CS347 SP2005 Exam 3 Key

This is a closed-book closed-notes exam. The *only* items you are permitted to bring are writing implements. Mark every sheet of paper you use with your name and the string "cs347sp2005 exam3" (omittance, even if it is partial, will be penalized at 1 point per sheet). If you are caught cheating, you will receive a zero grade for this exam. The max number of points per question is indicated in square brackets after each question. The sum of the max points is 75. You have 75 minutes to complete this exam. Good luck!

1. Give two advantages of Iterative Deepening Minimax algorithms over Depth Limited Minimax algorithms. [6]

   I) Solution availability: i.e., you always have the solution of the previous iteration available during the execution of the current iteration (this is particularly useful when under a time constraint).

   II) Information gleaned during the current iteration can be employed to increase pruning in successive iterations (e.g., history table). Because successive iterations require exponentially more CPU time, the overhead of searching at lower depths is typically insignificant while increased pruning at higher depths can be very significant.

2. In the context of Iterative Deepening Minimax for a given search depth, on the provided copy of below table, put check marks in cells to indicate agreement, otherwise leave cells blank. [5]

	Typically speeds up search	Does not influence search result
Alpha-Beta Pruning		
Quiescence Search		
Forward Pruning		
History Table Move Ordering Heuristic		
Transposition Table		

	Typically speeds up search	Does not influence search result
Alpha-Beta Pruning	x	x
Quiescence Search	1	
Forward Pruning	x	
History Table Move Ordering Heuristic	x	x
Transposition Table	x	x

1) This can be argued both ways. In the context of iterative deepening the search may initially take longer because of searching at increased depth; however, increased pruning at higher iterations may sufficiently compensate to reduce the overall search time.

3. (a) Explain what a transposition table is and what its use is in adversarial search. [5]

   A transposition table is a datastructure which stores the evaluation values of previously seen states in an adversarial search tree to prevent recomputations. Typically the datastructure employed is a hash table. The effect can be dramatic, sometimes as much as doubling the reachable search depth, although at the price of potentially huge amounts of storage space.

   (b) What is different between a transposition table used for ABIDM and one used for IDMTDf? [4]

   While it would be best to store α & β values for both, ABIDM will typically work okay without while IDMTDf absolutely requires them to work effectively. Alternatively IDMTDf can also store lower and upper bounds.

4. Explain briefly the idea behind the null-move pruning heuristic. [5]

   If in a reduced depth search allowing your opponent two consecutive moves still results in a fail high or fail low, then it is probably safe to cut off without searching at full depth. Null-move forward pruning allows a chess program to experience a dramatic reduction in branching factor with some manageable risk of missing something important.

5. The null-move pruning heuristic is an example of forward pruning, while $\alpha - \beta$ pruning is an example of backward pruning.

   (a) What is the advantage of forward pruning over backward pruning? [4]
   The advantage is that it accomplishes much more aggresive pruning based on a heuristic function rather than a partial search.

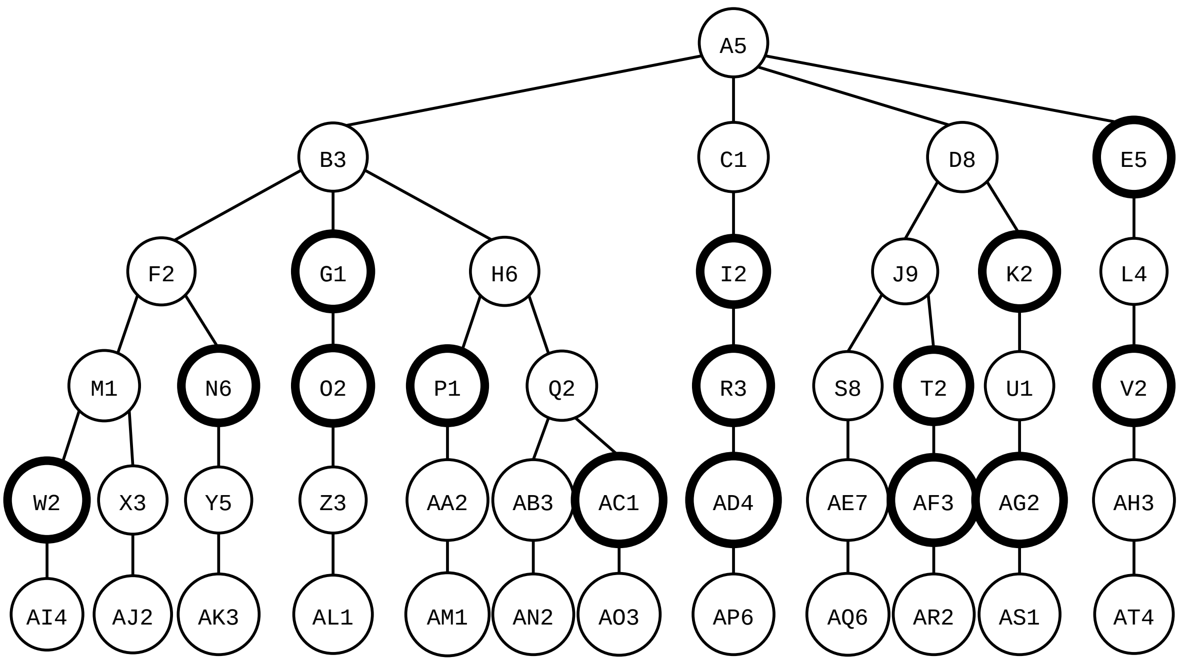
   (b) What is the advantage of backward pruning over forward pruning? [4]
   The advantage is that it maintains optimality while reducing time complexity.

   (c) Is it beneficial to combine forward and backward pruning in a single search? Explain your answer! [3]
   Yes, because they do not negatively infuence each other so combining them gives you the advantages of both.

6. Explain briefly how the idea of $\alpha - \beta$ pruning can be applied to game trees with chance nodes. [4]

   This can be accomplished by putting boundaries on the utility function which effectively imposes a boundary on the average, which in turn allows us to place an upper bound on the value of a chance node without looking at all its children.

7. The last 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., A5 indicates that the heuristic value of state A for the max player is 5). The order in which successors are generated is from left to right. Example: A generates first B, then C, then D, and finally E. Non-quiescent states are indicated by bold circled states.



A pre-initialized history table is provided as follows: all entries are zero except for those specified in the following table:

Move	AC	AE	BG	BH
HT value	1	2	2	2

(a) Give the execution trace for HTQSABIDM(A,3,2,$-\infty, \infty$). [30]

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

call	open	eval	value	α, β	best action,value
DLM(A,1,2,$-\infty, \infty$)	E2C1B0D0	E	Min(E,0,2,$-\infty, \infty$)=4 (SSS,QS)	$4, \infty$	AE,4 [EL:1]
	C1B0D0	C	Min(C,0,2,4,∞)=1	$4, \infty$	AE,4
	B0D0	B	Min(B,0,2,4,∞)=3	$4, \infty$	AE,4
	D0	D	Min(D,0,2,4,∞)=8	$8, \infty$	AD,8 [AD:1]
DLM(A,2,2,$-\infty, \infty$)	E2C1D1B0	E	Min(E,1,2,$-\infty, \infty$)=4 (SSS)	$4, \infty$	AE,4 [EL:2]
	C1D1B0	C	Min(C,1,2,4,∞)=4 (SSS,QS,prune)	$4, \infty$	AE,4 [CI:1,IR:1,R-AD:1]
	D1B0	D	Min(D,1,2,4,∞)=1	$4, \infty$	AE,4
	B0	B	Min(B,1,2,4,∞)=3	$4, \infty$	AE,4 [AE:3]
Min(D,1,2,4,∞)	J0K0	J	Max(J,0,2,4,∞)=9	$4, 9$	DJ,9
	K0	K	Max(K,0,2,4,9)=1 (SSS,QS,prune)	$4, 9$	DK,1 [DK:1,KU:1]
Min(B,1,24,∞)	G2H2F0	G	Max(G,0,2,4,∞)=3 (SSS,QS,prune)	$4, \infty$	BG,3 [BG:3,GO:1,OZ:1]
DLM(A,3,2,$-\infty, \infty$)	E3C1D1B0	E	Min(E,2,2,$-\infty, \infty$)=3 (SSS,QS)	$3, \infty$	AE,3 [EL:3,LV:1,V-AH:1]
	C1D1B0	C	Min(C,2,2,3,∞)=6 (SSS,QS)	$6, \infty$	AC,6 [CI:2,IR:2, R-AD:2,AD-AP:1]
	D1B0	D	Min(D,2,2,6,∞)=1	$6, \infty$	AC,6
	B0	B	Min(B,2,2,6,∞)=3	$6, \infty$	AC,6 [AC:2]
Min(D,2,2,6,∞)	K1J0	K	Max(K,1,2,6,∞)=1 (SSS,prune)	$6, \infty$	DK,1 [DK:2,KU:2]
Min(B,2,2,6,∞)	G3H2F0	G	Max(G,1,2,6,∞)=3 (SSS,QS,prune)	$6, \infty$	BG,3 [BG:4,GO:2,OZ:2]

(b) Which nodes, if any, get pruned by HTQSABIDM(A,3,2,$-\infty, \infty$)? [3]

Depth 1: none
Depth 2: F,H
Depth 3: J,S,T,AF,AR,F,M,N,Y,H,P,AA,Q

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

A,C,I,R,AD,AP