CS221 Practice Final

Autumn 2012

1 Other Finals

The following pages are excerpts from similar classes' finals. The content is similar to what we've been covering this quarter, so that it should be useful for practicing. Note that the topics and terminology differ slightly, so feel free to ignore the questions that we did not cover.

Certain topics are less emphasized in the past exams, but will be **more emphasized** in the final for the class. These include:

- Weighted CSPs and Markov Nets (the practice exams place more of an emphasis on Bayes Nets).
- Loss-based learning (the practice exams place an emphasis on Naive Bayes instead).
- Unsupervised learning (e.g., EM)
- Logic (covered in much greater depth in our class)

In contrast, the practice exams cover state space models fairly deeply. State space models will be **less emphasized** in the final for the class.

The first portion of the practice exam comes with solutions; the rest are provided as example problems, but without solutions. In terms of other miscellaneous notes:

- *Perceptron* refers to a classifier using the perceptron loss (see slide 34 in the lecture on loss minimization).
- The *forward* (and *backward*) algorithm for HMMs is just an instance of variable elimination, as you did in the first part of your Pacman projects, before implementing particle filtering. Relatedly, *Viterbi* is an algorithm to decode the MAP estimate in an HMM.

1. (17 points.) Search: A* Variants

Queuing variants: Consider the following variants of the A* *tree search* algorithm. In all cases, g is the cumulative path cost of a node n, h is a lower bound on the shortest path to a goal state, and n' is the parent of n. Assume all costs are positive.

(i) Standard A*

(ii) A*, but we apply the goal test before enqueuing nodes rather than after dequeuing

(iii) A*, but prioritize n by $g(n)$ only (ignoring $h(n)$)

(iv) A*, but prioritize n by $h(n)$ only (ignoring $g(n)$)

(v) A*, but prioritize n by $g(n) + h(n')$

(vi) A*, but prioritize n by $g(n') + h(n)$

(a) (3 points) Which of the above variants are complete, assuming all heuristics are admissible?

Solution: All but (iv) are complete if costs are positive, none are complete if costs can be zero (we accepted both answers). For a tree search to be incomplete, there must be some constant c such that there is an infinite path in the tree where all nodes have priority less than c. That can't happen with positive costs and g in the priority. (iv) is greedy search, which is clearly not complete for $h = 0$ in general infinite graphs.

(b) (3 points) Which of the above variants are optimal, again assuming all heuristics are admissible?

Solution: (i) is optimal, as proven in class. (ii) is suboptimal, as shown by an example in lecture (e.g. when the last arc of a suboptimal is really expensive but the prefix is cheap). (iii) is UCS, a subcase of (i). (iv) is greedy, which we saw was suboptimal in class. (v) is effectively an inadmissible heuristic, so not optimal. (vi) has the same problem as (ii).

Upper Bounds: A* exploits lower bounds h on the true completion cost h^*. Suppose now that we also have an *upper bound* $k(n)$ on the best completion cost (i.e. $\forall n, k(n) \geq h^*(n)$). We will now consider A* variants which still use $g + h$ as the queue priority, but save some work by using k as well. Consider the point at which you are inserting a node n into the queue (fringe).

(c) (3 points) Assume you are required to preserve optimality. In response to n's insertion, can you ever delete any nodes m currently on the queue? If yes, state a general condition under which nodes m can be discarded, if not, state why not. Your answer should involve various path quantities (g, h, k) for both the newly inserted node n and other nodes m on the queue.

Solution: You can delete any m where $g(m) + h(m) > g(n) + k(n)$. For these m, there is no way they can lead to a solution better than n will lead to.

In a *satisficing* search, you are only required to find *some* solution of cost less than some threshold t (if one exists). You need not be optimal.

(d) (3 points) In the satisficing case, in response to n's insertion, can you ever delete any nodes m currently on the queue? If yes, state a general condition, if not, state why not. Your answer should involve various path quantities (g, h, k) for both the newly inserted node n and other nodes m on the queue.

Solution: If $g(n) + k(n) < t$, then n definitely leads to a satisficing solution, and you can delete all other m.

ϵ-Admissible Heuristics: Suppose that we have a heuristic function which is not admissible, but ϵ-*admissible*, meaning for some known $\epsilon > 0$,

$$h(n) \leq h^*(n) + \epsilon \qquad \text{for all nodes } n$$

where $h^*(n)$ is the optimal completion cost. In other words, h is never more than ϵ from being optimal.

(e) (1 point) Is using A* with an ϵ-admissible heuristic complete? Briefly justify.

Solution: Yes, at least under the conditions where A* normally is (i.e. costs bounded below). Infinite paths eventually have unboundedly bad priorities, as above. In particular, consider a path to a goal G. Each node on that path has a finite priority, and there are a finite number of other nodes which could have that priority or smaller.

(f) (2 points) Assuming we utilize an ϵ admissible heuristic in standard A* search, how much worse than the optimal solution can we get? I.e., if c^* is the optimal cost for a search problem, what is the worst cost solution an ϵ admissible heuristic would yield? Justify your answer.

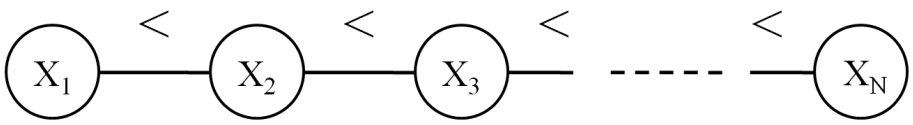
Solution: Whatever suboptimal goal G you pop first obeys $g(G) \leq f(G)$ because heuristics are non-negative, $f(G) \leq f(G^*)$ because we're popping G first, and $f(G^*) = g(G^*) + h(G^*) \leq c^* + \epsilon$.

(g) (2 points) Suggest a modification to the A* algorithm which will be guaranteed to yield an optimal solution using an ϵ-admissible heuristic with fixed, known ϵ. Justify your answer.

Solution: Option 1: keep popping after first goal G until you pop a node of $f(n) > f(G) + \epsilon$. Option 2: use $h'(n) = \max(h(n) - \epsilon, 0)$.

2. (13 points.) CSPs: A Greater (or Lesser) Chain

Consider the general less-than chain CSP below. Each of the N variables X_i has the domain $\{1 \dots M\}$. The constraints between adjacent variables X_i and X_{i+1} require that $X_i < X_{i+1}$.



For now, assume $N = M = 5$.

(a) (1 point) How many solutions does the CSP have?

Solution: Just one: 1, 2, 3 ...

(b) (1 point) What will the domain of X_1 be after enforcing the consistency of *only* the arc $X_1 \rightarrow X_2$?

Solution: $\{1, 2, 3, 4\}$

(c) (2 points) What will the domain of X_1 be after enforcing the consistency of *only* the arcs $X_2 \rightarrow X_3$ then $X_1 \rightarrow X_2$?

Solution: $\{1, 2, 3\}$

(d) (2 points) What will the domain of X_1 be after fully enforcing arc consistency?

Solution: $\{1\}$

Now consider the general case for arbitrary N and M.

(e) (3 points) What is the minimum number of arcs (big-O is ok) which must be processed by AC-3 (the algorithm which enforces arc consistency) on this graph before arc consistency is established?

Solution: $O(MN)$. You will have to process many arcs multiple times, one for each domain value.

(f) (4 points) Imagine you wish to construct a similar family of CSPs which forces one of the two following types of solutions: either all values must be ascending *or* all values must be descending, from left to right. For example, if $M = N = 3$, there would be exactly two solutions: $\{1, 2, 3\}$ and $\{3, 2, 1\}$. Explain how to formulate this variant. Your answer should include a constraint graph and precise statements of variables and constraints.

Solution: Several good answers. One is have trinary constraints on each adjacent triple that the triple should be either an increasing or decreasing triple. The overlap between triples enforces that the choice be global. Another is to introduce a global variable indicating ascent or descent and have trinary constraints between adjacent nodes and the global one, allowing, for example, triples like $(1, 2, <)$ or $(2, 1, >)$.

3. (18 points.) RL and MDPs: Two-Armed Bandit

Imagine you have two slot machine levers. You are playing a game where at each time step, you must pull exactly one lever. Lever A always pays a reward of 6. Lever B pays a reward of either 10 or 0. If B is a lucky lever (L=ℓ), it pays 10 with probability 4/5. If it is an unlucky one (L=$\neg\ell$), it pays 10 with probability 1/5. B is equally likely to lucky or unlucky a priori. Assume $\gamma = 1$ (which is ok for finite games, not a trick).

(a) (2 points) If you can only pull a lever once, what is the MEU?

Solution: 6. Lever A pays 6, lever B pays 10 with probability 1/2*4/5 + 1/2*1/5 = 1/2, and 0 with probability 1/2*1/5 + 1/2*4/5 = 1/2, for an expectation of 5.

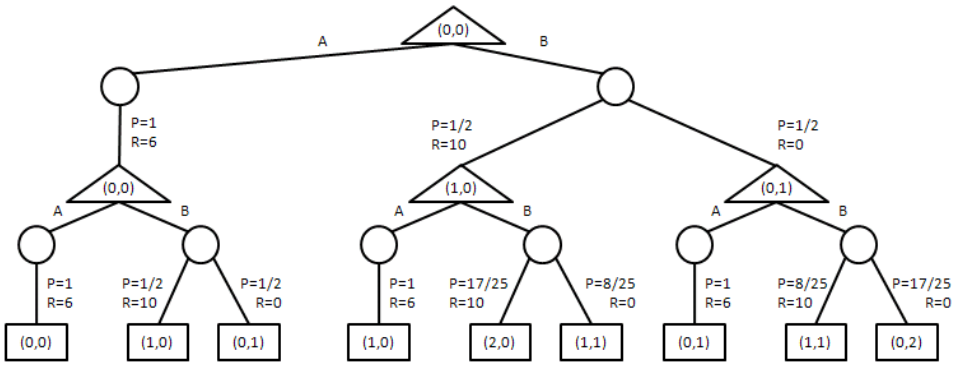
(b) (1 point) Which action(s) (A or B or both) give that MEU?

Solution: A

If you play this game multiple times, it becomes more difficult to figure out what actions to take. The game is formally a POMDP, but we can turn it into an MDP in which the states encode our past outcomes from lever B. In particular, a state will be of the form (m, n), where m is the number of times we pulled B and got 10 and n is the number of times we pulled B and got 0. We begin in state $(0, 0)$. If we then pull lever B and get the outcome 10 we will go to state $(1, 0)$, while getting the 0 outcome puts us in state $(0, 1)$. Your actions are {A, B}, and the rewards are as described above.

(c) (3 points) If you will play exactly two rounds, draw the computation tree which represents the possible outcomes of the MDP. Clearly indicate which nodes are of which type (min, max, expectation, etc).

Solution:



Note that if you pull lever B, the resulting payoff should change your beliefs about what kind of lever B is, and therefore what future payoffs from B might be. For example, if you get the 10 reward, your belief that B is lucky should increase.

(d) (2 points) If you are in state $(0, 1)$ and select action B, list the states you might land in and the probability you will land in them.

Solution: Because your one experience was the bad outcome, the posterior probability of ℓ is $1/5$. $(0, 2)$ with probability $4/5 \cdot 4/5 + 1/5 \cdot 1/5 = 17/25$. $(1, 1)$ with probability $4/5 \cdot 1/5 + 1/5 \cdot 4/5 = 8/25$.

(e) (2 points) If you are in state $(1, 0)$ and select action B, list the states you might land in and the probability you will land in them.

Solution: Because your one experience was the good outcome, the posterior probability of ℓ is $4/5$. $(2, 0)$ with probability $4/5 \cdot 4/5 + 1/5 \cdot 1/5 = 17/25$. $(1, 1)$ with probability $4/5 \cdot 1/5 + 1/5 \cdot 4/5 = 8/25$.

(f) (3 points) On the computation tree in (c), clearly mark the probabilities on each branch of any chance nodes.

(g) (3 points) Again in this two-round setting, what is the MEU from the state state, and which first action(s) (A or B or both) give it?

Solution: 12, given by A. But B is much closer than before, at 11.4.

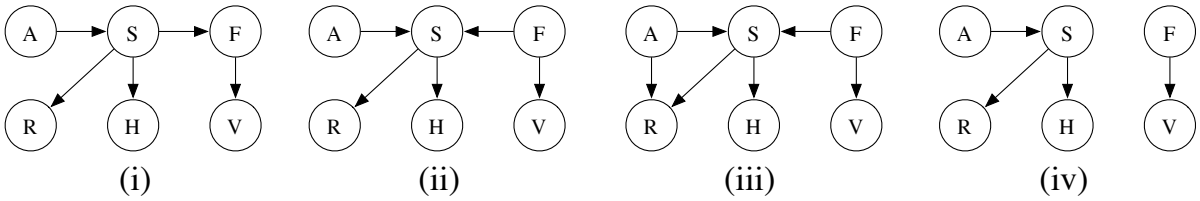
(h) (2 points) If the number of plays N is large enough, the optimal first action will eventually be to pull lever B. Explain why this makes sense using concepts from reinforcement learning.

Solution: Exploration is necessary to improve knowledge of true payoff rates. In the long run, it is worth a possible suboptimal trial of B to determine whether B is lucky. The improved knowledge has a chance to be exploited over time.

4. (15 points.) Bayes Nets: Snuffles

Assume there are two types of conditions: (S)inus congestion and (F)lu. Sinus congestion is is caused by (A)llergy or the flu.

There are three observed symptoms for these conditions: (H)eadache, (R)unny nose, and fe(V)er. Runny nose and headaches are directly caused by sinus congestion (only), while fever comes from having the flu (only). For example, allergies only cause runny noses indirectly. Assume each variable is boolean.



(a) (2 points) Consider the four Bayes Nets shown. Circle the one which models the domain (as described above) best.
Solution: (ii)

(b) (3 points) For each network, if it models the domain exactly as above, write *correct*. If it has too many conditional independence properties, write *extra independence* and state one that it has but should not have. If it has too few conditional independence properties, write *missing independence* and state one that it should have but does not have.

Solution: (many others are possible)

(i) Missing independence: $A \perp F$

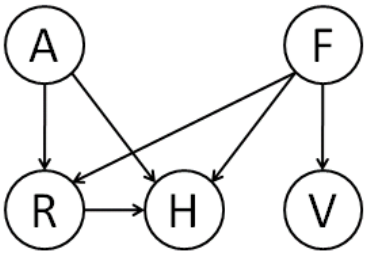
(ii) Correct

(iii) Missing independence: $A \perp R \mid S$

(iv) Extra independence: $F \perp A \mid S$

(c) (3 points) Assume we wanted to remove the Sinus congestion (S) node. Draw the minimal Bayes Net over the remaining variables which can encode the original model's marginal distribution over the remaining variables.

Solution: Note that there is an induced arrow between R and H. Also, other options are possible, such as the reverse of that arrow.



(d) (2 points) In the original network you chose, which query is more efficient to compute using variable elimination: $P(F|r, v, h, a, s)$ or $P(F)$? Briefly justify.

Solution: $P(F|r, v, h, a, s)$ because no nodes need to be eliminated. However, it was also acceptable to say that $P(F)$ is faster if you pre-process by deleting leaf nodes (in which case, only F is left). Note that is it also correct that $P(F)$ is given to you (though variable elimination doesn't exploit that unless you refer to the leaf node deletion process.

Assume the following samples were drawn from prior sampling:

$$a, s, r, \neg h, \neg f, \neg v$$
$$a, s, \neg r, h, f, \neg v$$
$$a, \neg s, \neg r, \neg h, \neg f, \neg v$$
$$a, \neg s, \neg r, h, f, \neg v$$
$$a, s, \neg r, h, \neg f, \neg v$$

(e) (1 point) Give the sample estimate of $P(f)$ or state why it cannot be computed.

Solution: 2/5

(f) (1 point) Give the sample estimate of $P(f|h)$ or state why it cannot be computed.

Solution: 2/3

(g) (1 point) Give the sample estimate of $P(f|v)$ or state why it cannot be computed.

Solution: Cannot be computed because no samples relevant to the query have been generated yet.

(h) (2 points) For rejection sampling in general (not necessarily on these samples), which query will require more samples to compute to a certain degree of accuracy, $P(f|h)$ or $P(f|h, a)$? Justify your answer in general terms.

Solution: $P(f|h, a)$ is worse because the evidence h, a can be no more common than the evidence h; therefore equal or more samples will be rejected for $P(f|h, a)$.

5. (19 points.) HMMs: Tracking a Jabberwock

You have been put in charge of a Jabberwock for your friend Lewis. The Jabberwock is kept in a large tugley wood which is conveniently divided into an $N \times N$ grid. It wanders freely around the N^2 possible cells. At each time step $t = 1, 2, 3, \ldots$, the Jabberwock is in some cell $X_t \in \{1, \ldots, N\}^2$, and it moves to cell X_{t+1} randomly as follows: with probability $1 - \epsilon$, it chooses one of the (up to 4) valid neighboring cells uniformly at random; with probability ϵ, it uses its magical powers to teleport to a random cell uniformly at random among the N^2 possibilities (it might teleport to the same cell). Suppose $\epsilon = \frac{1}{2}, N = 10$ and that the Jabberwock always starts in $X_1 = (1, 1)$.

(a) (2 points) Compute the probability that the Jabberwock will be in $X_2 = (2, 1)$ at time step 2. What about $P(X_2 = (4, 4))$?

Solution:
$P(X_2 = (2, 1)) = 1/2 \cdot 1/2 + 1/2 \cdot 1/100 = 0.255$

$P(X_2 = (4, 4)) = 1/2 \cdot 1/100 = 0.005$

At each time step t, you don't see X_t but see E_t, which is the row that the Jabberwock is in; that is, if $X_t = (r, c)$, then $E_t = r$. You still know that $X_1 = (1, 1)$.

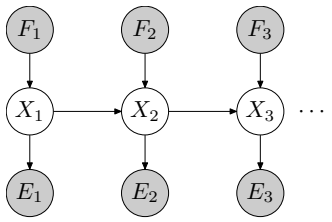
(b) (4 points) Suppose we see that $E_1 = 1, E_2 = 2$. Fill in the following table with the distribution over X_t after each time step, taking into consideration the evidence. Your answer should be concise. *Hint: you should not need to do any heavy calculations.*

Solution: (Note: we cut the previously-present third row when grading and gave credit to all.)

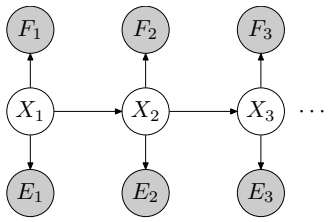
t	$P(X_t, e_{1:t-1})$	$P(X_t, e_{1:t})$
1	$(1,1) : 1.0$, (others) $: 0.0$	$(1,1) : 1.0$, (others) $: 0.0$
2	$(1,2), (2,1) : 51/200$, (others) $: 1/200$	$(2,1) : 51/200$, $(2, 2+) : 1/200$

You are a bit unsatisfied that you can't pinpoint the Jabberwock exactly. But then you remembered Lewis told you that the Jabberwock teleports only because it is frumious on that time step, and it becomes frumious independently of anything else. Let us introduce a variable $F_t \in \{0, 1\}$ to denote whether it will teleport at time t. We want to to add these frumious variables to the HMM.

Consider the two candidates:



(A)



(B)

(A)	(B)
$+\ X_1 \perp X_3 \mid X_2$	$X_1 \perp X_3 \mid X_2\ +$
$+\ X_1 \perp E_2 \mid X_2$	$X_1 \perp E_2 \mid X_2\ +$
$\quad X_1 \perp F_2 \mid X_2$	$X_1 \perp F_2 \mid X_2\ +$
$+\ X_1 \perp E_4 \mid X_2$	$X_1 \perp E_4 \mid X_2\ +$
$+\ X_1 \perp F_4 \mid X_2$	$X_1 \perp F_4 \mid X_2\ +$
$+\ E_3 \perp F_3 \mid X_3$	$E_3 \perp F_3 \mid X_3\ +$
$\quad E_1 \perp F_2 \mid X_2$	$E_1 \perp F_2 \mid X_2\ +$
$\quad E_1 \perp F_2 \mid E_2$	$E_1 \perp F_2 \mid E_2$

(c) (3 points) For each model, circle the conditional independence assumptions above which are true in that model.

(d) (2 points) Which Bayes net is more appropriate for the problem domain here, (A) or (B)? Justify your answer.

Solution: (A) because the choice of X depends on F in the problem description.

For the following questions, your answers should be fully general for models of the structure shown above, not specific to the teleporting Jabberwock. For full credit, you should also simplify as much as possible (including pulling constants outside of sums, etc.).

(e) (2 points) For (A), express $P(X_{t+1}, e_{1:t+1}, f_{1:t+1})$ in terms of $P(X_t, e_{1:t}, f_{1:t})$ and the CPTs used to define the network. Assume the E and F nodes are all observed.

Solution: $P(x_{t+1}, e_{1:t+1}, f_{1:t+1}) = P(e_{t+1}|x_{t+1})P(f_{t+1}) \sum_{x_t} P(x_{t+1}|x_t, f_{t+1})P(x_t, e_{1:t}, f_{1:t})$

(f) (2 points) For (B), express $P(X_{t+1}, e_{1:t+1}, f_{1:t+1})$ in terms of $P(X_t, e_{1:t}, f_{1:t})$ and the CPTs used to define the network. Assume the E and F nodes are all observed.

Solution: $P(x_{t+1}, e_{1:t+1}, f_{1:t+1}) = P(e_{t+1}|x_{t+1})P(f_{t+1}|x_{t+1}) \sum_{x_t} P(x_{t+1}|x_t)P(x_t, e_{1:t}, f_{1:t})$

Suppose that we don't actually observe the F_ts.

(g) (2 points) For (A), express $P(X_{t+1}, e_{1:t+1})$ in terms of $P(X_t, e_{1:t})$ and the CPTs used to define the network.

Solution: $P(x_{t+1}, e_{1:t+1}) = P(e_{t+1}|x_{t+1}) \sum_{f_{t+1}} P(f_{t+1}) \sum_{x_t} P(x_{t+1}|x_t, f_{t+1}) P(x_t, e_{1:t})$

(h) (2 points) For (B), express $P(X_{t+1}, e_{1:t+1})$ in terms of $P(X_t, e_{1:t})$ and the CPTs used to define the network.

Solution: $P(x_{t+1}, e_{1:t+1}) = P(e_{t+1}|x_{t+1}) \sum_{x_t} P(x_{t+1}|x_t) P(x_t, e_{1:t})$

6. (18 points.) Classification and VPI: Cat Cravings

Consider the following Naive-Bayes model for diagnosing whether your cat is (H)ungry. Signs of hunger include that the cat is (T)hin, (M)eowing, or (W)eak.

H	$P(H)$
h	0.5
¬h	0.5

H	T	$P(T\|H)$
h	t	0.6
h	¬t	0.4
¬h	t	0.4
¬h	¬t	0.6

H	M	$P(M\|H)$
h	m	0.6
h	¬m	0.4
¬h	m	0.4
¬h	¬m	0.6

H	W	$P(W\|H)$
h	w	0.5
h	¬w	0.5
¬h	w	0.0
¬h	¬w	1.0

(a) (3 points) If your cat is thin and meowing, but not weak, what is the probability that he is hungry?

Solution: $P(t, m, w, h) = 0.5 \cdot 0.6 \cdot 0.6 \cdot 0.5, P(t, m, w, \neg h) = 0.5 \cdot 0.4 \cdot 0.4 \cdot 1.0$, so normalizing $P(h|t, m, w, h) = 18/34$.

(b) (2 points) Which of the following smoothing options *might* have been applied to produce the CPTs above from training data? Assume non-extreme values of hyperparameters. Circle the best answer:

(i) Laplace smoothing only might have been applied

(ii) Linear interpolation only might have been applied

(iii) (x) Neither could have been applied

(iv) Either might have been applied

Solution: Any smoothing would have resulted in a non-zero value for $P(w|\neg h)$.

(c) (2 points) Assume that no smoothing has been applied (so these are the maximum likelihood estimates). Compute the linear interpolation smoothed estimate of $P_{\text{LIN}}(w|h)$ using $\alpha = 0.5$.

Solution: In the ML parameters, $P(w) = \sum_h P(w|h)P(h) = 0.25$, so $P_{\text{LIN}}(w|h) = 0.5 \cdot 0.5 + 0.5 \cdot 0.25 = 0.375$

(d) (2 points) In a single word, state why smoothing is necessary.

Solution: Many answers: best: overfitting, generalization; ok: accuracy, sampling, zeros...

Imagine you cannot tell whether your cat is weak or not.

(e) (2 points) Is it correct to simply skip over any unobserved evidence variables when classifying in a Naive Bayes model? That is, will you get the same answer as if you had marginalized out the missing nodes? Briefly justify why or why not.

Solution: Yes; from homework we know we can delete dangling nodes before inference in a Bayes Net, including Naive Bayes.

Now return to the original probabilities, reprinted here:

H	$P(H)$
h	0.5
¬h	0.5

H	T	$P(T\|H)$
h	t	0.6
h	¬t	0.4
¬h	t	0.4
¬h	¬t	0.6

H	M	$P(M\|H)$
h	m	0.6
h	¬m	0.4
¬h	m	0.4
¬h	¬m	0.6

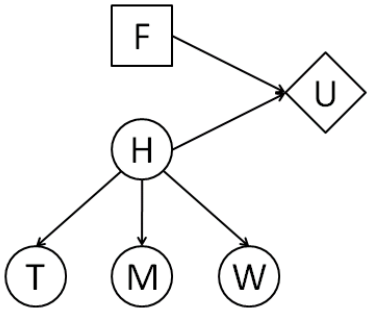
H	W	$P(W\|H)$
h	w	0.5
h	¬w	0.5
¬h	w	0.0
¬h	¬w	1.0

You can decide whether or not to give your cat a mega-feast (F) to counteract his (possible) hunger. Your resulting utilities are below:

H	F	$U(H,F)$
h	f	0
h	¬f	-100
¬h	f	0
¬h	¬f	10

(f) (2 points) Draw the decision diagram corresponding to this decision problem.

Solution:



If you do not know W, but wish to determine whether your cat is weak, you can apply the weak-o-meter test, which reveals the value of W.

(g) (3 points) In terms of *high-level quantities* (MEUs, EUs, conditional probabilities, or similar) and variables, give an expression for the maximum utility you should be willing pay to apply the weak-o-meter, assuming the cat is again thin and meowing?

Solution: Simplest answer was: $VPI_{t,m}(W)$, which expands to $\sum_w P(w|t,m)\,[MEU(t,m,w) - MEU(t,m)]$.

(h) (2 point) What is the maximum utility you should be willing to pay, as a specific real number?

Solution: 0. The MEU decision is to give the feast either way, so the evidence has no value.