Welcome

Class structures/rules (syllabus, HM, projects, tests)

What is AI?

What are the related fields of AI?

How was AI started? (history)

What is the state of the art of AI?
Wk1 Ch2: Intelligent Agents

- Show SuperBot movies and simulations
- **Project 1:** Description and assignment
- Agent/Robot
  - What is inside a robot/agent? (see, act, think, learn)
  - What does it seek? (goal, utility, solve problems, “fame/fortune”)
- Environment/world
  - What can be seen?
  - What can be changed?
  - Who else are there? (Obstacle and other agents)
- **HW:** Define an environment/world and a robot by yourself
How to represent a problem?
  * states, actions, initials, goals, (e.g., tic-tac-toe)

How to solve a problem?
  * Search: from here to there, initials to goals
  * Depth-first, breadth-first

How good is your solution? (fig 6.1, ALFE)
  * How good is your state? How costly is an action?
  * Best-first, Dynamic Programming, A*, etc.
  * Can you guarantee anything? (optimal vs heuristic)

How much do you want to pay for your solution?
  * How deep/wide can you go?
    * Predetermined vs dynamic (e.g., iterative deepening)
    * One way or many ways (bi-directional)?

How big is a problem? Can you put the whole world in your head?
  * Tower of Hanoi, chess, robot-and-world,

HM: state space for TOH, assign values for state and actions.
Wk2 Ch4: Optimization

- How to find the extremes of $f(x)$?
- Why is it so important?
- Why is it so hard? (no one has offered a general solution!)
  - Which way to go?
  - How much can you see? (local vs global)
  - How many points can you remember?
  - How big is your step? (skip THE point?)
  - How well can you guess?
  - How much do you know about the function?
  - How do you know you are done?
  - Will the function change by itself?
- **HW**: your own answers for the above questions
Key idea: optimization for yourself and your opponent

When you can see the whole game
  * Example, Tic-tac-toe (fig 5.1)
  * Optimal: Can I always win, or never lose?
  * Key: look ahead, chose the best
  * Minmax algorithm, Alpha-beta pruning
    * Fig 5.2, key idea: what can I gain if the opponent gives me the least

When cannot see the whole game (chess)
  * how far to look? how good is a state? Can you predict your opponent?

When there is a random player (dice)
  * Is random playing any good?

What are the best AI game players?

HW: Write a minmax algorithm for tic-tac-toe
Example: Map Coloring, can you think of another?

Formalism: Variables, Value Sets, Constraints,

  * Independent subset? Existing cycles?

The basic algorithms

  * Global Backtracking
    * Order variables: the most constrained variables first
    * Satisfy one at a time, if stuck, backtracking
  * Local search: heuristic, find the min-conflicts solution locally
  * Centralized vs. distributed (DCOP)

HW: Magic square, define it and solve it
The real world and its representation by logics

The world you created in simulation (e.g. wumpus)
- syntax (on paper, $\alpha, \beta$) $\neg, \land, \lor, \implies, \iff$
- semantics (in world/model $M$), ”satisfy” $\alpha \models \beta \iff M(\alpha) \subseteq M(\beta)$

Propositional logic: truth table for
- Theorem proving, inference rule, resolution, horn clause, forward/backward chaining,
- Propositional agent: current state, goal state, local search, global search

HW: PL rules for Wumpus or Tic-Tac-Toe
Wk4 Ch8: First-order logic

* Syntax and semantics
  * Add variables (predicates) and quantifiers
* Knowledge engineering
  * Capture and represent human knowledge, e.g., using FOL
* HW: FOL rules for Wumpus or Tic-Tac-Toe
Wk4 Ch9: Inference in FOL

* Modus Ponens with variables
  * Unification: finding the values for variables
* 1. Forward chaining
  * E.g. production systems, ACT, SOAR
* 2. Backward chaining
  * E.g., Prolog
* 3. Resolution
  * Convert to CNF, eliminating P, -P.
Project 2: Robot searches a target room on a map

Representing Actions
- Pre and post condition (single or set of states)

Searching for optimal solutions
- Graph-Based, Forward, Backward.
- Dynamic Programming, A*, Q-search (read ALFE 6.1)

Searching for satisfying solutions
- means-end-analysis
- Heuristic planning

Other planning methods
- Logic deduction: Boolean satisfactory, situation-calculus
- Constraint-satisfaction
- Partially ordered plans
Wk5 Ch11: Scheduling

* Why hard? Time, duration, resources, deadlines, orders, unknown effects, etc.
* Deterministic approaches
  * The longest first (critical path, take at least this long)
  * Hierarchical Planning
    * High-level actions first
    * Find primitive solutions
    * Find abstract solutions
* Non-deterministic approaches
  * Sensor-based, contingency, incremental,
  * Multi-agent planning (collaboration & coordination)
* HW: modify Wumpus as a scheduling problem
What is KR? Representation + Processes
Why important? Simon’s example cards/ddd
Format: Attribute-based and Relational-based
Ontology
- Objects: (attributes, part-of), both physical and mental
- Events (process, time),
- Relations, functions, operators: mostly mental
Reasoning
- Semantic net, logic, default (monotonic or not), truth maintenance
Example
- Work out an ontology for the Wumpus based on the example of Internet shopping in the book
Not all knowledge are certain

Earlier stories of expert systems (medical, legal)

Two big challenges

Common sense (vague): “Water flows down”

Uncertainty

Probability =?= Logic of Science
Where is probability? (in the world or in your mind)

Notation: variable $X$, value $x_i$, $P(X=x_i)$, $P(X)$ denotes for all values of $X$, $P(X,Y)$, $P(X,y)$

Remember two axioms
- **Sum axiom:** $P(A|B)+P(\neg A|B)=1$
- **Product axiom:** $P(AB|C)=P(A|C)P(B|AC)=P(B|C)P(A|BC)$

They can do much more than logics
- **Deductive reasoning:**
  - If $A\rightarrow B$ and $A$, then $B$
  - If $A\rightarrow B$ and $\neg B$, then $\neg A$
  - If $A\rightarrow B$ and $B$, then “$A$ become more plausible”
- **Inductive reasoning:**
  - If $A\rightarrow B$ and $\neg A$, then “$B$ become less plausible”
  - If $A\rightarrow B$ becomes more plausible” and $B$, then “$A$ become more plausible”

**HM:** work out the math why the above is true (ALFE, p102)
**Wk7 Ch14: Probability Reasoning**

- **Propositional logic + probability**
  - Full joint distribution (fig 13.3: truth table + probability)
  - Inferences
    - Using full joint distributions (1st part of product rule)
    - Marginalization or summing out
    - Using Bayesian rule (2nd part of product rule)
    - Example for Wumpus world
  - **Bayesian network**: topology + CPTs (fig 14.2)
    - Compare to truth-table format (2^5 rows)
    - Inferences, exact vs approximate

- **Relational and FOL + probability**

- **Other uncertain reasoning theories**
  - Dempster-Shafer, Fuzzy sets and logic,
Readings: AIMA 15

Examples: Speech recognition, see $=?$ truth

  * Given: a map of colored rooms, and an experience (act/see): $e_{1:t}$
  * Compute: which room the robot was/is/will-be in: $X$

States, observations, transitions, actions, sensors,

Definitions (ALFE 5.10)

We can infer:

  * $P(X_t|e_{1:t})$ which room I am in now (state estimation)
  * $P(X_{t+k}|e_{1:t})$ which room I will be in at time $t+k$ (prediction)
  * $P(X_k|e_{1:t})$ which room I was in at time $k$ (smoothing)
  * $P(x_{1:t}|e_{1:t})$ what rooms I have been in the past (explain)
  * $P(M_t|e_{1:t})$ can I improve my map (learning)

HW: Define an automata for your rooms in the simulation
Readings: AIMA 16, 17

- States have utility $U(s)$
- Maximum Expected Utility
  - $EU(a|e) = \text{Sum}[P(s|a,e)U(s)]$
- Markov Decision Process (MDP): state, action, transition $P(s'|s,a)$, reward: $s \rightarrow r$, or $(s,a) \rightarrow r$
  - Solutions are represented as policy: $s \rightarrow a$
  - Total discounted expected reward (Bellman):
    - Equation 17.5 or in ALFE page 167.
- Partially Observed MDP (POMDP)
  - States cannot be completely observed
  - Need a sensor model $P(z|s)$
- Objective: finding the optimal policy based on utilities
- Review for Midterm exam (if you have any questions)
- HW: Define your own world and robot using POMDP
Good luck on your test!
Wk11: Supervised Learning 1: Attribute-Based

* Types Readings: AIMA Ch18, ALFE 4

* Project 3: a robot learns a model of a colored room to find a target quickly

* Types of Machine Learning
  * Unsupervised vs. Supervised (examples, reinforcement, and environment)
  * Incremental vs. patch; online vs. offline

* Key Ideas: Hypotheses and examples
  * Hypotheses space (ALFE fig 4.3): Attribute-based vs relation-based
  * Learning = Search in hypothesis space for one to cover examples (both seen and unseen)
  * False negative (cover too little) $\Rightarrow$ Generalization
  * False positive (cover too much) $\Rightarrow$ Specialization
  * Correctness criteria: Overfitting, Identical, PAC, and others

* Algorithms for Attribute-Based Learning
  * Version Space
  * Decision Trees (HW:)
  * Neural Network (fig 18.19 and ALEF 4.11) (HW:)
  * Linear Regression: Learn $h=ax+b$ from examples $(x,y)$ by minimizing $\Sigma(y-h)^2$ [eq 18.2-3]
  * Support vector machines (HW: 18.16, 18.25)
  * Ensemble Learning and Boosting
    * Learn a new hypothesis based on the results of the last hypothesis (weight the examples)
Wk12: Supervised Learning 2
Relation-Based

* Readings: AIMA 19, ALFE 4.7-4.10
* A general challenge: a good hypothesis space
  * Too open: too slow to converge
  * Too restrict: rule out the good hypotheses
  * Good priori knowledge would help
* Relation-based learning
  * Deductive (e.g. EBL): use priori knowledge to explain examples → new hypotheses (fig 19.7)
  * Inductive algorithms (ALFE 4.7-4.10)
    * FOIL
    * CDL (Complementary Discrimination Learning)
* HW: learn the relational rules of X from examples
Wk13: Unsupervised Learning Probability-Based

* Readings: AIMA 20, ALFE 5.10
* Clustering
  * Naïve Bayes Models
  * By Bayesian Approach (Cheeseman’s CLASSIFY)
  * K-mean algorithm (see Andrew Ng’s lecture)
* The EM algorithm
  * Clustering with mixtures of Gaussians (fig 20.11)
  * Bayesian network with hidden variables (fig 20.13)
  * Hidden Markov models (fig 20.14)
  * PO-MDP (ALFE 5.10)
* HW: Learn a Bayesian model from examples
Week 13: Reinforcement Learning

- Reinforcement Learning
- Delayed supervision/reward
- When we only know the rewards from the goals
Wk14: Surprise-Based Learning

* Lecture by Dr. Nadeesha Ranasinghe
Wk15: Integrated Intelligence

* Actions and Perceptions
* Communication
  * Natural language overview
  * Speech understanding
* Collaboration
* Self-Organization and self-reconfiguration
  * Rubenstein 2010.
Mk16: Towards the Future

- Relational + Probabilistic Models
  - Existing research: FOL + Probability
- Structural Learning
  - Existing research: SBL,
- Scale up to the real world
- Integrate with all aspects
- Reconfigure both mind and body
Mk17: Final Exam

* Good Luck!
Extra Lectures

* See following slides
Support vector machines (HW: 18.16, 18.25)

- The key idea of max-margin separator (fig 18.30)
- Support vectors: the points closest to the separator, all other data points have zero weights
- When the data are not linearly separable (fig 18.31), rewrite separator in a higher dimension
- It is non-parametric, but (eq. 18.13) can use fewer examples than the training data