Draft Solutions

In the solutions below “A” refers to Version A of the test; “B” refers to Version B of the test.

State Space Search (20 points)

greedy best-first (5 pts.)

1. $A - B - D - J$
2. $A - B - D - J$

iterative deepening (5 pts.)

1. $(A-) A - B - C$
   
   $A - B - D - E - F - C - G - H$
   
   $A - B - D - J$

2. $A - B - D - J$

uniform-cost (5 pts.)

1. $A - C - B - H - F - E - G - D - 0$
   
   $-K - M - S - J$

2. $A - B - D - J$

A* (5 pts.)

1. $A - C - H - G - B - E - F - D - K - J$

2. $A - B - D - J$

Local Search (20 points)

1. The main disadvantage of hill-climbing is getting stuck at a local minimum/maximum. One method for alleviating this is to conduct a series of hill-climbing searches from randomly generated initial states, running each until it halts or makes no discernible progress, and returning the best of the results from each of the individual searches. Another solution is random-restart hill-climbing which restarts the hill-climbing search at a new random state after a pre-defined number of local steps. (Other answers are possible here.)

2. Simulated annealing allows steps away from the goal. As time passes, the probability of the downhill step decreases and the size of the downhill step decreases.

3. heuristic search and local search

4. GA($Fitness, Fitness\_threshold, p, r, m$)
   
   - $P \leftarrow$ randomly generate $p$ individuals
   - For each $i$ in $P$, compute $Fitness(i)$
• While \([\max_i \text{Fitness}(i)] < \text{Fitness Threshold}\)
  1. Probabilistically select \((1 - r)p\) members of \(P\) to add to \(P_S\).
  2. Probabilistically choose \(\frac{m}{2}\) pairs of individuals from \(P\). For each pair, \((i_1, i_2)\), apply crossover and add the offspring to \(P_s\).
  3. Mutate \(m \cdot p\) random members of \(P_s\)
  4. \(P \leftarrow P_s\)
  5. For each \(i\) in \(P\), compute \(\text{Fitness}(i)\)
• Return the individual in \(P\) with the highest fitness.

Adversarial Search (20 pts)

1. The values are:
   \(N = 7, G = 3, D = 10, B = 10, J = 0, K = 11, F = 11, C = 11, A = 11\)
   The player should take move C.

2. The first example of alpha-beta pruning shows allows nodes U and V to be pruned.

3. Yes.

4. On the order of \(b^d\).

5. On the order of \(b^{d/2}\). The branching factor at maximizing levels is \(b\); but at minimizing levels, it is 1.

First-Order Logic and Resolution Theorem-Proving (20 points total)

1. (a) Every dog owner is an animal lover. \(\forall x (\exists y \ Dog(y) \land \text{Owns}(x, y)) \rightarrow \text{AnimalLover}(x)\)
   (b) Jack owns a dog. \(\exists x : \text{Dog}(x) \land \text{Owns}(\text{Jack}, x)\)
   (c) No animal lover kills an animal. \(\forall x \ \text{AnimalLover}(x) \rightarrow \forall y \ \text{Animal}(y) \rightarrow \neg \text{Kills}(x, y)\)
   (d) Tuna is a cat. \(\text{Cat}(\text{Tuna})\)
   (e) Either Jack or Curiosity kills Tuna. \(\text{Kills}(\text{Jack}, \text{Tuna}) \lor \text{Kills}(\text{Curiosity}, \text{Tuna})\)
   (f) Cats are animals. \(\forall x \ \text{Cat}(x) \rightarrow \text{Animal}(x)\)

2. \(\text{(1) } \neg \text{Dog}(S(x)) \lor \neg \text{Owns}(x, S(x)) \lor \text{AnimalLover}(x)\) (provided)
   \(\text{(2a) } \text{Dog}(\text{D}) \quad (D \text{ is the function that finds Jack's dog})\)
   \(\text{(2b) } \text{Owns}(\text{Jack}, D)\)
   \(\text{(3) } \neg \text{Dog}(S(x)) \lor \neg \text{Owns}(x, S(x)) \lor \text{AnimalLover}(x)\) (provided)
   \(\text{(4) } \text{Cat}(\text{Tuna})\)
   \(\text{(5) } \text{Kills}(\text{Jack}, \text{Tuna}) \lor \text{Kills}(\text{Curiosity}, \text{Tuna})\)
   \(\text{(6) } \neg \text{Cat}(z) \lor \text{Animal}(z)\)

3. (a) Add \(\neg \text{Animal}(\text{Tuna})\) to the KB as statement 7.
   (b) Resolve statements 7 and 6 (unifier: \(z = \text{Tuna}\)) to get \(\neg \text{Cat}(\text{Tuna})\) (which becomes statement 8).
   (c) Resolve statements 8 and 4 to produce nil/fail.