CSCI–4150 Introduction to Artificial Intelligence, Fall 2004
Final examination information & topics

Final examination information

The final examination is on Friday December 17 from 3:00–6:00pm in DCC 318. You may feel free to bring food as long as you clean up after yourself. The examination is closed book and closed notes. No calculators are allowed or necessary; you will probably have to do a little simple arithmetic[1].

The examination will be designed to test both:

- conceptual understanding — the ideas behind the algorithms, which algorithm to apply to a problem and what the tradeoffs are
- detailed understanding — the intricacies of how an algorithm works and issues in its implementation.

There will be some questions involving factual recall or simple explanation of concepts or algorithms, but there will also be questions asking you to apply the course material to various problems and situations. You may be asked to extrapolate from course material or apply related concepts to new problems; I think one characteristic of a good examination is that students should learn something from it.

The exam will have around 6–8 sections, each with roughly 3–6 questions. Each section focuses on a single topic (e.g., constraint satisfaction or neural networks). The questions in each section may be independent short-answer questions, they may work through an application of an algorithm, or some combination.

I will release on the web page the midterm examinations for this class from Fall 1999 and 2000. This is so that you can get a feel for the format of the exam. I have never released previous years' final exams before; I will reconsider this, but I wouldn't count on it. I highly recommend reviewing your quizzes.

Formulas provided on the final examination

- Information

$$I(P(v_1), \ldots, P(v_n)) = \sum_i -P(v_i)\log_2 P(v_i) \qquad \text{(p. 659)}$$

- Bayes classifiers (see slides or online reference)

$$v = \operatorname*{argmax}_{v_j \in V} \sum_{h_i \in H} P(v_j|h_i)P(h_i|D)$$

$$v = \operatorname*{argmax}_{v_j \in V} P(v_j)\prod_i P(a_i|v_j)$$

$$h = \operatorname*{argmax}_{h_i \in H} P(D|h_i)P(h_i)$$

- Sequential decision problems & reinforcement learning:

$$U(s) = R(s) + \gamma \max_a \sum_{s'} T(s,a,s')U(s') \qquad (17.5)$$

$$U^\pi(s) \leftarrow U^\pi(s) + \alpha(R(s) + \gamma U^\pi(s') - U^\pi(s)) \qquad (21.3)$$

$$U(s) = \max_a Q(a,s) \qquad (21.6)$$

$$Q(a,s) \leftarrow Q(a,s) + \alpha(R(s) + \gamma \max_{a'} Q(a',s') - Q(a,s)) \qquad (21.8)$$

- Perceptron learning

$$\vec{W} \leftarrow \vec{W} + \alpha \times \vec{I} \times Err$$

$$W_j \leftarrow W_j + \alpha \times Err \times g'(in) \times x_j \qquad (20.12)$$

- Backpropagation (see handout and text)

$$\vec{W}_j^{i,n} \leftarrow \vec{W}_j^{i,n} + \alpha \times \vec{a}_j \times \Delta_{i,n}$$

$$\vec{W}_k^{j,m} \leftarrow \vec{W}_k^{j,m} + \alpha \times \vec{a}_k \times \Delta_{j,m}$$

$$\Delta_{i,n} = Err_{i,n} \times g'(in_{i,n})$$

$$\Delta_{j,m} = g'(in_{j,m}) \sum_{n=1}^{r} W_{j,m}^{i,n} \times \Delta_{i,n}$$

$$g(x) = \frac{1}{1+e^{-x}}$$

$$g'(x) = g(1-g)$$

[1] This means either: multiplying a one digit and a two digit number and adding two digit numbers; or working with fractions whose numerator and denominator are one or two digits. If you find yourself doing more complicated arithmetic, you are probably doing something wrong. (Or should use fractions instead of decimal numbers.)

Final examination topics

Introduction	
What is AI?	1.1
Agent structure & environments	2

Search	
Blind search	
Formulating search problems	3.1–2
State space versus search tree	3.3
Optimality, completeness, time & space complexity	3.3
Six blind searches	3.4
Avoiding repeated states	3.5
Heuristic search	
Greedy search	4.1
A* search search	4.1
Heuristic functions	4.2
Admissibility, monotonicity/consistency	4.1–2
Memory bounded A* algorithms	4.1
Iterative improvement algorithms	4.3
Hill climbing	
Simulated annealing	
Local beam search	
Genetic Algorithms	
Constraint satisfaction problems	
CSP & assignment problems	5.1
Constructive approaches	5.2
Blind search approaches	
Backtracking, forward checking	
Heuristics to improve blind search strategies	
Constraint propagation	
Repair approaches	5.3
Min-conflicts heuristic	
Game playing search	
MINIMAX search	6.1–2
Perfect vs. imperfect decisions	6.4
Evaluation functions	6.4
Alpha-beta pruning	6.3
Probabalistic games (EXPECTIMAX)	6.5

Logic	
Knowledge representation & logical systems	7.1–3
Inference & entailment	
Soundness & completeness	
Propositional logic	7.4
Horn normal form	7.5
Conjunctive & implicative normal forms	7.5
Inference in propositional logic	7.5
Forward and backward chaining	
Resolution refutation proofs with set of support strategy	

Logic, continued	
First order logic	7.1–3
Quantifiers, Inference in first order logic	9.1
Horn normal form	
Conjunctive normal form	9.5
Skolemization	
Unification	9.2
Forward and backward chaining	9.3–4
Resolution refutation proofs with set of support strategy	9.5

Learning	
Introduction	18.1–2
Classification problems	
Decision trees	18.3–4, slides
Basic algorithm, information gain heuristic	
Dealing with missing attribute values	
Overfitting	
χ^2 pruning	
Gain ratio	
Rule post pruning	
Bayesian learning/classifiers	slides, (20.1–2)
Probability basics	13
Conditional independence	
Bayes rule	
Brute force classifier	
Optimal classifier	
Naive classifier	
Reinforcement learning	
Introduction	21.1
Utility	16.1–3
Sequential decision problems	17.1–3
Value iteration	
Policy iteration	
Passive reinforcement learning	21.2
Direct utility estimation	
Adaptive dynamic programming	
Temporal differencing	
Active reinforcement learning	21.3
Exploration	
Q-learning	
Neural networks	20.5
Perceptrons	
Perceptron learning rule	
Representational power of perceptrons	
Multilayer feed-forward networks	
Sigmoid units	
Backpropagation	20.5 & handout
Representational power	

Key algorithms/techniques

The following are the most likely algorithms and techniques to be the subject of a question in which you are asked to solve a specific problem by applying that algorithm or technique.

Blind search
Breadth first search
Depth first search
Depth-limited search
Iterative deepening search
Uniform cost search
Bi-directional search

Heuristic search
Greedy search
A* search

Constraint satisfaction search
Constructive methods (with forward checking, constraint propagation, and heuristics)
Heuristic repair (with min-conflicts heuristic)

Game playing search
MINIMAX
MINIMAX with alpha-beta pruning

Logic
Translating/transforming into logic sentences and into normal forms
Forward chaining
Backward chaining
Resolution refutation proof with set of support strategy

Decision trees
Decision tree learning with information gain heuristic
Dealing with missing attribute values
Rule post-pruning

Artificial neural networks
Perceptron learning rule
Backpropagation

Bayes classifiers
Bayes naive classifier

Reinforcement learning
Value iteration
Policy iteration
Temporal differencing
Q-learning