Syllabus

Course Homepage: HLT Link

Homework Submission: on eLearning

Piazza: Piazza Link

Optional Textbook: *Artificial Intelligence: A Modern Approach* Russell and Norvig, 3rd edition.

Miscellaneous Notes:

1. Honorlock will be used for exams.
2. Don't email for class-related questions, use Piazza private posts.
3. Textbook is not required, only what is presented in lecture will be tested.

Prerequisites:

1. Python for projects, maybe C/C++ and Java
2. Big-O notation
3. Propositional and first-order logic (will have review lecture)
4. Elementary knowledge of probability theory (will have review lecture)

Collaboration Policy:

1. Assignments can be done in group of two
2. Term projects can be done in group of two
3. Only need to turn in one solution/program per group

Late Policy:

1. 6 free late days ($\leq$ 2 late days can be used per homework)
2. One day late is 10% penalty, two days late is 30%

Day 1

General

The main goal of the course is **problem solving and decision making**. There are two main types of problems:

- Problems do not require *sophisticated knowledge*

One example is the **Path Finding** problem. One way to solve the problem is to search through all possible paths and find the minimum. However, that is often not feasible because there are too many possible paths.

Another example is the **Map Coloring** problem. One way to solve the problem is to perform a complete (or exhaustive) search, which would take a very long time.

The focus for these search problems is to find efficient solutions to solving these difficult problems. Note that hardware has improved in the last 20 or 30 years, we do not care about those improvements because algorithmic improvements almost always outperform hardware improvements.

- Problems that do require *sophisticated knowledge*

One example is asking a computer "what is the Capital of California." Although this is easy for a human who had learned this knowledge before to answer, it is hard for a computer to know even if they are given the Wikipedia article for the state of California. Humans are able to process contextual information based on prior knowledge as well as comprehension level; since computers are not able to do that, humans must turn natural language into some kind of **knowledge representation** so that the resulting information can be easily understood by a machine.

The second challenge is to use **knowledge reasoning** to derive new knowledge from existing knowledge. However, deriving information from some statements that are only partially true requires **Probabilistic Reasoning** to understand the final probability of a different statement.

AI Goal

> **Intelligence**
> "the capacity to learn and solve problems"

Views of AI falls into fall different perspectives:

1. Thinking: being able to think
2. Acting: being able to do something that can be observed
3. Human: emotion might be involved in decision-making
4. Rational: decisions are always rational and ideal

1. Acting Humanly (Turing Test)

Turing proposed the question of "can machines act intelligently?" The Test consists of a test subject who is chatting with either a real human or a machine. After a certain period of time, if the test subject

can not tell if they're talking to the machine or the human, then the machine would have passed by the Turing Test.

Key components from the Turing Test:

1. Natural Language Processing: processing the user's input and respond intelligently
2. Knowledge Representation: store and manipulate information from knowledge base
3. Automated reasoning: used store info to draw conclusion to answer questions
4. Machine Learning: to adapt and extrapolate from existing data

One example of a "human impersonator" is *ELIZA*, which attempts to impersonate a psychotherapist. It was built in the 1960s, and used basic strategies to decide on what response is the most fitting. *ALICE*, which was built 30 years later, is a lot smarter and can intelligently answer questions with additional information from a knowledge base.

2. Thinking humanly: modeling cognitive processes

Cognitive Science studies the internal activities of the brain, and *Cognitive Neuroscience* attempts to mirror the human brain.

3. Thinking rationally: formalizing the "laws of thought"

Using logic to make the right inferences. The classic example is if Socrates is a man, and all men are mortal, then the logical conclusion is that "Socrates is mortal".

However, since not all decisions can be made by direct logical reasoning, logic has limited usage in AI.

4. Acting rationally: rational agent

Rational behavior means doing the right thing, or the best thing to **maximize goal achievement given the available information**, which might mean using a math textbook to score well on a math test.

Building Intelligent Machines

Two approaches

1. One approach is building an exact model of human cognition (main view from psychology and cognitive sciences)
2. Developing methods to match/exceed human performance, not necessary through emulating human behavior (main focus of this course as well as recent AI progress)

AI has many domains, and it also leverages the knowledge from a lot of different disciplines (philosophy, CS theory, mathematics/physics).

Historical Perspective

The goal is to obtain an understanding of the human mind. Major people include:

1. Formalizing the laws of human thought: George Boole, Gottlob Frege, and Alfred Tarski
2. Thinking as computation: Alan Turing, John von Neumann, and Claude Shannon

Direct founders;

1. John McCarthy, Marvin Minsky, Herbert, Simon, and Allen Newell

History of AI:

- 1943: 40-neuron network, model of artificial neurons
- 1950: Turing's "Computing machinery and intelligence"
- 1960s: Dartmouth meeting with founders
- 1950s: Early enthusiasm with great expectations for its future
- 1958: John McCarthy's LISP
- 1965: The **resolution principle**, which was a basis for automated theorem proving
  - At this time, the world is simplified and limited to very small number of states
- 1960s: "dose of reality", *NP-Completeness* (intractability of problems)
- 1970s: Expert systems that require a knowledge base (correlate symptoms with diseases)
- 1980s: Neural nets, LISP
- 2000-: focus on integration of reasoning

Achievements:

1. Microsoft '97, Answer Wizard

2. Deep Blue beating Russian chess grandmaster
3. 1999: *Proverb* Solving crossword puzzles
4. Robocup, Autonomous Vehicle

The recent progress is from the advancement of **machine perception**, which means that machines can see and hear from the outside world. Techniques that were developed by *assuming* outside input are now able to be applied into technology like autonomous driving. There is also advancement in both processing speed as well as memory space, which is getting close to humans' capabilities.

Other progress comes from crowd-sourced human data for training data, engineering teams at big companies, as well as research activity at many institutions. Recent achievements include Watson winning *Jeopardy!*, Google's AlphaGo, as well as Watson automating insurance claim workers' jobs.

Day 2: 1/25

Uninformed Search

Search is very important in AI, and researchers try to use humans' reasoning to design better searching algorithms. Since some problems like map-coloring is NP-complete (no efficient polynomial algorithms, must perform complete search), searching is crucial to finding a solution.

Defining a Search Problem

State space is described by the following:

1. **Initial state**: the first starting state
2. **Actions**: possible actions from each state
3. **Successor function**: given a state *x*, return a set of `<action, successor>` pairs.

A **path** is a sequence of states connected by a sequence of actions. A **goal test** determines if a state is the goal (target) state. **Path cost** is a function that assigns a cost to a path, useful if we need to determine the *shortest path* from many different paths. The assumption for search problems is that the cost of a path is the sum of the costs of each steps of the path (assume to be nonnegative).

Search Description Examples

The 8-Puzzle

Description: there are eight tiles on a 3x3 board, how many moves to get to target state?
State: the tiles on the board
Initial State: the initial board
Goal test: true if the board is the same as the target board, false otherwise
Successor function: move one tile to the free spot
Path cost: N/A

Cryptarithmetic

Description: solve `SEND+MORE=MONEY` if each letter is a digit.
State: digits assigned to each of the 8 letters
Initial State: "null", no values assigned yet
Goal test: true if the sum is correct, false otherwise
Successor function: assign the next unassigned letter from the rest of the available digits.

Generic Search Algorithm

```
L = make-quee(initial-state)
loop
    node = remove-front(L) // save to return as part of path to goal
    if goal-test(node) = true: return (path to goal)
    S = successors(node)
    insert(S,L)
until L is empty
return failure
```

The search algorithm generates a **search tree**, which has branches from each node to each of its successor nodes generated by the **successor function**. The search tree only cares about nodes that are relevant to finding the goal step, and we want the tree to be as small as possible to save memory space since the state space can be huge.

Fun fact: the set of leaves nodes currently in the search tree is the same set of nodes currently in the queue.

Node Data Structure

A **Node** contains more information than a **State**, because we might store additional information such as the parent node to return information about the final steps. This is because the goal step needs to go back to the root state to retrieve the final *goal path*. Other information could include path cost, path depth, and other information that might be useful for the search process.

Evaluating a Search Strategy

Completeness: is the strategy guaranteed to find a solution?

Time Complexity: how long does it take to find the solution?

Space Complexity: how much memory does the strategy require?

Optimality: does the strategy find the best (highest-quality) solution when there are several solutions.

Evaluating BFS
Completeness: Yes, because all nodes are explored
Optimal: Yes
Time Complexity: $O(b^{d+1})$ need to traverse at most $d+1$ levels where d is the target's level
Space Complexity: $O(b^{d+1})$ because all nodes must be kept until target has been found

Evaluating DFS
Completeness: Yes, up till a finite depth
Optimal: No (depth limit)
Time Complexity: $O(b^m)$ where m is the max depth
Space Complexity: $O(bm)$ because each route can be deleted after no goal is found

Advantages of **BFS**: Optimal solution

Advantages of **DFS**: Space efficient

Combining the two, we get **iterative deepening**, which we will explore next lecture.