in a pen sensing each other. Example: the 3 friends or 3 strangers theorem. Diversion: Ramsey's Theorem (for two colors), Ramsey Numbers, why we cannot figure out $43 \leq R(5,5) \leq 49$ by "brute force".

3 Ordinary and Strong Induction, Recursive Definitions, Structural Induction

1. Ordinary induction. Intuition: the induction "machine". Example: sum of first n odd numbers. The proof pattern.

2. Examples of proof by ordinary induction. (a) $\sum_{i=1}^{n} i \cdot i! = (n+1)! - 1$; (b) $q \geq 2 \Rightarrow \sum_{i=0}^{n} q^i < q^{n+1}$; (c) any triangulation of a polygon with 4 or more vertices contains at least two exterior triangles. Induction trap: all of Abe's sheep have the same color.

3. $f : \mathbb{N} \to \mathbb{R}$ is monotone (increasing) iff $\forall n \in \mathbb{N}\ f(n) \leq f(n+1)$

4. Proof of the Binomial Theorem by ordinary induction, using Pascal's identity.

5. Strong induction. Fibonacci numbers, rabbits, and the Golden Ratio. Example: $\forall n,\ F_n < 2^n$. Induction trap: every Fibonacci number is even.

6. Example: the number of ways to park Fiats and Hummers in n contiguous spots.

7. The Well Ordering Principle and its relationship with induction.

8. Recursive definitions of data (recursive data types). Constructors. Example: strings over an alphabet, A^*, the empty string λ.

9. Recursive definitions of functions whose domain consists of recursively defined data. Example: length of a string $|s|$, string concatenation $s \cdot t$.

10. Structural induction. Example: $|s \cdot t| = |s| + |t|$.

11. The natural numbers as a recursive data type. Bijection between $\mathbb{N}$ and $\{z\}^*$, numbers in base ... 1!

12. Recursive definitions of data (recursive data types), Example: strings of matched brackets as a recursive data type. Prove that the number of open brackets equals the number of closed brackets. Addendum: checking well-formedness with a stack.

13. Ambiguous recursive definitions.

14. Arithmetic expressions and their evaluation.

4 A Tiny Bit of Probabilities

1. Example: the Month Hall Problem. Modeling "in four steps": (1) define outcome space, (2) define events of interest, (3) determine outcome probabilities, and (4) compute event probabilities. Example: strange dice.
2. Probability spaces: sample spaces, outcomes, events, probabilities.
3. Properties of probability: monotonicity rule, the sum rule, the complement rule. The inclusion-exclusion principle for probability. Boole's inequality and the union bound.
4. Uniform probability spaces. Examples with dice and coins.
5. Careful analysis of sample space: the chord problem.
6. Conditional probability. Its properties.
7. Independence, notation $E \perp F$. Relationship with conditional probability. Property: If $E \perp F$ then. $\Pr[E \cup F] = 1 - (1 - \Pr[E])(1 - \Pr[F])$
8. 3 or more events: mutual independence.

5 Graphs

1. Vertices (nodes) and edges (arrows, arcs): the stuff graphs are built of. The bewildering zoo of graph terminology; directed graphs (digraphs), simple digraphs (directed simple graphs), multidigraphs, (undirected) graphs, simple graphs, multigraphs, labeled version of the previous, etc. Examples.
2. Directed simple graphs. (Binary relations!) Indegree, outdegree. Walk, path, closed walk, cycle, and their length. Walk merge (concatenation).
3. Shortest walk (=shortest path), distance between vertices in the graph, the triangle inequality.
4. Adjacency matrix of a graph. The relationship between counting number of walks and matrix multiplication.
5. Walk relations. Binary relations on a set: transitivity, reflexivity, transitive closure, transitive-reflexive closure.
6. Directed acyclic graph (DAG). Topological sort. Example: task scheduling.
7. Irreflexivity, asymetry, antisymmetry. Strict partial order. Partial order. Posets. Examples: set inclusion, divisibility. Minimum, minimal, maximum, and maximal elements. Chain. Antichain. Total (linear) order. Product order. Lexicographic order.
8. Equivalence relation, equivalence relation defined by a function on its domain.

9. Equivalence classes, partition. Example: strongly connected components in a digraph.

10. Simple undirected graph, incidence, adjacency, degree, handshaking lemma, K_n, L_n, C_n.

11. Walk, path, closed walk, cycle, subgraph, connectivity, connected components. Examples: counting walks, paths, cycles in special graphs.

12. Minimum number of edges in a connected graph (more generally, every graph has at least $|V| - |E|$ connected components).

13. Forest, tree, leaves. Tree properties: unique path between nodes, adding an edge creates a cycle, removing an edge disconnects the graph, $|V| = |E| + 1$. Spanning tree of a connected graph.

14. Graphs in which the edges have length/weight. Shortest paths, minimum weight spanning tree.

15. Rooted trees become digraphs. Parent/child, ancestor/descendant relationships. Example: phylogenies.

16. Ordered rooted trees. Example: binary (search) trees.

17. Recursive definition for binary (rooted and ordered) trees. Complete binary trees. Tree height. Any binary tree of height $h \geq 0$ has at most 2^h leaves.

6 Wrap-Up

18. Looking ahead at databases. Tables and the relational data model. Using logic to formulate queries.

19. Looking ahead at automata. Modeling states and state transitions.

20. Looking ahead at algorithms. Modeling with graphs and graph manipulations. Boolean expressions. SAT. Hardness.

7 Reading Assignments

The following parts of Albert Meyer's textbook were assigned:

Sections 4.1, 4.2, 4.3, 14.1, 14.2; 14.3, 14.4, 14.5; 1.5, 1.6, 1.7, 1.8, 1.9; 1.1, 1.2, 4.4, 4.5, 14.10 (except 14.10.3), 2.1, 2.2, Chapter 5, Sections 14.8 (only 14.8.1 and 14.8.2), 14.9 (except 14.9.5), 16.1, 16.2, 16.3, 16.4 (except 16.4.4), 16.5, 16.6 (except 16.6.6), Chapter 6, Chapter 9, Sections 11.1, 11.3, 11.8, 11.9 (except 11.9.2), 11.11 (except Prim and Kruskal's algorithms).