CS472
Foundations of Artificial Intelligence

Final Exam Solutions

December 19, 2003

Search (and a bit o' machine learning)

1. A* becomes best-first search when f(n) = h(n), i.e. when g(n) is assumed to be 0.
2. BFS is a uniform-cost search with all step costs equal to 1 (or some positive constant).
3. A* is optimal when h(n) is admissible for all nodes n. f must also be monotonically increasing along any path to the goal, which is easy to ensure as long as we disallow negative path costs.
4. **states** Each state is a (hypothesis for the best) decision tree.

   initial state Empty decision tree, i.e. a decision tree with no nodes.

   goal state/test The "goal" decision tree is one that covers all of the training examples and has homogeneous leaves w.r.t. class value.

   successor function The successor function transforms one decision tree hypothesis into another (more specific) hypothesis. It (1) selects a non-homogeneous leaf; (2) creates n successors, one for each remaining attribute; (3) each successor tree replaces the non-homogeneous leaf with a decision node for the associated attribute, partitioning the the remaining instances according to their values along the selected attribute.

   heuristic function Each state (i.e. each resulting tree) is evaluated w.r.t. information gain. The tree with the highest IG is selected as the next current state.

Planning (and knowledge representation)

1. At (child, A)
   At (candy, B)
   At (stool, C)
   Height (child, low)
   Height (stool, low)
   Height (candy, high)
2. S_0 is the initial situation. All axioms are fluents, and therefore need to include the situation argument.

   At (child, A, S_0)
   At (candy, B, S_0)
   At (stool, C, S_0)
   Height (child, low, S_0)
   Height (stool, low, S_0)
   Height (candy, high, S_0)

3. At (Agent, Loc, s) $\wedge$ At (Object, Loc, s) $\wedge$ Height (Agent, Ht, s) $\wedge$ Height (Object, Ht, s) $\wedge$ $\neg$ Holding (Agent, Object, s) $\rightarrow$ POSS *GRASP* (Agent, Object, s)

   GRASP (Agent, Object, s) $\rightarrow$ Holding (Agent, Object, Result(Grasp (Agent, Object, s)))

4. Action (Go (agent,from,to),
   precond: At (agent, from)
   effect: At (agent, to) $\wedge$ $\neg$ At (agent, from)

5.
```
START
  | effects: [list of all axioms from 1]
  |
  |
  | precond: Holding(child, candy)
  v
FINISH
```

6. Lots of possibilities here for a legitimate answer...

7. True. (See p. 378 R&N.) The STRIPS semantics embodies the "STRIPS assumption" — every literal not mentioned in an action's effect remains unchanged in the subsequent state. This obviates the need for defining many frame axioms that describe those portions of the state representation that stay the same after each action.

Machine learning I

1. Weights are all 1 and threshold is 2.5. No hidden nodes needed.

2. Solution not shown. There are a number of ways to do this...

3. For the conjunction to be true, every input that is associated with a positive literal has to be a 1 and every input associated with a negative literal has to be a 0. Weights on positive literal links are all 1; weights on all negative literal links are all -1. Threshold is p-1.

4. Sketch of the answer...

   Why: simulated annealing can converge to a global minimum, whereas simple error back-propagation (i.e. gradient descent) can converge only to a local minimum. Other global search algorithms will have unreasonable memory requirements.

   How: successor chosen randomly; e.g. from ϵ-sphere in weight space. The energy function should be the error function. the annealing schedule may need to be adjusted based on progress.

Game playing

1. Right (guarantees an 88)

2. Left (guarantees a 17)

3. The heuristic function values change (i.e. become more accurate) as search proceeds.

4. Can prune the 22 node (and its children).

5. Yes. Alpha-beta only prunes parts of the search space that cannot possibly influence the the final decision.

6. Chance nodes are inserted into game trees to represent any element of chance (e.g. a dice roll, card selection) that can affect a player's choice of moves.

7. $expectiminimax(n) = \sum_{successors\ s} P(s)\ minimax(s)$ where n is a chance node and $P(s)$ is the probability of each outcome of the dice roll, card selection, etc.

Machine learning II

1. **Reinforcement learning**.

   (a) Utility of (1,3) becomes a 1. Explanation: Because (1,3) was never seen before, the naive updating algorithm updates the associated table of frequencies for states to 1, and the reward-to-go obtained for the state is a 1, so the new utility value is a 1.

   (b) Upon seeing (1,3) during the trial, the utility of (1,3) is set to a 0, the reward for the state. Upon seeing (2,3), U(1,3) is updated according to the TD-lambda rule:
   $U(1,3) \leftarrow U(1,3) + \alpha(reward(1,3) + U(2,3) - U(1,3))$
   $U(1,3) \leftarrow 0 + 0.1(0 + -0.8 - 0)$
   $U(1,3) \leftarrow 0 - 0.08 = -0.08$

2. **Genetic algorithms.** No solution provided.

Logic and Knowledge-based Systems

1. A reasonable answer: If two people speak the same language then they understand each other.

2. $Understands(x, y)$ means that x understands y; $Friend(x, y)$ means x is a friend of y.

   (a) $\forall x, y\ Understands(x, y) \Rightarrow Friend(x, y)$

   (b) $\forall x, y, z\ Friend(x, y) \wedge Friend(y, z) \Rightarrow Friend(x, z)$

3. KB converted to clausal form:
   A1: $\neg SL(x, l) \vee \neg SL(y, l) \vee Und(x, y)$
   A2: $\neg SL(a, m) \vee \neg SL(b, m) \vee Und(a, b)$
   B1: $\neg Und(c, d) \vee Friend(c, d)$
   B2: $\neg Friend(e, f) \vee \neg Friend(f, g) \vee Friend(e, g)$
   C1: SL(Ann, French)
   C2: SL(Bob, French)
   C3: SL(Bob, German)
   C4: SL(Cal, German)

   Prove: Friend(Ann, Cal). So add its negation to the KB.
   P: $\neg Friend(Ann, Cal)$

   Resolution proof:

resolve C1 and A1 with {x=Ann; l=French} to get

N1: $\neg SL(y, French) \vee Und(Ann, y)$

resolve C2 and N1 with {y=Bob} to get

N2: Und(Ann,Bob)

resolve C3 and A2 with {a=Bob; l=German} to get

N3: $\neg SL(b, German) \vee Und(Bob, b)$

resolve C4 and N3 with {b=Cal} to get

N4: Und(Bob,Cal)

resolve N2 and B1 with {c=Ann; d=Bob} to get

N5: Friend(Ann,Bob)

resolve N4 and B1 with {c=Bob; d=Cal} to get

N6: Friend(Bob,Cal)

resolve N5 and B2 with {e=Ann; f=Bob} to get

N7: $\neg Friend(Bob, g) \vee Friend(Ann, g)$

resolve N7 and N5 with {g=Cal} to get

N8: Friend(Ann,Cal)

resolve N8 and P to get

NIL

So Ann must be a friend of Cal.

Natural Language Processing

1. perception
2. semantic interpretation
3. parsing
4. pragmatics