Artificial Intelligence Ph.D. Qualifier Study Guide [Rev. 6/18/2014]

The Artificial Intelligence Ph.D. Qualifier covers the content of the course Comp Sci 347 - Introduction to Artificial Intelligence. To prepare for this qualifier you are suggested to:

- Take the course Comp Sci 347
- Study from the Comp Sci 347 textbook (Stuart Russell and Peter Norvig "Artificial Intelligence: A Modern Approach", Third Edition, ISBN-13: 978-0-13-604259-4) the following: §1.1, Chapter 2, Chapter 3, §4.1, §4.5, Chapter 5
- Practice the old Comp Sci 347 exams posted on the course website (`http://web.mst.edu/~tauritzd/courses/cs347/`)
- Practice both the August 2013 and January 2014 AI Qualifier tests following this study guide and check your practice answers against the test keys following the tests.

Note that a particular qualifier can only cover a small sampling of all the above listed content. So while the practice qualifier following this study guide gives an indication of the length and difficulty you may expect, the questions on your particular qualifer might cover a completely different sampling of the above listed content.

Exam code number:

Artificial Intelligence Ph.D. Qualifier

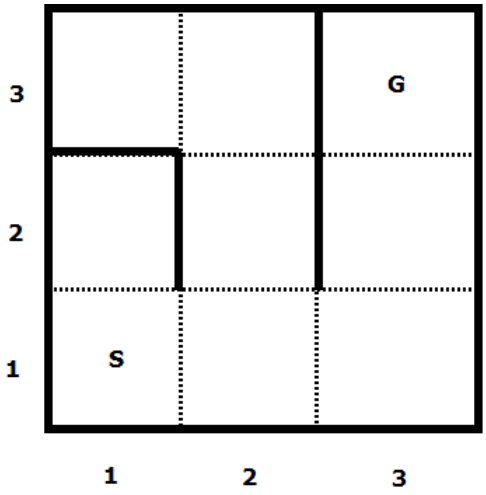
August 2013

This is a closed-book, closed-notes exam. The only items you are allowed to use are writing implements. Write your exam code number in the indicated field at the top of EACH sheet of your exam. Do NOT write your name anywhere on your exam. The max number of points per question is indicated in square brackets after each question. The sum of the max points for all the questions is 100. You have exactly one hour to complete this exam. Keep your answers clear and concise while complete. To receive full credit, you need to show all steps of how you derived your answer. Good luck!

1. Prove that A* Tree Search employing heuristic $h(n)$ is optimal if $h(n)$ is admissible. To receive full credit, you need to show all steps of your proof. [30]

2. Expectiminimax is a variant of the minimax algorithm for stochastic adversarial environments. Either prove by contradiction that no type of alpha-beta pruning is possible with expectiminimax due to the existence of chance nodes in the game trees corresponding to stochastic adversarial environments, or explain how alpha-beta pruning can be applied to expectiminimax and include a sample gametree with number example illustrating how this would work. To receive full credit, you need to show all steps of your answer. [25]

3. The remaining questions are about the 3x3 grid environment illustrated by the following diagram, where S is the start location of the agent, G is the goal location, and where the agent can move left, right, up, and down through dotted lines, but not through solid lines, and where moving to a grid location to the right has one unit cost, up has two unit cost, left has three unit cost, down has four unit cost, and heuristic $h(n)$ is defined by the Manhattan distance between the grid location of n and the grid location containing the goal.



(a) Draw the weighted state space graph capturing the diagram information relevant to executing Learning Real-Time A* (LRTA*). [5]

(b) Give the full LRTA* trace for the weighted state space graph from the previous question, employing the specified heuristic $h(n)$, terminating either when the goal is found or after the 15th call to LRTA*-COST, where nodes are expanded counter-clockwise, ending at exactly 9 o'clock and when multiple actions with equal LRTA*-COST are found, you use the one generated first. [30]

(c) What is the Competitive Ratio (CR) based on the final state of your LRTA* trace? Explain your answer and make sure to list the final state you are using. Note that in the case of call limit termination, if the LRTA*-COST call you terminated on returned an action, then for the purpose of computing the CR, the action is assumed to have been executed. [5]

(d) Explain concisely the type of environment in which one would employ LRTA* and why. [5]

Exam code number:

Artificial Intelligence Ph.D. Qualifier Key

August 2013

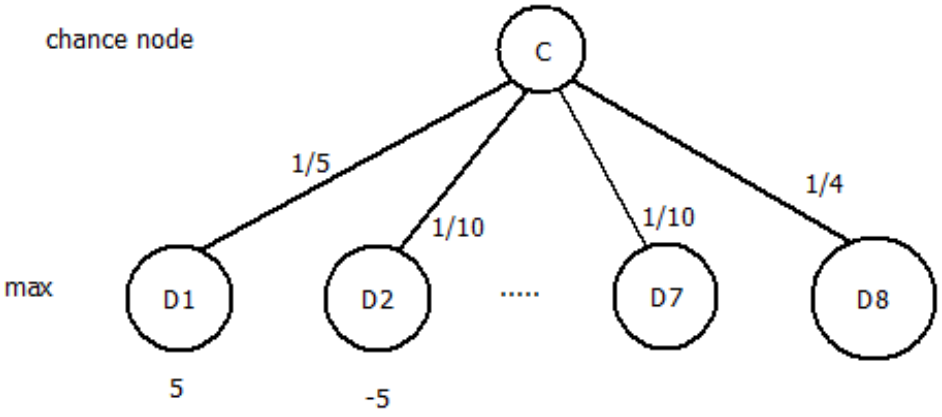
This is a closed-book, closed-notes exam. The only items you are allowed to use are writing implements. Write your exam code number in the indicated field at the top of EACH sheet of your exam. Do NOT write your name anywhere on your exam. The max number of points per question is indicated in square brackets after each question. The sum of the max points for all the questions is 100. You have exactly one hour to complete this exam. Keep your answers clear and concise while complete. To receive full credit, you need to show all steps of how you derived your answer. Good luck!

1. Prove that A* Tree Search employing heuristic $h(n)$ is optimal if $h(n)$ is admissible. To receive full credit, you need to show all steps of your proof. [30]

   Suppose suboptimal goal node G appears on the frontier and let the cost of the optimal solution be C^*. From the definition of $h(node)$ we know that $h(G) = 0$ and because G is suboptimal we know $f(G) > C^*$. Together this gives $f(G) = g(G) + h(G) = g(G) > C^*$.
   If there is an optimal solution, then there is a frontier node N that is on an optimal solution path. Our proof is for an admissible heuristic, so we know $h(N)$ does not overestimate, therefore $f(N) = g(N) + h(N) \leq C^*$. Together this gives $f(N) \leq C^* < f(G)$. As A* Tree Search expands lower f-cost nodes before higher f-cost nodes, N will always be expanded before G, ergo A* is optimal!

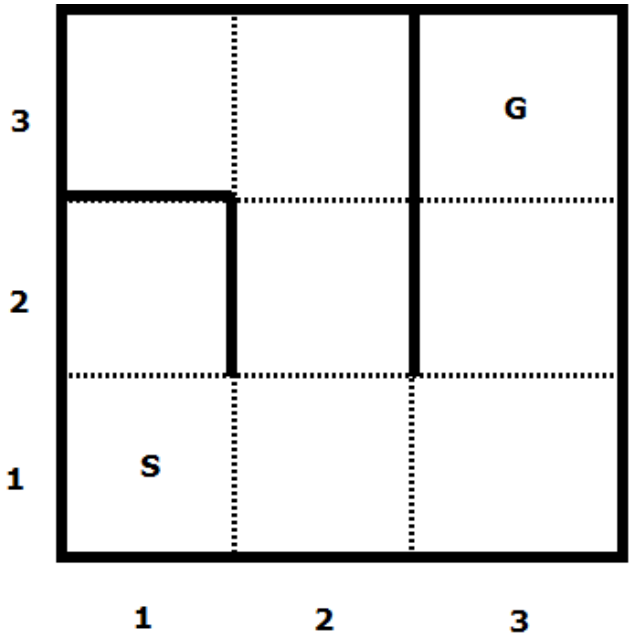
2. Expectiminimax is a variant of the minimax algorithm for stochastic adversarial environments. Either prove by contradiction that no type of alpha-beta pruning is possible with expectiminimax due to the existence of chance nodes in the game trees corresponding to stochastic adversarial environments, or explain how alpha-beta pruning can be applied to expectiminimax and include a sample gametree with number example illustrating how this would work. To receive full credit, you need to show all steps of your answer. [25]

   Alpha-beta pruning can be accomplished with expectiminimax by iteratively shrinking the interval of possible values until alpha or beta fall outside that interval. For example, assume the following sample gametree with a bound of [-10,15] on the state eval values:

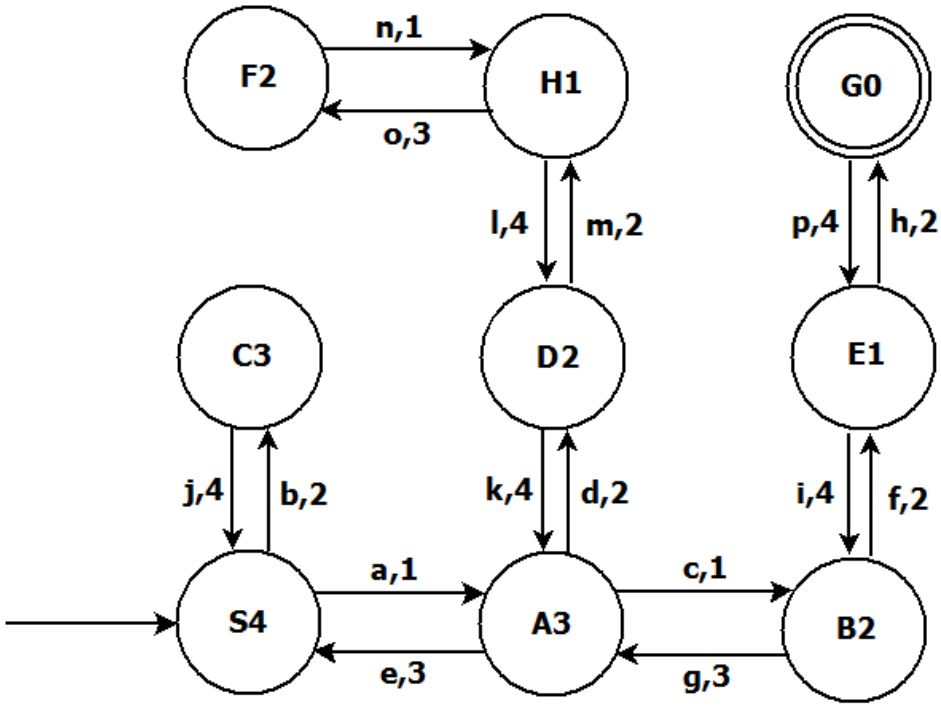


   Before evaluating any max node, the bound on C is [-10,15]. After evaluating $D1$, the bound can be computed as follows:
   $[\frac{1}{5} \cdot 5 + \frac{4}{5} \cdot -10, \frac{1}{5} \cdot 5 + \frac{4}{5} \cdot 15] = [-7, 13]$
   After evaluating $D2$, the bound is tightened to:
   $[\frac{1}{5} \cdot 5 + \frac{1}{10} \cdot -5 + \frac{7}{10} \cdot -10, \frac{1}{5} \cdot 5 + \frac{1}{10} \cdot -5 + \frac{7}{10} \cdot 15] = [-6.5, 11]$
   If the interval shrinks to the point where the upper bound of the interval is smaller than alpha, then this is a fail-low and would cause an alpha-beta prune for the min player. If the interval shrinks to the point where the lower bound of the interval is larger than beta, then this is a fail-high and would cause an alpha-beta prune for the max player.

3. The remaining questions are about the 3x3 grid environment illustrated by the following diagram, where S is the start location of the agent, G is the goal location, and where the agent can move left, right, up, and down through dotted lines, but not through solid lines, and where moving to a grid location to the right has one unit cost, up has two unit cost, left has three unit cost, down has four unit cost, and heuristic $h(n)$ is defined by the Manhattan distance between the grid location of n and the grid location containing the goal.



(a) Draw the weighted state space graph capturing the diagram information relevant to executing Learning Real-Time A* (LRTA*). [5]



(b) Give the full LRTA* trace for the weighted state space graph from the previous question, employing the specified heuristic $h(n)$, terminating either when the goal is found or after the 15th call to LRTA*-COST, where nodes are expanded counter-clockwise, ending at exactly 9 o'clock and when multiple actions with equal LRTA*-COST are found, you use the one generated first. [30]

current state	last action	previous state	cost estimate	world knowledge	LRTA*-COST	min action, cost
S	-	-	H[S]=4	-	(S,a,-)=4	a,4
				-	(S,b,-)=4	[a],4
A	a	S	H[A]=3	R[S,a]=A	(S,a,A)=1+3=4	a,4
					(S,b,-)=4	a,[4]
			H[S]=4		(A,c,-)=3	c,3
					(A,d,-)=3	c,3
					(A,e,-)=3	[c],3
B	c	A	H[B]=2	R[A,c]=B	(A,c,B)=1+2=3	c,3
					(A,d,-)=3	c,3
					(A,e,-)=3	c,[3]
			H[A]=3		(B,f,-)=2	f,2
					(B,g,-)=2	[f],2
E	f	B	H[E]=1	R[B,f]=E	(B,f,E)=2+1=3	f,3
					(B,g,-)=2	g,[2]
			H[B]=2		(E,i,-)=1	i,1

LRTA*-COST call limit reached

(c) What is the Competitive Ratio (CR) based on the final state of your LRTA* trace? Explain your answer and make sure to list the final state you are using. Note that in the case of call limit termination, if the LRTA*-COST call you terminated on returned an action, then for the purpose of computing the CR, the action is assumed to have been executed. [5]

$CR = \frac{c(S,a,A)+c(A,c,B)+c(B,f,E)}{c^*(S,E)} = \frac{4}{4} = 1$

(d) Explain concisely the type of environment in which one would employ LRTA* and why. [5]

Online search algorithms like LRTA* are *necessary* when operating in unknown environments where the agent does not know what states exist or what its actions do. They are *useful* in dynamic or semi-dynamic environments where there is a penalty for sitting around and computing too long. They are also *useful* in nondeterministic environments because they allow agents to focus its computational efforts on the contingencies that actually arise rather than those that might happen but probably will not. In environments where a sufficiently accurate heuristic estimate of the remaining path-cost to the nearest goal state is available, LRTA* outperforms uninformed online search algorithms.

Exam code number:

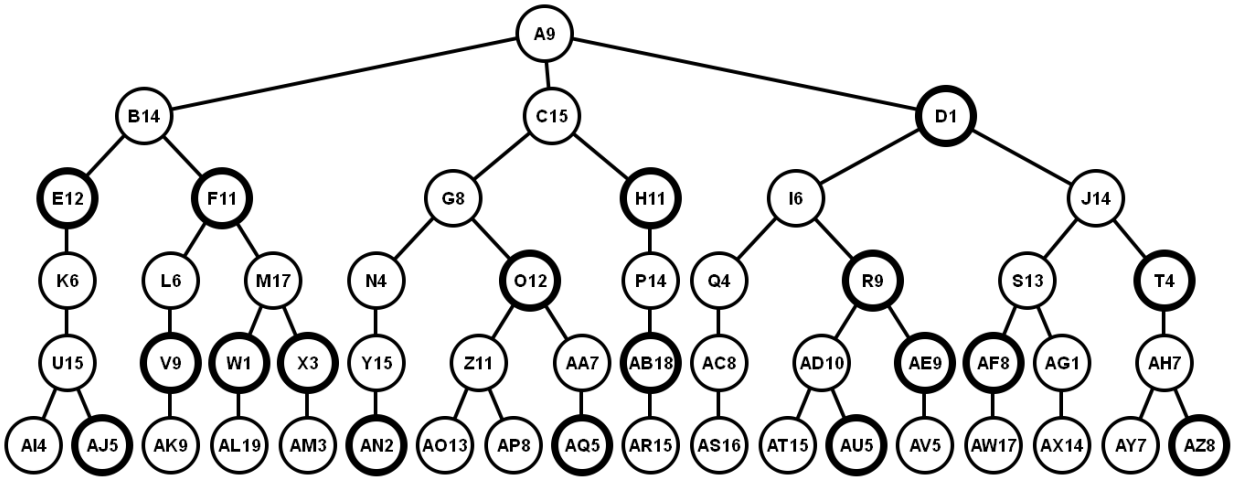
Artificial Intelligence Ph.D. Qualifier

January 2014

This is a closed-book, closed-notes exam. The only items you are allowed to use are writing implements. Write your exam code number in the indicated field at the top of EACH sheet of your exam. Do NOT write your name anywhere on your exam. The max number of points per question is indicated in square brackets after each question. The sum of the max points for all the questions is 100. You have exactly 90 minutes to complete this exam. Keep your answers clear and concise while complete. To receive full credit, you need to show all steps of how you derived your answer. Good luck!

1. Prove that A* Graph Search employing heuristic $h(n)$ is optimal if $h(n)$ is consistent. To receive full credit, you need to show all steps of your proof. [25]

2. The next three questions are about the following adversarial search tree. State evaluation heuristic values for the max player are provided in the form of numbers following the letter labels of the states (e.g., A9 indicates that the heuristic value of state A for the max player is 9). The order in which successors are generated is from left to right. Example: A generates first B, then C, and finally D. Non-quiescent states are indicated by bold circled states.

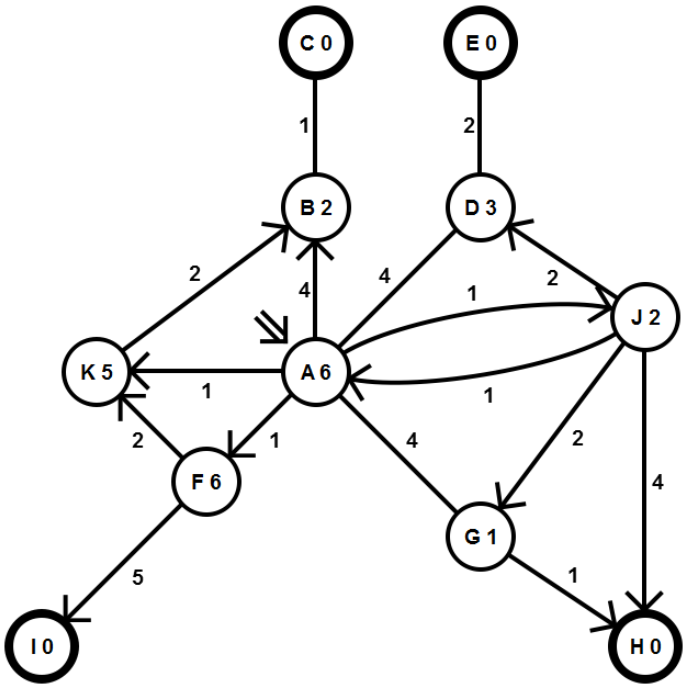


(a) Give the execution trace for HTQSABIDM (A,3,2,$-\infty, \infty$). That is, give the execution trace for Iterative-Deepening Minimax with History-Table, Quiescense-Search, and Alpha-Beta Pruning, starting in node A, with a regular search depth of 3, a quiescence search depth of 2, and a $(-\infty, \infty)$ alpha-beta window. [25]

(b) Indicate for each depth iteration of HTQSABIDM(A,3,2,$-\infty, \infty$) which nodes, if any, get pruned. [7]

(c) What is the Principal Variant (PV) found by HTQSABIDM(A,3,2,$-\infty, \infty$)? [3]

3. The last questions are about the following state space graph. Let A be the initial state and C, E, I, and H the goal states. The edge labels indicate step-cost, the vertex labels contain the node identifier in the form of a letter. Heuristic $h1(s)$ is defined as the minimum number of steps from state s to the closest goal state; for example, $h1(A) = 2$. Heuristic $h2(s)$ is defined by the values following the node labels in the state space graph; for example, $h2(A) = 6$. The order in which successors are generated is counterclockwise, ending at exactly 9 o'clock. Example: A generates first F, then G, then J, then D, then B, and finally K. When sorting by f-value, nodes with equal f-value are ordered such that the earlier a node is generated, the higher its priority. Nodes already on the frontier have higher priority than newly added nodes with equal f-value. Uniform Cost Tree Search (UCTS) finds a solution with a path-cost of 4. You may use the following abbreviations without defining them: DLR = Depth Limit Reached, NGF = No Goal Found, GF = Goal Found.



(a) Give the execution trace for Iterative Deepening Depth First Tree Search (ID-DFTS). [10]

(b) Is ID-DFTS optimal for this problem? Explain your answer! [1]

(c) Give the execution trace for A^* Graph Search (A^*GS) employing heuristic $h1$. [10]

(d) Give the execution trace for A^* Graph Search (A^*GS) employing heuristic $h2$. [10]

(e) Is for this problem $h2$ admissible? Explain your answer! [3]

(f) Is for this problem $h2$ consistent? Explain your answer! [1]

(g) Is A^*GS employing heuristic $h2$ optimal for this problem? Explain your answer! [2]

(h) Given that $h1$ is both admissible and consistent for this problem, is the max composite heuristic $h_c(s)$ defined as $max\{h1(s), h2(s)\}$ consistent for this problem? Explain your answer! [3]

Exam code number:

Artificial Intelligence Ph.D. Qualifier Key

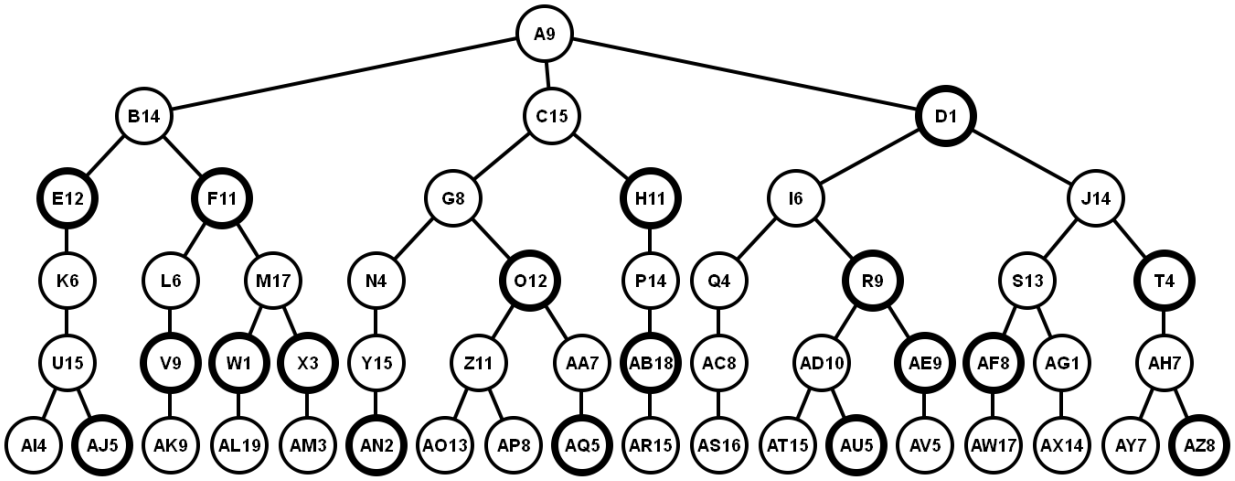
January 2014

This is a closed-book, closed-notes exam. The only items you are allowed to use are writing implements. Write your exam code number in the indicated field at the top of EACH sheet of your exam. Do NOT write your name anywhere on your exam. The max number of points per question is indicated in square brackets after each question. The sum of the max points for all the questions is 100. You have exactly 90 minutes to complete this exam. Keep your answers clear and concise while complete. To receive full credit, you need to show all steps of how you derived your answer. Good luck!

1. Prove that A* Graph Search employing heuristic $h(n)$ is optimal if $h(n)$ is consistent. To receive full credit, you need to show all steps of your proof. [25]

Suppose $h(n)$ is consistent and $c(n, a, n')$ is the cost to go with action a from node n to succesor node n'. Then $g(n') = g(n) + c(n, a, n')$. Also, per the definition of consistency, $h(n) \leq c(n, a, n') + h(n')$. Together this gives $f(n') = g(n') + h(n') = g(n) + c(n, a, n') + h(n') \geq g(n) + h(n) = f(n)$. So $f(n') \geq f(n)$ and thus the values of $f(n)$ along any path are monotonically non-decreasing. Whenever A* Graph Search selects a node n for expansion, the optimal path to that node has been found, because, were this not the case, then there would have to be another frontier node n' on the optimal path from the start node to n, but because f is non-decreasing along any path, n' would have lower f-cost than n and would have been selected first. From the two preceding observations, it follows that the sequence of nodes expanded by A* Graph Search is in non-decreasing order of $f(n)$. Hence, the first goal node selected for expansion must be an optimal solution, because f is the true cost for goal nodes and all later goal nodes will be at least as expensive. $\square$

2. The next three questions are about the following adversarial search tree. State evaluation heuristic values for the max player are provided in the form of numbers following the letter labels of the states (e.g., A9 indicates that the heuristic value of state A for the max player is 9). The order in which successors are generated is from left to right. Example: A generates first B, then C, and finally D. Non-quiescent states are indicated by bold circled states.



(a) Give the execution trace for HTQSABIDM (A,3,2,$-\infty,\infty$). That is, give the execution trace for Iterative-Deepening Minimax with History-Table, Quiescense-Search, and Alpha-Beta Pruning, starting in node A, with a regular search depth of 3, a quiescence search depth of 2, and a $(-\infty,\infty)$ alpha-beta window. [25]

#define DLM( ) HTQSABDLM( ), #define Max( ) HTQSABMaxV( ), #define Min( ) HTQSABMinV( )

call	frontier	eval	value	α,β	best action,value
DLM(A,1,2,$-\infty$,∞)	B0C0D0	B	MinV(B,0,2,$-\infty$,∞)=14	14, ∞	AB, 14
	C0D0	C	MinV(C,0,2,14,∞)=15	15, ∞	AC, 15
	D0	D	MinV(D,0,2,15,∞)=6 (QS)	15, ∞	AC, 15 [AC:1]
MinV(D,0,2,15,∞)	I0J0	I	MaxV(I,0,1,15,∞)=6 (Prune)	15, ∞	DI, 6 [DI:1]
DLM(A,2,2,$-\infty$,∞)	C1B0D0	C	MinV(C,1,2,$-\infty$,∞)=8	8, ∞	AC, 8
	B0D0	B	MinV(B,1,2,8,∞)=6	8, ∞	AC, 8
	D0	D	MinV(D,1,2,8,∞)=6	8, ∞	AC, 8 [AC:2]
MinV(C,1,2,$-\infty$,∞)	G0H0	G	MaxV(G,0,2,$-\infty$,∞)=8	$-\infty$, 8	CG, 8
	H0	H	MaxV(H,0,2,$-\infty$,8)=14 (QS,SSS,Prune)	$-\infty$, 8	CG, 8 [HP:1,CG:1]
MinV(B,1,2,8,∞)	E0F0	E	MaxV(E,0,2,8,∞)=6 (QS,SSS,Prune)	8, ∞	BE, 6 [EK:1,BE:1]
MinV(D,1,2,8,∞)	I1J0	I	MaxV(I,0,2,8,∞)=6 (Prune)	8, ∞	DI, 6 [DI:2]
DLM(A,3,2,$-\infty$,∞)	C2B0D0	C	MinV(C,2,2,$-\infty$,∞)=7	7, ∞	AC, 7
	B0D0	B	MinV(B,2,2,7,∞)=6	7, ∞	AC, 7
	D0	D	MinV(D,2,2,7,∞)=5	7, ∞	AC, 7 [AC:3]
MinV(C,2,2,$-\infty$,∞)	G1H0	G	MaxV(G,1,2,$-\infty$,∞)=7	$-\infty$, 7	CG, 7
	H0	H	MaxV(H,1,2,$-\infty$,7)=14 (SSS,Prune)	$-\infty$, 7	CG, 7 [HP:2,CG:2]
MaxV(G,1,2,$-\infty$,∞)	N0O0	N	MinV(N,0,2,$-\infty$,∞)=4	4, ∞	GN, 4
	O0	O	MinV(O,0,2,4,∞)=7 (QS)	7, ∞	GO, 7 [GO:1]
MinV(O,0,2,4,∞)	Z0AA0	Z	MaxV(Z,0,1,4,∞)=11	4, 11	OZ, 11
	AA0	AA	MaxV(AA,0,1,4,11)=7	4, 7	OAA, 7 [O-AA:1]
MinV(B,2,2,7,∞)	E1F0	E	MaxV(E,1,2,7,∞)=6 (SSS,Prune)	7, ∞	BE, 6 [EK:2,BE:2]
MinV(D,2,2,7,∞)	I2J0	I	MaxV(I,1,2,7,∞)=5 (Prune)	7, ∞	DI, 5 [DI:3]
MaxV(I,1,2,7,∞)	Q0R0	Q	MinV(Q,0,2,7,∞)=4	7, ∞	IQ, 4
	R0	R	MinV(R,0,2,7,∞)=5 (QS)	7, ∞	IR, 5 [IR:1]
MinV(R,0,2,7,∞)	AD0AE0	AD	MaxV(AD,0,1,7,∞)=10	7, 10	RAD, 10
	AE0	AE	MaxV(AE,0,1,7,10)=5 (QS,SSS,Prune)	7, 10	RAE, 5 [AE-AV:1,R-AE:1]

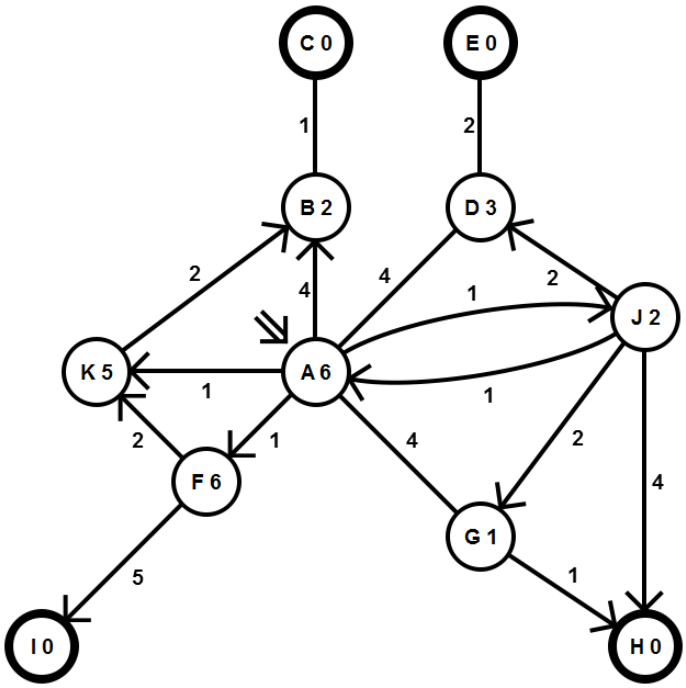
(b) Indicate for each depth iteration of HTQSABIDM(A,3,2,$-\infty, \infty$) which nodes, if any, get pruned. [7]

Depth 1: J
Depth 2: F,L,M,J
Depth 3: F,L,M,J,S,T,AH

(c) What is the Principal Variant (PV) found by HTQSABIDM(A,3,2,$-\infty, \infty$)? [3]

$A \rightarrow C, C \rightarrow G, G \rightarrow O, O \rightarrow AA$

3. The last questions are about the following state space graph. Let A be the initial state and C, E, I, and H the goal states. The edge labels indicate step-cost, the vertex labels contain the node identifier in the form of a letter. Heuristic $h1(s)$ is defined as the minimum number of steps from state s to the closest goal state; for example, $h1(A) = 2$. Heuristic $h2(s)$ is defined by the values following the node labels in the state space graph; for example, $h2(A) = 6$. The order in which successors are generated is counterclockwise, ending at exactly 9 o'clock. Example: A generates first F, then G, then J, then D, then B, and finally K. When sorting by f-value, nodes with equal f-value are ordered such that the earlier a node is generated, the higher its priority. Nodes already on the frontier have higher priority than newly added nodes with equal f-value. Uniform Cost Tree Search (UCTS) finds a solution with a path-cost of 4. You may use the following abbreviations without defining them: DLR = Depth Limit Reached, NGF = No Goal Found, GF = Goal Found.



(a) Give the execution trace for Iterative Deepening Depth First Tree Search (ID-DFTS). [10]

depth-limit=0

frontier	eval
A	A

depth-limit reached and no goal found

depth-limit=1

frontier	eval
A	A
F G J D B K	F
G J D B K	G
J D B K	J
D B K	D
B K	B
K	K

depth-limit reached and no goal found

depth-limit=2

frontier	eval
A	A
F G J D B K	F
I K G J D B K	I

goal found; solution = AFI; path-cost(AFI) = 6

(b) Is ID-DFTS optimal for this problem? Explain your answer! [1]

No, because UCTS was stated to have found a lower path-cost solution.

(c) Give the execution trace for A^* Graph Search (A^*GS) employing heuristic $h1$. [10]

frontier	explored	eval
A2	-	A2
F2 J2 K3 G5 D5 B5	A	F2
J2 K3 G5 D5 B5 I6	A F	J2
K3 G4 D4 B5 H5 I6	A F J	K3
G4 D4 B4 H5 I6	A F J K	G4
D4 B4 H4 I6	A F J K G	D4
B4 H4 E5 I6	A F J K G D	B4
H4 C4 E5 I6	A F J K G D B	H4

goal found; solution = AJGH; path-cost(AJGH) = 4

(d) Give the execution trace for A^* Graph Search (A^*GS) employing heuristic $h2$. [10]

frontier	explored	eval
A6	-	A6
J3 G5 B6 K6 F7 D7	A	J3
G4 H5 B6 K6 D6 F7	A J	G4
H4 B6 K6 D6 F7	A J G	H4

goal found; solution = AJGH; path-cost(AJGH) = 4

(e) Is for this problem $h2$ admissible? Explain your answer! [3]

No, because for instance $h2(A) = 6 > 4 =$*path-cost*$(AJGH)$.

(f) Is for this problem $h2$ consistent? Explain your answer! [1]

No, because it is not admissible as shown previously.

(g) Is A^*GS employing heuristic $h2$ optimal for this problem? Explain your answer! [2]

Yes, because it found a solution with the same path-cost as UCTS was stated to have found and UCTS is known to be optimal when the branching factor is finite and the step costs are all positive.

(h) Given that $h1$ is both admissible and consistent for this problem, is the max composite heuristic $h_c(s)$ defined as $max\{h1(s), h2(s)\}$ consistent for this problem? Explain your answer! [3]

No, because for instance $h_c(A) = 6 > 3 = c(A, J) + h_c(J)$.