Midterm Study Guide

Midterm Time and Place:
- Thursday, March 10, 1:10 – 2:25pm
- Sinclair Auditorium (our usual room)

Format:
The test will be held in class. You can expect the following types of questions: true/false, short answer, and smaller versions of homework problems. It will be closed book and closed notes. However, you may bring one 8 ½ x 11” “cheat sheet” with handwritten notes on one-side only. Also, all calculators, PDAs, portable audio players (e.g., iPods) and cell phones must be put away for the duration of the test.

Coverage:
In general, anything from the assigned reading or lecture could be on the test. In order to help you focus, I have provided a partial list of topics that you should know below. In some cases, I have explicitly listed topics that you do not need to know. In addition, you do not need to reproduce the pseudo-code for any algorithm, but you should be able to apply the principles of the major algorithms to a problem as we have done in class and on the homework.

- Ch. 1 – Introduction
  - rationality
  - definitions of “artificial intelligence”
  - The Turing Test
  - you do not need to know:
    - dates and history
- Ch. 2 - Agents
  - PEAS descriptions
    - performance measure, environment, actuators, sensors
  - properties of task environments
    - fully observable vs. partially observable, deterministic vs. stochastic vs. strategic, episodic vs. sequential, static vs. dynamic, discrete vs. continuous, single agent vs. multiagent, known vs. unknown
  - agent architectures
    - simple reflex agents, goal-based agents, utility-based agents
  - state representations
    - atomic, factored, structured
  - you do not need to know:
    - learning agents
- Ch. 3 – Search
  - problem description
    - initial state, actions, transition model, goal test, path cost/step cost
  - tree search
    - expanding nodes, frontier
    - branching factor
  - graph search
    - explored set
  - uninformed search strategies
    - breadth-first, depth-first, uniform cost
    - similarities and differences / benefits and tradeoffs between strategies
    - evaluation criteria
      - completeness, optimality, time complexity, space complexity
• best first search
  ■ evaluation function
• informed search
  ■ heuristics
  ■ greedy best-first, A*
  ■ admissible heuristics
  ■ similarities and differences / benefits and tradeoffs between strategies
• you do not need to know:
  ■ depth-limited, iterative deepening or bidirectional search
  ■ details of proof that A* is optimal if h(n) is admissible
  ■ memory bounded heuristic search
  ■ learning heuristics from experience

• Ch. 5 - Game playing (Sect. 5.1-5.2, 5.4, 5.7-5.9)
  o two-player zero-sum games
  o problem description
    ■ initial state, actions, transition model, terminal test, utility function
  o minimax algorithm
  o optimal decision vs. imperfect real-time decisions
  o evaluation function, cutoff-test
  o you do not need to know:
    ■ alpha-beta pruning
    ■ forward pruning
    ■ details of any state-of-the-art game playing programs

• Ch. 7 – Logical Agents (Sect. 7.1-7.4, 7.5.3-7.5.4, 7.7-7.8)
  o knowledge-based agents
    ■ TELL, ASK
  o propositional logic
    ■ syntax and semantics
  o entailment, models, truth tables
  o model checking
  o inference procedures
    ■ forward-chaining
    ■ backward-chaining
    ■ sound, complete
  o you do not need to know:
    ■ details of the Wumpus world
    ■ circuit-based agents

• Ch. 8 – First-Order Logic
  o syntax and semantics
    ■ be able to translate English sentences into logic sentences
  o quantification
    ■ existential, universal
  o domain, model, interpretation
  o equality/inequality
    ■ making statements about quantity (e.g., exactly two brothers)
  o you do not need to know:
    ■ specific axioms from the domains given in class or the book
• “Intro to Prolog Programming” Reading, Ch. 1
  o syntax
    ▪ be able to write rules and facts in Prolog
    ▪ translating to FOL and vice versa
  o you do not need to know:
    ▪ negation as failure / closed world assumption
    ▪ backward-chaining, depth-first search