SOLUTIONS: Artificial Intelligence

1. (a) i. The clausal form for this sentence is the following:
$\{\neg P(S_1, S_2), \neg Q(S_2, S_1), R(x, f(x)), S(f(x), z)\}$
where S_1, S_2 are skolem constants and f is a skolem function.

ii. In general, does the procedure that you use in part i above result in clauses that preserve the validity of a sentence?
No. The clausal form conversion procedure described in Manna and Waldinger's *The Deductive foundations of Computer Programming* is a validity preserving transformation; it is based on the notion of the *existential closure* of a sentence.
Does it preserve the satisfiability of a sentence? [2 Points]
Yes. To prove the unsatisfiability of a set of sentences, we prove the unsatisfiability of the sentences in clausal form.
Note that if you chose the procedure described in Manna and Waldinger's book, the answer to the first part of the question would be yes, and that to the second part would be no. Our answers here assume that you use the procedure described in LFAI.

(b) i. Since u has to be made equal to A, all the variables have to be made equal to A. An mgu then is: $\{u \leftarrow A, w \leftarrow A, x \leftarrow A, y \leftarrow A, z \leftarrow A\}$.

ii. This set is not unifiable, since x would have to be made equal to $F(x, y)$, and the *occurs check* fails.

iii. An mgu for this set is:
$\{z \leftarrow G(A), y \leftarrow F(G(A), A, B), x \leftarrow G(F(G(A), A, B))\}$

(c) C_1. $\{Skier(x), Climber(x)\}$
C_2. $\{\neg Likes(x, Rain), \neg Climber(x)\}$
C_3. $\{\neg Skier(x), Likes(x, Snow)\}$
C_4. $\{\neg Likes(Mike, x), \neg Likes(Tom, x)\}$
C_5. $\{Likes(Mike, x), Likes(Tom, x)\}$
C_6. $\{Likes(Tom, Rain)\}$
C_7. $\{Likes(Tom, Snow)\}$
C_8. $\{\neg Climber(x), Skier(x)\}$

A resolution with set of support refutation for this set of clauses, where $\{C_8\}$ is the set of support, is the following.

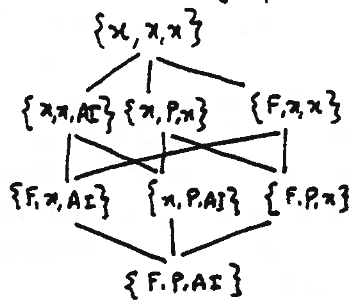
C_9: $\{Skier(x)\}$	C_8, C_1, Factoring.
C_{10}: $\{Likes(x, Snow)\}$	C_9, C_3.
C_{11}: $\{\neg Likes(Tom, Snow)\}$	C_{10}, C_4.
C_{12}: $\{\}$	C_{11}, C_7.

2. LEARNING

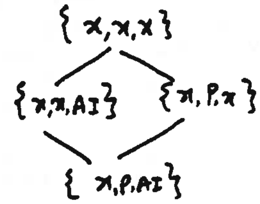
15 Points

a] [i] We start with the version graph with Sphere as the sole +ve instance.

This graph is:

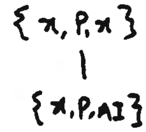


. Next, with GEB as another +ve instance, we prune the relations restricted to F. The graph now is:



With LFAI there's no change to this graph.

Next, with Terminal Man as a −ve instance, we remove those nodes inconsistent with it so we are left with.

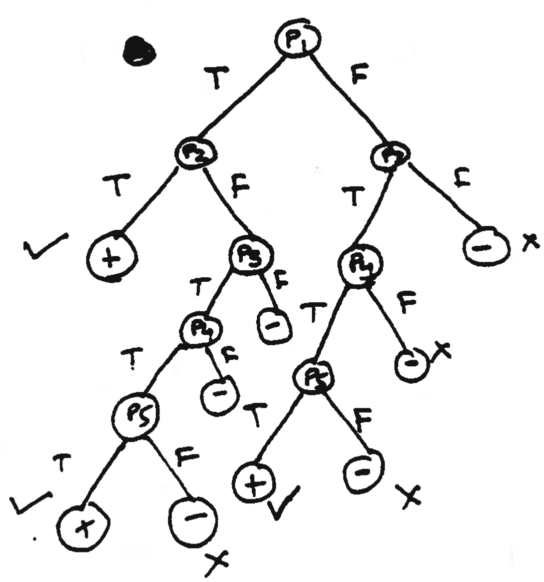


. Finally, with Airport, we remove the node incosistent with subject Air Tra

& our final version graph is → {x, P, AI}

[ii] In this case the candidate elimination algorithm converges to a single concept. Both the most general and the most specific concepts consistent with this set are {x, P, AI}, i.e. all AI paperba

b] One such decision tree is:



Any tree rooted on either P3, P4 or P5 must strictly be larger than this tree, because the subtrees to test for {P1(x)∧I the subtrees to test for {P1(x)∧I must be a child of each of P3, P4 & P5 nodes.

3. SEARCH

15 Points

(a) i. A, D, B, H, F, L.

ii. - For cutoff = 1, 2, only expand A.
- For cutoff = 3, expand A and D.
- For cutoff = 4, expand A, D, B, H, F, L.

(b) In hill climbing search we examine only the nodes that are directly reachable, i.e. one step away, from the last examined node. This gives hill climbing a depth first flavor. In best-first search, the unexplored nodes that were encountered earlier in the search are considered as well. This gives best-first search a breadth-first flavor.

4. PROBABILISTIC REASONING

[10 Points]

(a)

$$O(H|E_1) = \frac{p(H|E_1)}{P(\neg H|E_1)} = \frac{p(E_1|H)p(H)}{p(E_1|\neg H)p(\neg H)} = \frac{(.8)(.1)}{(.2)(.9)} = \frac{4}{9}$$

(b)

$$O(H|E_1 \wedge E_2) = \frac{p(H|E_1 \wedge E_2)}{P(\neg H|E_1 \wedge E_2)} = \frac{p(E_1 \wedge E_2|H)p(H)}{p(E_1 \wedge E_2|\neg H)p(\neg H)} = \frac{p(E_1|H)p(E_2|H)p(H)}{p(E_1|\neg H)p(E_2|\neg H)p(\neg H)}$$

$$= \frac{p(E_2|H)}{p(E_2|\neg H)} O(H|E_1) = \frac{.75}{.6} \times \frac{4}{9} = \frac{5}{9}$$