Written by (anonymous ID): ____________________

Written re-examination
TIN173/DIT410, Artificial Intelligence

Wednesday 1 June 2016, 8:30–12:30

Examiner: Peter Ljunglöf, 031–772 1065

This examination consists of six questions. A correctly answered question gives you 2 points, the total number of points is 12.

Grades: To get grade 3/G/pass you need at least 66% correct, i.e., 8 points.

This is only for students from previous years:

To get Chalmers grade 4 you need at least 10 points.

To get GU grade VG/distinction you need at least 11 points.

To get Chalmers grade 5 you need all 12 points.

Tools: Paper and pencil.

No extra books, papers or calculators.

Notes: Answer every question directly on the question paper, and write your ID number at the top of every paper.

If you have any additional papers with associated calculations, you should hand them in too.

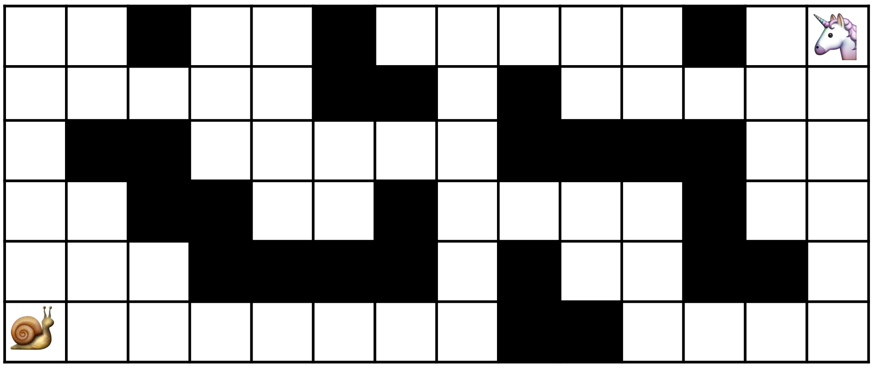
Write legibly, and explain your answers!

http://www.smbc-comics.com/index.php?id=4124

1. Best friends forever

The snail and the unicorn are best friends but are trapped in a maze such as the one below. To not feel too alone they want to meet up – they don't care where, but they want to meet in as few turns as possible. At each turn both move at the same time, and each of them moves *exactly one* square up, down, left or right.

Neither of them can enter (or jump over) any walls (i.e., the black squares), but they have GPS and mobile phones, so they know where each other is and can cooperate in finding the best path.



A natural representation of the search states in this problem is as a 4-tuple (x_s, y_s, x_u, y_u), where x_s, y_s are the (integer) coordinates for the snail and x_u, y_u are the coordinates for the unicorn.

a) What is the maximal branching factor of this search problem? (disregarding the size and form of the maze)

correct?

Both the unicorn and the snail has 4 options each,
so the maximal branching factor is 4 x 4 = 16

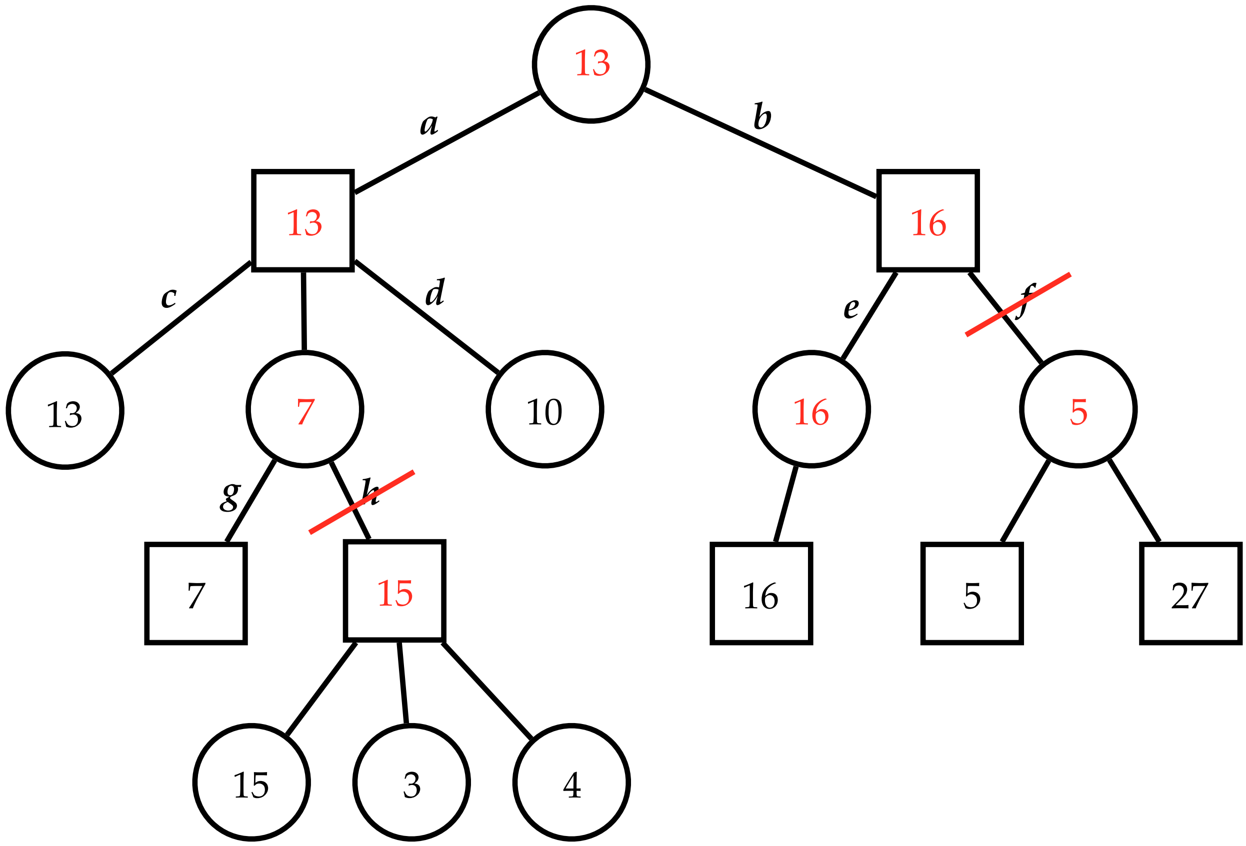
b) Which of the following heuristic functions are admissible for this problem?

correct?

- ☐ $|x_s - x_u| + |y_s - y_u|$ (the Manhattan distance)
- ☐ $2 \cdot (|x_s - x_u| + |y_s - y_u|)$ (twice the Manhattan distance)
- ☑ $½ \cdot (|x_s - x_u| + |y_s - y_u|)$ (half the Manhattan distance)
- ☐ $\sqrt{(x_s - x_u)^2 + (y_s - y_u)^2}$ (the Euclidean distance)
- ☐ $2 \cdot \sqrt{(x_s - x_u)^2 + (y_s - y_u)^2}$ (twice the Euclidean distance)
- ☑ $½ \cdot \sqrt{(x_s - x_u)^2 + (y_s - y_u)^2}$ (half the Euclidean distance)

2. Minimax and alpha-beta pruning

Assume the following zero-sum game tree,
where ◯ are the minimising nodes, and ☐ are the maximising nodes:



a) Perform the minimax algorithm on the game tree above, and write the resulting min/max values inside the empty nodes.

correct?

See above

b) Suppose you had used alpha-beta pruning, which branches would have been cut off from the game tree?

correct?

☐ *a* ☐ *b* ☐ *c* ☐ *d*

☐ *e* ☑ *f* ☐ *g* ☑ *h*

3. Classifying search algorithms

a) What are the ***advantages*** of using iterative deepening search (IDS), as opposed to depth-first search (DFS) and breadth-first search (BFS)? (Write one advantage per algorithm).

correct?

IDS vs. DFS:

IDS will always find a solution if there is one,
while DFS can get stuck in an infinite loop

(also, IDS will find an optimal solution, if all edge costs are equal)

IDS vs. BFS:

IDS uses linear space, while BFS has exponential space usage
(in the length of the shortest path)

b) Classify the following search algorithms according to the given criteria. (Check the boxes that apply).

correct?

Algorithm	*Terminates and returns a solution? (if there is one)*	*Returns the optimal solution? (if it terminates)*	*Uses at most polynomial* space?*	*Uses at most polynomial* time?*
Depth-first search	☐	☐	☑	☐
Breadth-first search	☑	☐ 1	☐	☐
Iterative deepening	☑	☐ 1	☑	☐
A* search	☑	☑ 2	☐	☐
Gradient descent local search	☐ 3	☐ 3	☑	☐ 3

*) *polynomial in the length of the shortest path*

1) Optimal only if all edges have the same cost

2) Optimal only if the heuristic is admissible
(I gave correct if you didn't check, but wrote a comment about admissibility)

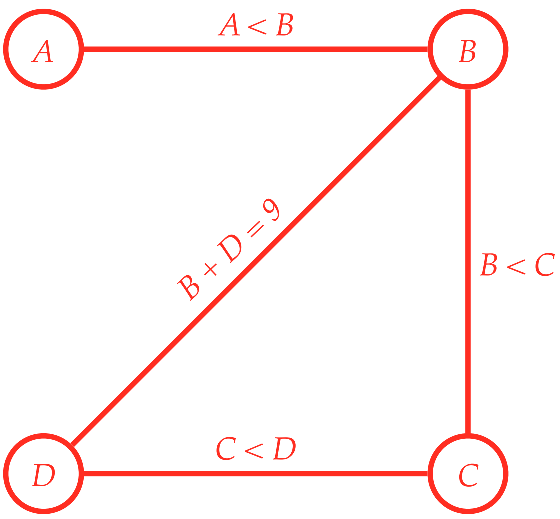
3) Depending on how you define a "solution" for local search
(I gave correct for both check and not check)

4. Constraint satisfaction

Assume that you have four variables, A, B, C and D, all with domains {1, 2, 3, 4, 5, 6}. The constraints on the values are that $A < B < C < D$ and $B + D = 9$.

a) Draw the constraint graph of this problem. Only use binary constraints.

correct?



b) What are the resulting domains after the graph is made arc consistent?

correct?

Variable	*Final domain*
A	***1, 2, 3***
B	***3, 4***
C	***4, 5***
D	***5, 6***

5. Shrdlite boxes

Assume that we have a Shrdlite world containing many boxes of different sizes and colours. $^2/_3$ of the objects are green and the rest are yellow. Furthermore, $^2/_5$ of the green boxes are small, and $^3/_4$ of the yellow ones are large.

Formulate and solve the following questions in terms of the random variables S and C (which denote the size and colour of a box, respectively).

a) What is the probability that a randomly picked box is small?

correct?

P(S=small) = [*using marginalisation*]

$= \sum_c P(S{=}small \mid C{=}c)\, P(C{=}c)$

$= P(S{=}small \mid C{=}yellow)\, P(C{=}yellow) + P(S{=}small \mid C{=}green)\, P(C{=}green)$

$= {}^1/_4 \cdot {}^1/_3 + {}^2/_5 \cdot {}^2/_3 = {}^1/_{12} + {}^4/_{15} = {}^{21}/_{60}$

$= {}^7/_{20}$

b) What is the probability that a randomly picked small box is yellow?

correct?

P(C = yellow | S=small) = [*using Bayes rule*]

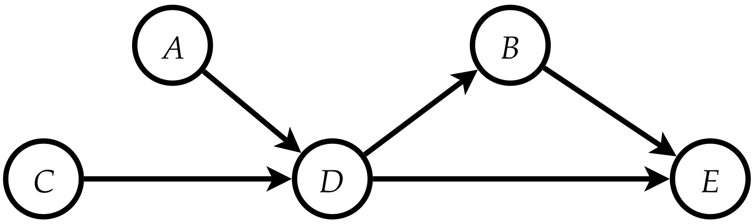
$= P(S{=}small \mid C = yellow) \cdot P(C{=}yellow) \;/\; P(S{=}small)$

$= ({}^1/_4 \cdot {}^1/_3) \;/\; ({}^7/_{20}) = {}^{20}/_{12 \cdot 7}$

$= {}^5/_{21}$

6. Bayesian networks

Consider the following Bayesian network containing five boolean random variables:



a) Write an expression for computing $P(a, \neg b, c, d, \neg e)$ given only information that is in the associated conditional probability tables (CPTs) for this network.

correct?

$P(a) \cdot P(c) \cdot P(d \mid a, c) \cdot P(\neg b \mid d) \cdot P(\neg e \mid \neg b, d)$

b) How would you calculate the probability $P(a \mid \neg b, \neg e)$ given the same network?

correct?

We calculate $P(A \mid \neg b, \neg e)$ for A = true, false, and then normalise:

$$P(A \mid \neg b, \neg e) = \alpha^{-1} \cdot \sum_C \sum_D P(A) \cdot P(C) \cdot P(D \mid A, C) \cdot P(\neg b \mid D) \cdot P(\neg e \mid \neg b, D)$$

where α is the normalisation constant,
and we sum over the hidden variables C and D.

$$P(A \mid \neg b, \neg e) = \alpha^{-1} \cdot \sum_C \sum_D P(A) \cdot P(C) \cdot P(D \mid A, C) \cdot P(\neg b \mid D) \cdot P(\neg e \mid \neg b, D) =$$

$$\alpha^{-1} \cdot P(A) \cdot \sum_C P(C) \cdot \sum_D P(D \mid A, C) \cdot P(\neg b \mid D) \cdot P(\neg e \mid \neg b, D) =$$

$$\alpha^{-1} \cdot P(A) \cdot \sum_C P(C) \cdot \mathbf{f}_D(A, C) =$$

$$\alpha^{-1} \cdot P(A) \cdot \mathbf{f}_C(A) = \text{etc...}$$

$\mathbf{f}_D(A, C)$ is a factor over A and C (i.e., a table with 2x2 values), and $\mathbf{f}_C(A)$ is a factor over A (i.e., a table with 2 values). Both factors are calculated by marginalising over the sums $\sum_D$ and $\sum_C$.

Alternatively, with another variable elimination order:

$$P(A \mid \neg b, \neg e) = \alpha^{-1} \cdot \sum_D \sum_C P(A) \cdot P(C) \cdot P(D \mid A, C) \cdot P(\neg b \mid D) \cdot P(\neg e \mid \neg b, D) =$$

$$\alpha^{-1} \cdot P(A) \cdot \sum_D P(\neg b \mid D) \cdot P(\neg e \mid \neg b, D) \cdot \sum_C P(C) \cdot P(D \mid A, C) =$$

$$\alpha^{-1} \cdot P(A) \cdot \sum_D P(\neg b \mid D) \cdot P(\neg e \mid \neg b, D) \cdot \mathbf{f}_{C'}(A, D) =$$

$$\alpha^{-1} \cdot P(A) \cdot \mathbf{f}_{D'}(A) = \text{etc...}$$