CS 181 (Formal Languages and Automata Theory) Notes

Lecture Notes

4/2 - Week 1

- As models get more complex, you're increasing the expressive power, but you lose knowledge of whether or not the models will eventually stop
    - Model that does everything, but tradeoff in terms of how much you can understand.
- DFA -> DPDA -> PDA -> TM
- Acyclic graphs can be referred to as trees. Rooted trees mean that you can pick up a node and let everything else fall down, and you will have a tree.
- The length of a path is ambiguous because you could be counting nodes or edges.
- A computer follows some sort of program.

4/4 - Week 1

- A string can be a sequence of characters, but can also represent input to a computer. You can also represent execution, computational problems, and classes of problems as strings.
- Any finite set of symbols can be an alphabet.
    - Binary alphabet: 0,1
    - The literal alphabet: a, b, c, …
- A string is a finite sequence of zero or more symbols from the given alphabet.
- Length of a string is just the number of symbols.
- You cannot have epsilon as an alphabet symbol. If a word w = epsilon, then the length is 0.
- No real way to get a particular slice of a string after concatenation.
- Proper _ means that the result is a _ of the original but doesn't equal the original.
    - Prefixes of proc = p, pr, pro, and proc where proc is the only non-proper
    - Proper substring is any substring of the string which is not the original string.
- Epsilon is a substring of itself but is not a proper substring.
- Epsilon is a prefix/suffix of every string.
- A "run" is a sequence of symbols that are the same and adjacent to each other.
    - 0011101100011
    - 1 is not a run in the above, but 111 is.
- Language is a set of strings over an alphabet.
- A machine M will take in a string and give us a yes/no answer.

4/19 - Week 3

- Doing an inductive proof

    - Suppose k>= 1, then break problem(k) into pieces X,Y,Z and then do reasoning on the pieces and then put back together to show that problem(k) is true.

4/20 - Week 3

- Both union and intersection track pairs, and union accepts if one of the states is accepting and intersection only accepts if both are accepting.

Rest of notes in notebook

Textbook Notes

Chapter 0 - Introduction

0.1 Automata, Computability and Complexity

- Complexity theory deals with the question of what makes a problem computationally easy or difficult.
- No computer algorithm can determine whether a mathematical statement is true or false.
- In computability, we are concerned with whether or not a particular problem is solvable or not.
- Automata theory deals with the definitions and properties of mathematical models of computation.

0.2 Mathematical Notions and Terminology

- A is a proper subset of B if A is a subset of B and not equal to B.
- Sequence of objects is a list of these objects in some order.
- Power set of A is the set of all subsets of A.
- Function is an object that sets up an input output relationship.
- Function with k arguments is called a k-ary function and k is the arity of the function.
- Equivalence relations are those where the relation is reflexive (xRx for all x), symmetric (xRy implies yRx), and transitive.
- Number of edges at a particular node in a graph is the degree of that node.
- Graph G is a subgraph of graph H if the nodes of G are a subset of the nodes of H and the edges of G are the edges of H.
- A path in a graph is a sequence of nodes connected by edges. Simple path doesn't repeat any nodes. Graph is connected if every two nodes has a path between them. Path is a cycle if it starts and ends with the same node, and is simple if there are no repeats.
- Graph is a tree if it is connected and has no simple cycles.
- Directed path is one where all the arrows point in the same direction.
- Directed graph is strongly connected if a directed path connects every two nodes.

0.4 Types of Proof

- Proof is a convincing logical argument that a statement is true.
- Theorem is a mathematical statement proved true. Lemmas assist in the proof of another statement. Corollaries are statements that we can conclude from the proofs.
- For theorems that prove that a type of object exists, you can use proof by construction to show how to create the object.
- Proof by contradiction is to assume that a theorem is false and then show that this leads to an obviously false consequence and thus a contradiction.
- Proof by induction is used to show that all elements of an infinite set have a specified property.
    - Useful for when you have to prove that something is true in all cases. You have to have a basis and an induction step. Basically gotta show that P(1) is true and that if P(i) is true, then P(i+1) is true.
    - The assumption that P(i) is true is called the induction hypothesis.

Chapter 1 - Regular Languages

1.1 Finite Automata

- The simplest computational model is called a finite state machine. An example is when a controller is in one of many possible states, and there could be several events/inputs to take the agent from one state to another. This is shown through a state diagram.
- State diagrams can be used to introduce finite automata.
- A finite automation has
    - A **set of states** and **rules** for going from one state to another (transition function), depending on the input.
    - An **input alphabet** that indicates all the allowed symbols.
    - **Start state** and a set of **accepted states**
- Thus, a finite automation is a 5 tuple with the above 5 characteristics.
- Transition function has to specify exactly one next state for each possible combination of a state and an input symbol.
- The language of a machine is the set of things that the machine accepts.
- A machine M will accept some string w if every member in the string is a member of the alphabet, the machine starts in the start state, the machine goes from state to state according to the transition function, and we end up in an accept state.
- A language is called a regular language if some finite automaton recognizes it.
- One of the keys with this type of machine is that you have a finite number of states and you have a finite memory so you can't remember all of the inputs to the machine.
- If you have two languages A and B
    - Union will take all the strings in both A and B and lump them into one language.
    - Concatenation attaches a string from A in front of a string from B in al possible ways to get the strings in the new language.

    - Star is a unary operation that attaches any number of strings in A together in the new language.
- Collection of objects is closed under some operation if applying that operation to members of the collection returns an object still in the collection.
    - Class of regular languages is closed under the union operation. If A and B are regular languages, then A U B is regular too.
    - Class of regular languages is closed under the concatenation operation

1.2 Nondeterminism

- Deterministic computation is where we know what the next state of a machine given an input symbol is.
- In a nondeterministic machine, several choices may exist for the next state at any point.
- Every state in DFA (deterministic finite automaton) has exactly one exiting transition arrow for each input it could receive.
    - In NFA, a state may have 0, 1, or many exiting arrows for each alphabet symbol.
- If you come across multiple possibilities in NFA, then the machine spits into multiple copies of itself and follows all the possibilities in parallel. Then, at the end, if any of the copies is in the accept state, the NFA accepts the string.
- The empty input means that one copy will stay at the current state while another will go in the direction of that arrow.
- Every NFA can be converted into an equivalent DFA but sometimes that DFA will have a lot more states.
- NFA is represented as a 5 tuple as well but the transition function takes a state and an input or the empty string, and produces the set of possible next states.
- An NFA named N accepts some string w if every member in the string is a member of the alphabet, the machine starts in the start state, the next state is one of the allowable states, and we end up in an accept state.
- Deterministic and nondeterministic finite automata recognize the same class of languages.
- Two machines are equivalent if they recognize the same language.

1.3 Regular Expressions

- Regular expressions are those that can describe languages.
- In reg expressions, the star operation is done first, then concatenation, and then union.
- R is a regular expression is R is
    - A for some a in alphabet B
    - Empty string (Epsilon) - Represents language containing a single string
    - None - Represents language that doesn't contain any strings
    - (R1 U R2) where both are reg expressions
    - (R1 x R2) where both are reg expressions
    - (R1*) where R1 is reg expression
- Inductive definition is when we define _ in terms of smaller _.
- Regular expressions and finite automata are equivalent in their descriptive power.

- A language is regular if and only if some regular expression describes it.
- Converting DFAs into equivalent regular expressions is called a generalized nondeterministic finite automaton.
    - Nondeterministic because the transition arrows can have any regular expression as labels instead of only members of the alphabet or the empty string.
- GNFAs must always have
    - The start state has transition arrows to every other state but no arrows coming in from any other state.
    - Only one accept state and it has arrows in from every other state, but no arrows to any other state.
    - Accept state is not the same as the start state.
    - Except for start and accept, one arrow goes from every state to every other state and also from each state to itself.

Chapter 2 - Context Free Languages

2 - Intro

- We know that we can describe languages with finite automata as well as with regular expressions.
- Context free grammars give you another more powerful way to describe the languages that couldn't be described with finite automata.
    - Useful with languages with recursive structures.
- Languages associated with context free grammars are called context free languages.
- Pushdown automata are a class of machines that recognize context free languages.

2.1 - Context Free Grammars

- Grammar consists of substitution rules, set of variables, set of terminals, and a start variable.
- The sequence of substitutions to obtain a string is called a derivation. That derivation can also be shown with a parse tree.
- The strings generated by the specifications of the grammar constitute the language.
- A language that can be generated with a context free grammar is a context free language.
- Many CFLs are the union of simpler CFLs.
- If a grammar generates the same string in several different ways, then the string is said to have been derived ambiguously in that grammar.
    - If a grammar generates some string ambiguously, then the whole grammar is said to be ambiguous.
- Leftmost derivation is a derivation where the leftmost variable is the one that is replaced.
- Grammar G is ambiguous if it generates some string ambiguously, which means that is a string has two or more leftmost derivations.

- If a context free language can only be generated by ambiguous grammars, then those languages are inherently ambiguous.
- Chomsky normal form is where every rule is either A -> BC or A -> a or S -> epsilon where S is start variable.
- Any CFL is generated by a context free grammar in Chomsky normal form.

2.2 Pushdown Automata

- Pushdown automata are like NFAs but they also have a stack which provides additional memory.
    - This is what allows us to recognize some non regular languages.
- Stacks are valuable because they can hold an unlimited amount of info.
- Nondeterministic pushdown automata can recognize languages that deterministic pushdown automata can't recognize.
- The current state, the next input symbol, and the top symbol of the stack all influence the next move in the pushdown automata.
- PDA can't test for an empty stack but it can test for $ which is pretty much our symbol for the empty stack.
- Context free grammars and PDAs are equivalent in expression power.
- A language is context free only if some PDA recognizes it.
- Every regular language is context free.

2.3 Non Context Free Languages

- For some languages, we can use the pumping lemma to show that a language is not regular.
- Now, every context free language must have a special pumping length such that all longer strings in the language can be pumped. If that length can't be found, then the language is not context free.
- Basically, there is a pumping lemma for determining if a language is context free, similar to the one that we used to determine whether some language is regular.

2.4 Deterministic Context Free Languages

- Nondeterministic PDAs are more powerful than deterministic ones.
- Languages recognized by deterministic PDAs are deterministic CFLs.
- Class of DCFLs is closed under complement.
- For defining DCFGs, we use a bottom up approach where we go from terminals and then process the derivation in reverse, until we reach the start variable. This is called a reduction.
- The string that is replaced at every step in the reduction is called the reducing string.
- In leftmost reduction, each reducing string is reduced only after all other reducing strings that lie entirely to the left.
- The handle can be thought of as the part of the current string that is about to get replaced by a variable. Once we know the handle of the string, we know the next reducing step.

- Ambiguous grammars have strings where some of them have several handles. Which handle you choose will impact whether or not that string will be able to be recognized.
- A handle h of some valid string v = xhy is a forced handle if h is the unique handle in every valid string xhy' where y' can be any string in the language.
- A DCFG is a context free grammar such that every valid string has a forced handle.

Chapter 3 - The Church Turing Thesis

3.1 - Turing Machines

- Turing machine is like a finite automaton but it has an unlimited and unrestricted memory.
- The differences are:
    - Turing machine can write and read from tape.
    - Tape is infinite.
    - Special states for rejecting and accepting take effect immediately.
    - Read write head can move to left and right.