Day 10A: Midterm Review

- The Test
  - In-Class Exam, Thursday 3/18
  - Mixed Format Short Answer
- Content:
  - Some multiple choice
  - Some concept matching
  - Some short text answers (sentence not essay)
  - Problems to solve
  - A project in lisp, pseudolisp, or pseudopython
  - Based on lectures an Nota Bene readings

Outline

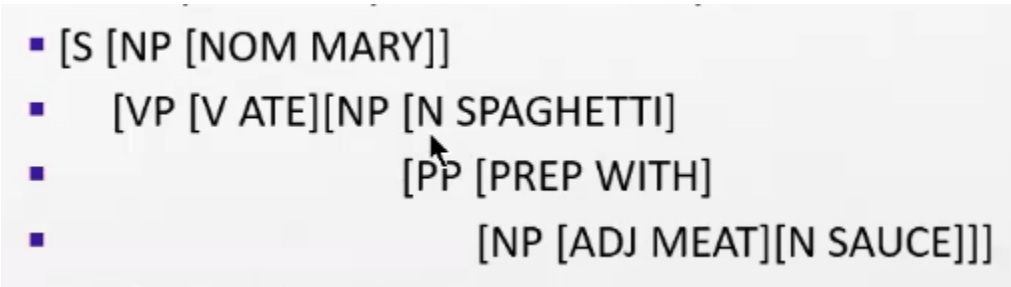
- Section 1: AI History & Challenges
  - Turing Test
  - Physical Symbol System
    - A machine which contains a collection of symbol structures and a program which can change those symbols over time. (ie. computers can change their minds over time)
  - Attempts at defining AI
    - Minksy, Schank (what me and my friends do, not FORTRAN, not Chomsky), Pollack (IIRC: I'll know it when I see it)
  - AI techniques compared to normal programs
  - Sample question was "what does AI stand for", which seems very easy.

- Section 2: Lisp/Pseudolisp/Python
  - Cons/Car/Cdr/List/Append
    - List is composed of cons cells, each of which has a car and a cdr, which can point to whatever
    - Know recursive definitions of some basic list functions, like append
  - Recursive function definitions
  - Let & lambda
  - Lisp loop macros
  - Sample questions are very easy: why is append defined recursively, what is the result of a list operation?
- Section 3: Problem Spaces
  - General Problem Solver Paper
    - Means-end analysis (MEA) — know the acronym
  - How to formalize problems as states, rules, and operators
  - Search direction
  - Classic puzzles (Hanoi, 8 Queens, Water Jug, etc.)
  - Sample Questions: Problem Spaces
    - Describe how to represent a problem
    - Describe the operators needed for a problem
    - How many states exist for a particular problem?
      - Permutations: $n!$, or $\frac{n!}{(n-m)!}$
      - Combinations: $\frac{{}_nP_r}{r!} = \frac{n!}{r!(n-r)!}$
  - Law of representation: Objects are mapped into an internal representation, operated on, then converted back, *and the results hold through that process*.
    - More compact/clever representations usually result in more complicated transition rules, and vis versa

- Weak Methods:
    - Brute force: generate every possible arrangement, loop through them all, check for the first one that's a legal solution.
    - Hill climbing/gradient descent: good in some cases, but bad in many others. Know the trade-offs
    - Breadth-First search
        - Takes a lot of storage and time, but always finds the shortest path
    - Depth-First search
        - Very fast, minimal space, but can loop (unless you have cycle checking). Better for dissipative problems
    - Best-First Search
        - Know that a perfect heuristic doesn't exist
    - Sample Question: know the data structures for, and be able to do, a BFS or DFS

- Section 4: NLP
    - Speech recognition/generation vs. NLP
    - Chomsky's grammar levels:
        - Finite State Machines/Regular languages
        - Phrase Structure Grammars/Context-free languages
        - Transformational Grammars/Context-sensitive languages
        - Turing-equivalent languages
            - First two insufficient for English
    - NLP layers: lexical, syntactic, morphological, semantic, dialog
    - Know about Eliza (robo-therapist) and it's limitations/what it's good at
        - Large database of patterns and possible responses
        - Small set of state variables to trigger certain prompts

    - Doesn't scale because of rule conflict, among other things
- Syntactic Parsing
- Conceptual Dependency
    - 11 primitive acts
    - Sample Question: Write a parse tree for a sentence (in lisp form, because you can't draw in COVID), interpret a CD structure
        - Lisp form: `(Type ...Children)`, ex:



- What's wrong with pattern recognition for NLP?
    - First 90% is easy, last 10% needs full AI
- Knowledge Representation
    - Formal Logic
        - Know the difference between deduction and abduction
        - Apply Demorgan's Law: `~(P & Q) = (~P | ~Q)`, `~(P | Q) = (~P & ~Q)`
        - Double Negation: `~~X = X`
        - Disjunctive Syllogism: `(A | B | C) & (D | ~B | E) = (A | C | D | E)`
        - Something form: a series of OR'd values, each group of ORs AND'd together
        - Resolve two clauses using resolution