For the Mid-term Exam, “Full credit” gives the percentage who received 100%, “Partial credit” gives the percentage who received partial credit, and “Zero credit” gives the percentage of students who received zero.

Problem 1:
full credit: ~15% (~12 students)
partial credit: ~84% (~66 students)
zero credit: ~1% (~1 student)
Common mistake: Not doing AC-3 until quiescence

Problem 2:
full credit: ~47% (~37 students)
partial credit: ~41% (~32 students)
zero credit: ~13% (~10 students)
Common mistake: Canceling 2 literals, e.g.: **NO!!** Resolve (A B C) with ((¬ B) (¬ C) D) to produce (A D) **NO!!**

Problem 3:
full credit: ~87% (~69 students)
partial credit: ~10% (~8 students)
zero credit: ~3% (~2 students)

Problem 4:
full credit: ~53% (~42 students)
partial credit: ~47% (~37 students)
zero credit: ~0% (~0 students)
Common mistakes: Many students did not seem to know the definitions of Valid, Unsatisfiable, Satisfiable.

Problem 5:
full credit: ~58% (~46 students)
partial credit: ~42% (~33 students)
zero credit: ~0% (~0 students)
Common mistake: Many students seemed to “guess” Alpha-Beta pruning, instead of working the algorithm.

Problem 6:
full credit: ~29% (~23 students)
partial credit: ~70% (~55 students)
zero credit: ~1% (~1 student)
Common mistakes: Getting the order of successor nodes wrong; getting the goal-test wrong; forgetting to do IDS at depth=0

Problem 7:
full credit: ~49% (~39 students)
partial credit: ~47% (~37 students)
zero credit: ~4% (~3 students)
Common mistake: Forgetting to account for the current Temperature when computing the exponent.

Problem 8:
full credit: ~63% (~50 students)
partial credit: ~37% ~(29 students)
zero credit: ~0% (~0 students)

Problem 9:
full credit: ~61% (~48 students)
partial credit: ~39% (~31 students)
zero credit: ~0% (~0 students)

CS-171, Intro to A.I. — Mid-term Exam — Winter Quarter, 2015

YOUR NAME: _____

YOUR ID: _____ ID TO RIGHT: _____ ROW: _____ SEAT: _____

The exam will begin on the next page. Please, do not turn the page until told.

When you are told to begin the exam, please check first to make sure that you have all ten pages, as numbered 1-10 in the bottom-right corner of each page. We wish to avoid copy problems. We will supply a new exam for any copy problems.

The exam is closed-notes, closed-book. No calculators, cell phones, electronics.

Please turn off all cell phones now.

Please clear your desk entirely, except for pen, pencil, eraser, a blank piece of paper (for scratch pad use), and an optional water bottle. Please write your name and ID# on the blank piece of paper and turn it in with your exam.

This page summarizes the points for each question, so you can plan your time.

1. (14 pts total, 2 pts each) SUDOKU AS A CONSTRAINT SATISFACTION PROBLEM.

2. (8 pts total) ONE FISH, TWO FISH, RED FISH, BLUE FISH. (With apologies to Dr. Seuss.)

3. (5 points total, -1 pt each error, but not negative) CONSTRAINT GRAPH CONSTRUCTION.

4. (12 pts total, 2 pts each) VALID, UNSATISFIABLE, SATISFIABLE.

5. (18 pts total, 1 pt each) GAME SEARCH WITH TIC-TAC-TOE AND WIN-PATHS HEURISTIC FUNCTION.

6. (15 pts total, 3 pts each) STATE SPACE SEARCH.

7. (8 pts total, 1 pt each) LOCAL SEARCH --- SIMULATED ANNEALING.

8. (10 pts total, 1 pt each) ADVERSARIAL (GAME) SEARCH CONCEPTS.

9. (10 pts total, 1 pt each) CONSTRAINT SATISFACTION PROBLEM (CSP) CONCEPTS.

The Exam is printed on both sides to save trees! Work both sides of each page!

1. (14 pts total, 2 pts each) SUDOKU AS A CONSTRAINT SATISFACTION PROBLEM.
A Sudoku board consists of $n \times n$ squares, some of which are initially filled with digits from 1 to n. The objective of the Sudoku puzzle is to fill in all the remaining squares such that no digit appears twice in any row, column, or $\sqrt{n} \times \sqrt{n}$ box. **Consider the case in which $n = 4$.**

The Sudoku puzzle with $n = 4$ can be formulated as a constraint satisfaction problem with $n^2 = 16$ variables, where every variable stands for one of the 16 squares on the 4x4 Sudoku board. If we denote rows with letters A-D and columns with numbers 1-4, we end up with 16 variables: A1, A2, A3, A4, B1, B2, …, D3, D4. Each variable has domain values {1, 2, 3, 4

See R&N Chapter 6. See lecture slides Constraint Satisfaction Problems/ Propagation (Tue.-Thu., 27-29 Jan.).

What remains to be done is to specify t…se binary constraints for each pair of variables, but might end up wit… alternative is to use global $AllDiff$ constraints, where $AllDiff(X_1$…$_1, X_2, \ldots, X_n$, must have different values.

1.a. (2 pts) How many $AllDiff$ constraints are required to specify that no digit may appear twice in any row, column, or 2×2 box? 12 (= 1 per row, 1 per column, and 1 per box, = 3 constraints across 4 cases)

	1	2	3	4
A	1 2 3 4	1 2 3 4	1 2 ~~4~~	1 2 ~~4~~
B	1 2 ~~4~~	1 2 ~~4~~	3	1 2 ~~4~~
C	1 2 3 4	1 2 3 4	1 2 ~~4~~	1 2 3 4
D	1 2 3 4	1 2 3 4	1 2 ~~4~~	1 2 3 4

See Section 6.3.2.

1.b. (2 pts) FORWARD CHECKING.
Consider the 4×4 Sudoku board on the left. Possible domain values for each variable are shown in each square. Variable B3 has just been assigned value 3 (circled).

Cross out all values from the domains of the remaining unassigned variables that now would be eliminated by Forward Checking.

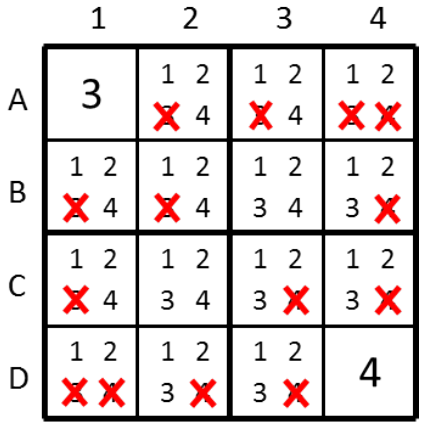
	1	2	3	4
A	1 2 3 ~~4~~	1 ~~2~~ 3 ~~4~~	~~1~~ ~~2~~ ~~3~~ 4	1 2 ~~3~~ ~~4~~
B	1 2 ~~3~~ ~~4~~	1 ~~2~~ ~~3~~ 4	3	1 2 ~~3~~ ~~4~~
C	~~1~~ ~~2~~ ~~3~~ 4	2	1 ~~2~~ ~~3~~ ~~4~~	~~1~~ ~~2~~ ~~3~~ ~~4~~
D	1 ~~2~~ 3 ~~4~~	1 ~~2~~ 3 ~~4~~	~~1~~ ~~2~~ ~~3~~ ~~4~~	4

See Section 6.3.2.

1.c. (2 pts) ARC CONSISTENCY.
Consider the 4×4 Sudoku board on the left. Possible domain values for each variable are shown in each square. Variables B3, C2, and D4 have been assigned values, but no constraint propagation has been done.

Cross out all values from the domains of the remaining unassigned variables that now would be eliminated by Arc Consistency (AC-3 in your book).

****** TURN PAGE OVER AND CONTINUE ON THE OTHER SIDE ******

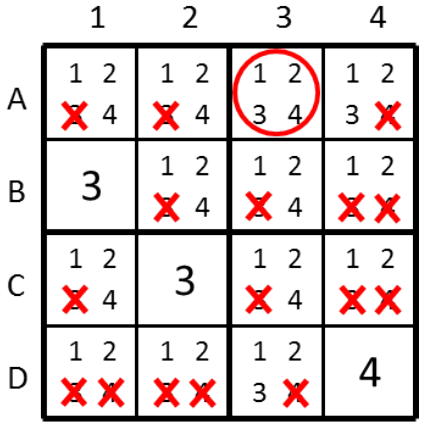


1.d. (2 pts) MINIMUM-REMAINING-VALUES (MRV) HEURISTIC.

See Section 6.3.1.

Consider the 4 × 4 Sudoku board on the left. Possible domain values for each variable are shown in each square. Variables A1 and D4 are assigned and constraint propagation has been done.

List all unassigned variables (in any order) that now might be selected by the Minimum-Remaining-Values (MRV) Heuristic:
A4, D1 .

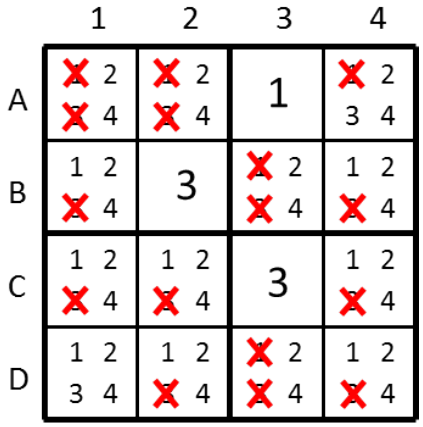


1.e. (2 pts) LEAST-CONSTRAINING-VALUE (LCV) HEURISTIC.

See Section 6.3.1.

Consider the 4 × 4 Sudoku board on the left. Possible domain values for each variable are shown in each square. Variables B1, C2, and D4 are assigned and constraint propagation has been done. A3 (circled) is chosen for assignment next.

List all values (in any order) for variable A3 (circled) that now might be selected by the Least-Constraining-Value (LCV) Heuristic: 3 .

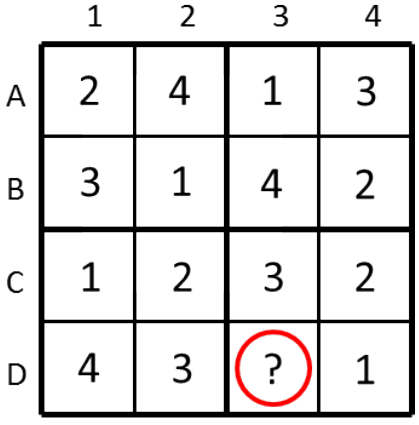


1.f. (2 pts) DEGREE HEURISTIC (DH).

See Section 6.3.1.

Consider the 4 × 4 Sudoku board on the left. Possible domain values for each variable are shown in each square. Variables A3, B2, and C3 are assigned and constraint propagation has been done.

List all unassigned variables (in any order) that now might be selected by the Degree Heuristic (ignore MRV for this problem):
D1 .



1.g. (2 pts) MIN-CONFLICTS HEURISTIC.

See Section 6.4.

Consider the 4 × 4 Sudoku board on the left showing a complete but inconsistent assignment. Current values for each variable are shown in each square. Variable D3 (circled) has just been selected to be assigned a new value during local search for a complete and consistent assignment.

What new value for variable D3 (circled) would be chosen now by the Min-Conflicts Heuristic? 2 .

2. (8 pts total) ONE FISH, TWO FISH, RED FISH, BLUE FISH. (With apologies to Dr. Seuss.)
Amy, Betty, Cindy, and Diane went out to lunch at a seafood restaurant. Each ordered one fish. Each fish was either a red fish or a blue fish. **Among them they had exactly two red fish and two blue fish.**

You translate this fact into Propositional Logic (in prefix form) as:

/* Ontology: Symbol A/B/C/D means that Amy/Betty/Cindy/Diane had a red fish. */

(or	(and A B (¬ C) (¬ D))	(and A (¬ B) C (¬ D))
	(and A (¬ B) (¬ C) D)	(and (¬ A) B C (¬ D))
	(and (¬ A) B (¬ C) D)	(and (¬ A) (¬ B) C D)))

See R&N Section 7.5.2.

Their waiter reported:
"Amy, Betty, and Cindy had exactly one red fish among them; I don't remember who had what. Betty, Cindy, and Diane had exactly one red fish among them; I don't remember who had what."

You translate these facts into Propositional Logic (in prefix form) as:

(or	(and A (¬ B) (¬ C))	(and (¬ A) B (¬ C))	(and (¬ A) (¬ B) C))
(or	(and B (¬ C) (¬ D))	(and (¬ B) C (¬ D))	(and (¬ B) (¬ C) D))

Betty's daughter asked, "Is it true that my mother ha
You translate this query into Propositional Logic as "(¬ B)" and fo

Your resulting knowledge base (KB) plus the negated go

(A B C)	((¬ A) (¬ B) (¬ C))
(A B D)	((¬ A) (¬ B) (¬ D))
(A C D)	((¬ A) (¬ C) (¬ D))
(B C D)	((¬ B) (¬ C) (¬ D))
((¬ A) (¬ B))	((¬ A) (¬ C))
((¬ B) (¬ C))	((¬ B) (¬ D))
((¬ C) (¬ D))	(B)

Think about what you are trying to prove, then find a proof that mirrors how you think. If Betty had a red fish, then nobody else can have a red fish. But among them, they had two red fish. So Betty did not have a red fish. In fact, the same is true for Cindy, so Cindy did not have a red fish. From the problem statement, you can deduce that Amy and Diane had red fish, while Betty and Cindy had blue fish. You were asked only to prove that Betty had a blue fish, not to prove the entire outcome, in order to keep problem length reasonable for a timed test with several other questions.

Write a resolution proof that Betty had a blue fish.
For each step of the proof, fill in the first two blanks with CNF se luce
the CNF result that you write in the third (resolvent) blank. The re t two
sentences. Add the resolvent to KB, and repeat. Use as many st y
clause. The empty clause indicates a contradiction, and therefor ce.

The shortest proof I know of is only four lines long. (A Bo
Longer proofs are OK provided they are correct. *Think about it, then find a proof that mirrors how you think.*

Resolve (A C D) with ((¬ A) (¬ B)) to produce: ((¬ B) C D)

Resolve ((¬ B) C D) with ((¬ B) (¬ C)) to produce: ((¬ B) D)

Resolve ((¬ B) D) with ((¬ B) (¬ D)) to produce: (¬ B)

Resolve (¬ B) with (B) to produce: ( )

Resolve _____ with _____ to produce: _____

Resolve _____ with _____ to produce: _____

Resolve _____ with _____ to produce: _____

Resolve _____ with _____ to produce: _____

The most common error was to cancel two literals simultaneously, of the form:
NO!! Resolve (A B C) with ((¬ B) (¬ C) D) to produce (A D) **NO!!**
But this error has been discussed thoroughly in lecture, **both** in the lecture on propositional logic inference (Thu., 5 Feb., Propositional Logic B, slides #19-23), **and** in the Mid-term Review lecture (Tue., 10 Feb., slides #99-103).

It perplexes me that so many students made so many errors on material that has been covered in so much care and detail during lecture. If you do not attend lecture and discussion section, or if you spend time in lecture or discussion section "multi-tasking" by shopping, playing games, reading email, surfing the web, or visiting social media accounts, then you are placing yourself at a severe disadvantage and **you are losing points on quizzes and exams.**

Please attend both lecture and discussion, and please pay attention.

Other proofs are OK provided that they are correct. For example, another correct proof is:

Resolve B with ((¬ A) (¬ B)) to produce: (¬ A)

Resolve B with ((¬ B) (¬ C)) to produce: (¬ C)

Resolve B with ((¬ B) (¬ D)) to produce: (¬ D)

Resolve (¬ A) with (A C D) to produce: (C D)

Resolve (¬ C) with (C D) to produce: (D)

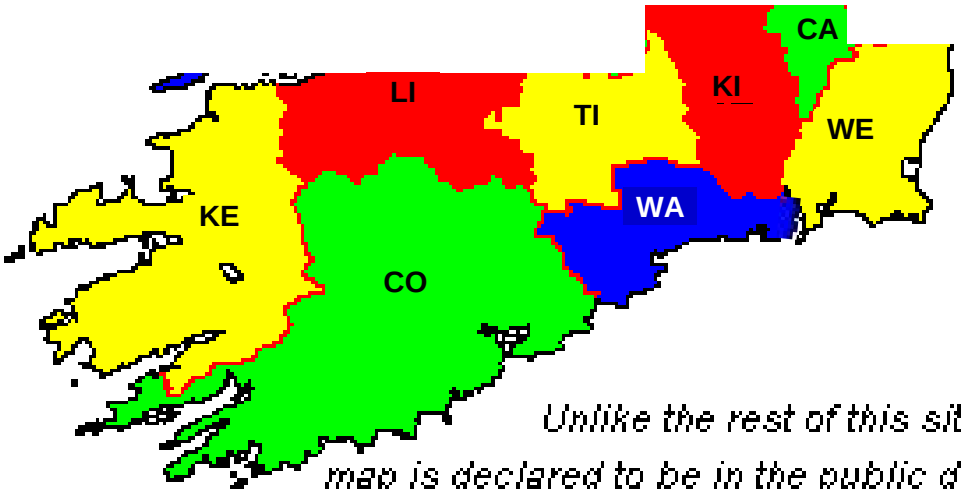
Resolve (¬ D) with (D) to produce: ( )

Resolve _____ with _____ to produce: _____

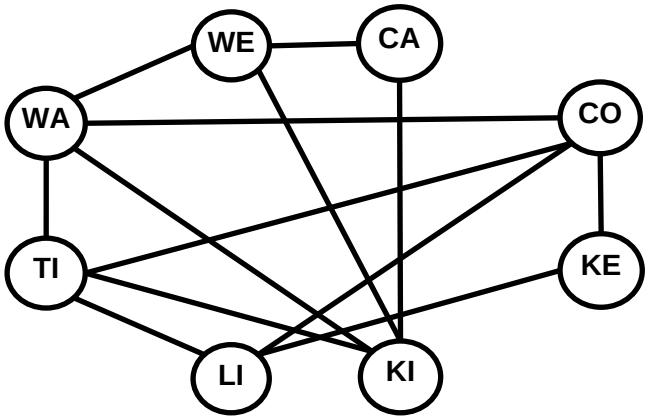
3. (5 points total, -1 pt each error, but not negative) CONSTRAINT GRAPH CONSTRUCTION.

You are a map-coloring robot assigned to color this map of southern Ireland counties. Adjacent regions must be colored a different color (R=Red, B=Blue, G=Green). Please note that all borders between regions are marked by a thin line to highlight the border. This is only for visual clarity, and is not part of any region.

Draw the constraint graph. The nodes are provided. <u>You need draw only the arcs.</u>



CA = Carlow
CO = Cork
KE = Kerry
KI = Kilkenny
LI = Limerick
TI = Tipperary
WA = Waterford
WE = Wexford



See R&N Section 6.1. See also lecture slides for Constraint Satisfaction Problems (slide #9, Tue., 27 Jan.) and Mid-term Review (slide #71, Tue., 10 Feb.).

4. (12 pts total, 2 pts each) VALID, UNSATISFIABLE, SATIS
For each sentence below, write V=valid if the sentence is valid
and S=satisfiable if the sentence is satisfiable but is neither va

See R&N Section 7.5. See also lecture slides for Propositional Logic A (slide #33, Tue., 3 Feb.) and Mid-term Review (slide #97, Tue., 10 Feb.).

4.a. (2 pts) (write V, U, or S) <u>V</u> (A OR (¬ A)) **True in all worlds.**

4.b. (2 pts) (write V, U, or S) <u>U</u> (A AND (¬ A)) **False in all worlds.**

4.c. (2 pts) (write V, U, or S) <u>S</u> (A AND (¬ B)) **True only in worlds with A=TRUE and B=FALSE.**

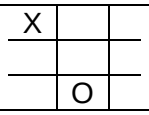
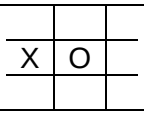
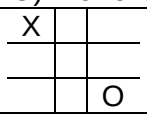
4.d. (2 pts) (write V, U, or S) <u>V</u> ((A AND (¬ A)) **True in all worlds, because antecedent is false.**

4.e. (2 pts) (write V, U, or S) <u>S</u> ((A OR (¬ A)) => B) **True only in worlds with B=TRUE.**

4.f. (2 pts) (write V, U, or S) <u>U</u> ((A OR (¬ A)) => (A AND (¬ A)))

False in all worlds, because antecedent is always true and consequent is always false.

5. (18 pts total, 1 pt each) GAME SEARCH WITH TIC-TAC-TOE AND WIN-PATHS HEURISTIC FUNCTION.
This problem asks about MiniMax Search and Alpha-Beta pruning in Tic-Tac-Toe with the Win-paths static heuristic evaluation function. Recall that the Win-paths heuristic function counts the number of possible win-paths for MAX (= X) and subtracts the number of possible win-paths for MIN (= O). For example:

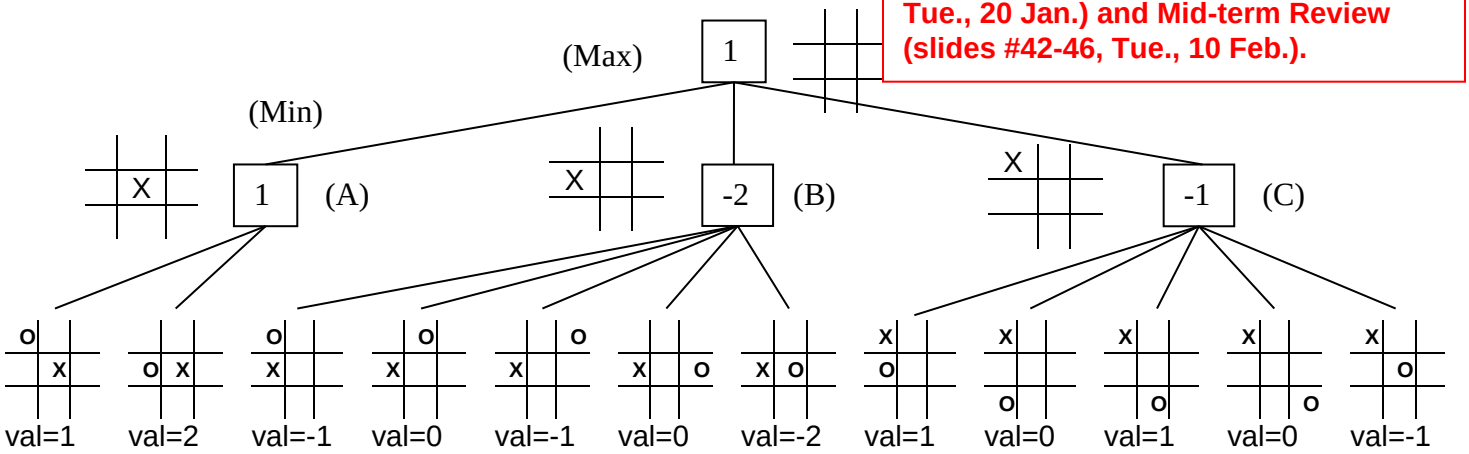
		
MAX (= X) # win paths=6 MIN (= O) # win paths=5 Heuristic value = 1 (= 6-5)	MAX (= X) # win paths=4 MIN (= O) # win paths=6 Heuristic value = -2 = (4-6)	MAX (= X) # win paths=5 MIN (= O) # win paths=5 Heuristic value = 0 (= 5-5)

5.a. (6 pts total, 1 pt each blank branch node [4 in part 1] or answer space [1 each in parts 2&3])
In the game tree below it is **Max**'s (= X's) turn to move. At each leaf node is the estimated score of that resulting position as returned by the Win-path heuristic static evaluator (written below as "val=*n*").
(1) Perform Mini-Max search and label each branch node with its return value (4 branch nodes).

(2) What is Max's best move (A, B, or C)? A

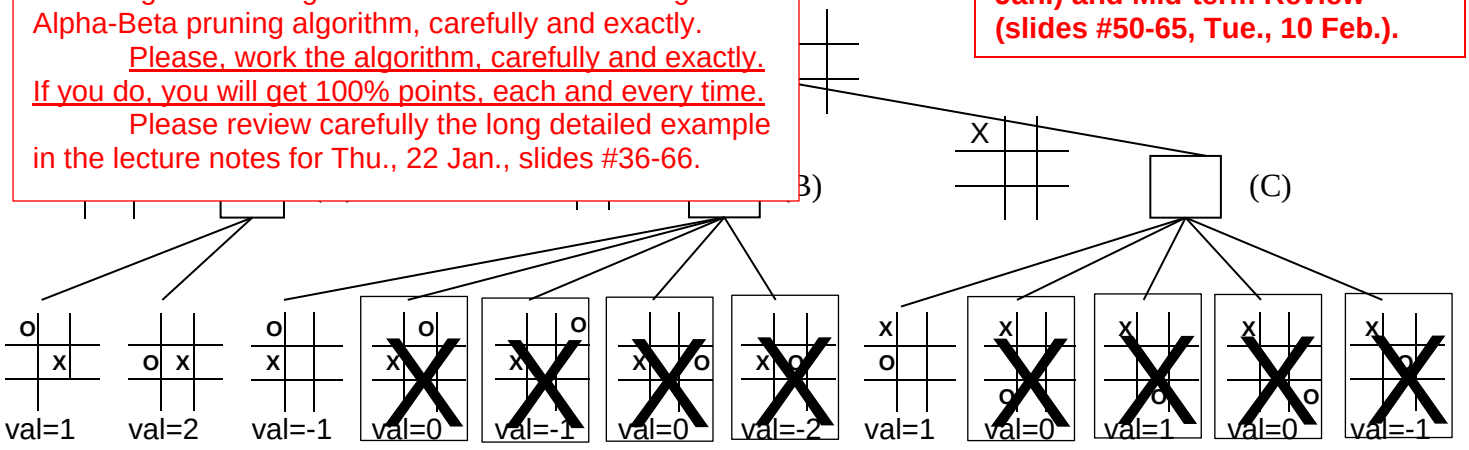
(3) What value does Max expect to get? 1

See R&N Section 5.2.1. See also lecture slides for Games/Adversarial Search/MiniMax Search (slides #9-12, Tue., 20 Jan.) and Mid-term Review (slides #42-46, Tue., 10 Feb.).



Most students got the Minimax part correct, but many students lost points on the Alpha-Beta pruning.
It is always difficult to infer intent; but, from the pattern of answers, many students lost points when they tried to "guess" the right answer instead of working the Alpha-Beta pruning algorithm, carefully and exactly.
Please, work the algorithm, carefully and exactly. If you do, you will get 100% points, each and every time.
Please review carefully the long detailed example in the lecture notes for Thu., 22 Jan., slides #36-66.

See R&N Section 5.3. See also lecture slides for Games/Adversarial Search/ Alpha-Beta Pruning (Thu., 22 Jan.) and Mid-term Review (slides #50-65, Tue., 10 Feb.).



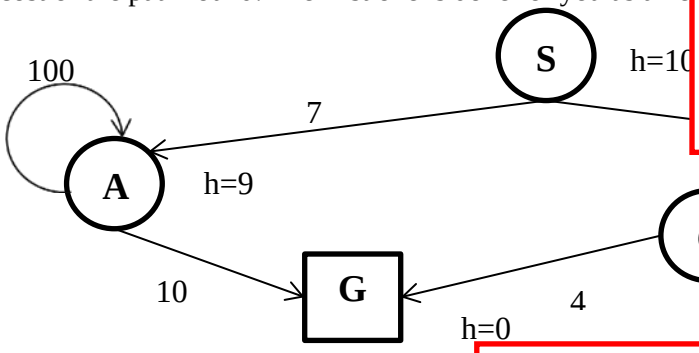
****** TURN PAGE OVER AND CONTINUE ON THE OTHER SIDE ******

6. (15 pts total, 3 pts each) STATE SPACE SEARCH.

Execute Tree Search through this graph (i.e., do not remember visited nodes). Step costs are given next to each arc. Heuristic values are given next to each node (as h=x). The successors of each node are indicated by the arrows out of that node. **Successors are returned in left-to-right order (successors of A are (A, G), and of B are (C, B), in that order).**

For each search strategy below, show the order in whic
children are generated), ending with the goal node that is found
the cost of the path found. The first one is done for you as an e

See your textbook, R&N Chapter 3.
Please see the lecture slides for Uninformed Search, topic "When to do Goal-Test? When generated? When popped?" for clarification about exactly what to do in practical cases.
See lecture slides Uninformed Search (slides #11-17, Thu., 8 Jan.); and Mid-term Review (slides #13-18, Tue., 10 Feb).



DFS does the Goal-test before the child is pushed onto the queue. The goal is found when A is expanded.

6.a. (example) DEPTH FIRST SEARCH.

See Section 3.4.3 and Fig. 3.17.

Order of node expansion: S A G

Path found: S A G Cost of path found: 17

6.b. (3 pts) BREADTH FIRST SEARCH.

BFS does the Goal-test before the child is pushed onto the queue. The goal is found when A is expanded.

See Section 3.4.1 and Fig. 3.11.

Order of node expansion: S A G

Path found: S A G Cost of path found: 17

6.b. (3 pts) UNIFORM COST SEARCH.

UCS does goal-test when node is popped off queue. This precaution avoids the error of finding a short (in steps) expensive goal before a long cheap goal.

See Section 3.4.2 and Fig. 3.14.

Order of node expansion: S B A C G

Path found: S B C G Cost of path found: 12

6.c. (3 pts) GREEDY (BEST-FIRST) SEAR

GBFS has the same behavior whether the goal-test is done before node is pushed or after node is popped. Here, B has the lowest heuristic value (h=2) of any node on the queue, so it just loops.

Order of node expansion: S B B B B B B ...

See Section 3.5.1 and Fig. 3.23.

Path found: none Cost of path found: none

6.d. (3 pts) ITERATED DEEPENING SEARCH.

IDS does the Goal-test before the child is pushed onto the queue. The goal is found when A is expanded.

Order of node expansion: S S A G

See Sections 3.4.4-5 and Figs. 3.18-19.

Path found: S A G Cost of path found: 17

6.e. (3 pts) A* SEARCH.

A* does goal-test when node is popped off queue, for the same reasons as cited above for UCS.

Order of node expansion: S B C G

See Section 3.5.2 and Figs. 3.24-25.

Path found: S B C G Cost of path found: 12

Is the heuristic admissible (Yes or No)? Yes

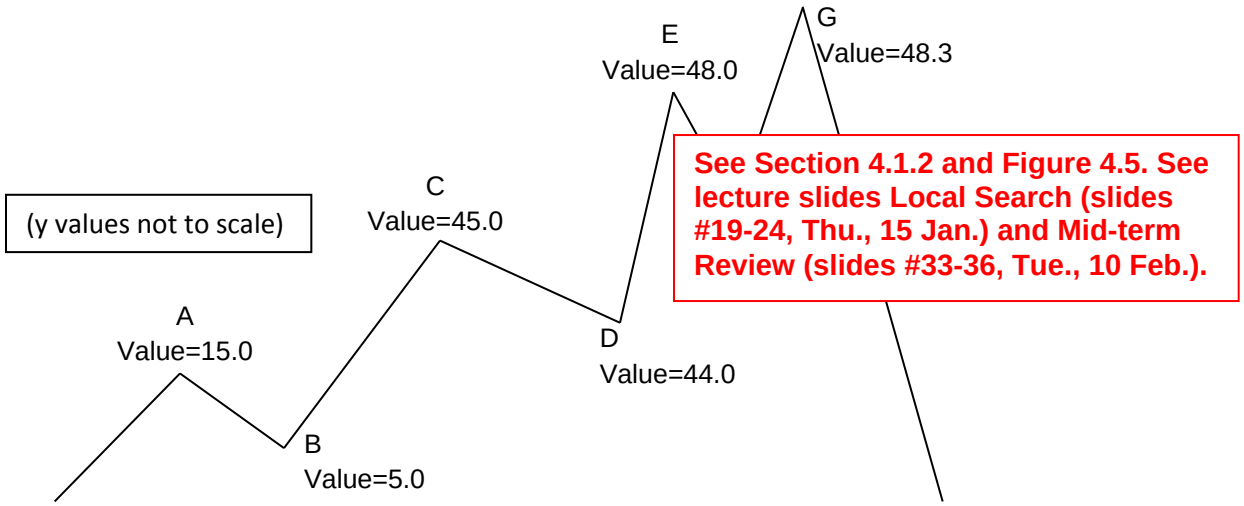
7. (8 pts total, 1 pt each) LOCAL SEARCH --- SIMULATED ANNEALING.
This question asks about Simulated Annealing local search. In the value landscape cartoon below, you will be asked about the probability that various moves will be accepted at different temperatures. Recall that Simulated Annealing always accepts a better move (ΔValue = Value[next] – Value[current] > 0.0); but it accepts a worse move (ΔValue < 0.0) only with probability e^(ΔValue/T), where T is the current temperature on the temperature schedule.

Please use this temperature schedule (usually, it is a decaying exponential; but it is simplified here):

time (t)	1-100	101-200	201-300
Temperature (T)	10.0	1.0	0.1

You do not need a calculator; the values given have been chosen to follow this table:

x	0.0	-0.1	-0.4	-1.0	-4.0	-40.0
e^x	1.00	≈0.90	≈0.67	≈0.37	≈0.02	≈4.0e-18



See Section 4.1.2 and Figure 4.5. See lecture slides Local Search (slides #19-24, Thu., 15 Jan.) and Mid-term Review (slides #33-36, Tue., 10 Feb.).

Give your answer to two significant decimal places. **The first one is done for you as an example.**

7.a. (example) You are at Point A and t=23. **The probability you will accept a move A -> B =** 0.37

7.b. (1 pt) You are at Point B and t=23. **The probability you will accept a move B -> C =** 1.00

7.c. (1 pt) You are at Point C and t=123. **The probability you will accept a move C -> B =** 4.0e-18

7.d. (1 pt) You are at Point C and t=123. **The probability you will accept a move C -> D =** 0.37

7.e. (1 pt) You are at Point E and t=123. **The probability you will accept a move E -> D =** 0.02

7.f. (1 pt) You are at Point E and t=123. **The probability you will accept a move E -> F =** 0.90

7.g. (1 pt) You are at Point G and t=123. **The probability you will accept a move G -> F =** 0.67

7.h. (1 pt) You are at Point G and t=223. **The probability you will accept a move G -> F =** 0.02

7.i. (1 pt) With a very, very, very long slow annealing schedule, are you more likely, eventually in the long run, to wind up at point A or at point G? **(write A or G)** G .

****** TURN PAGE OVER AND CONTINUE ON THE OTHER SIDE ******

8. (10 pts total, 1 pt each) ADVERSARIAL (GAME) SEARCH CONCEPTS.
For each of the following terms on the left, write in the letter corresponding to the best answer or the correct definition on the right.

See Chapter 5.

D	Game Strategy	A	Approximates the value of a game state (i.e., of a game position)
H	Cut-off Test	B	In all game instances, total pay-off summed over all players is a constant
E	Alpha-Beta Pruning	C	Tree where nodes are game states and edges are game moves
G	Weighted Linear Function	D	Function that specifies a player's move in every possible game state
J	Terminal Test	E	Returns same move as MiniMax, but may prune more branches
I	ExpectiMiniMax	F	Optimal strategy for 2-player zero-sum games of perfect information, but impractical given limited time to make each move
C	Game Tree	G	Vector dot product of a weight vector and a state feature vector
A	Heuristic Evaluation Function	H	Function that decides when to stop exploring this search branch
B	Zero-sum Game	I	Generalizes MiniMax to apply to games with chance (stochastic games)
F	MiniMax Algorithm	J	Function that says when the game is over

9. (10 pts total, 1 pt each) CONSTRAINT SATISFACTION PROBLEM (CSP) CONCEPTS.
For each of the following terms on the left, write in the letter corresponding to the best answer or the correct definition on the right.

See Chapter 6.

G	Minimum Remaining Values Heuristic	A	Set of allowed values for some variable
H	Degree Heuristic	B	Specifies the allowable combinations of variable values
J	Min-Conflicts Heuristic	C	Every variable is associated with a value
E	Solution to a CSP	D	The values assigned to variables do not violate any constraints
I	Least Constraining Value Heuristic	E	A complete and consistent assignment
A	Domain	F	Nodes correspond to variables, links connect variables that participate in a constraint
B	Constraint	G	Chooses the next variable to expand to have the fewest legal values in its domain
D	Consistent Assignment	H	Chooses the next variable to expand to have the largest number of constraints on other unassigned variables
C	Complete Assignment	I	Prefers to search next the value that rules out the fewest choices for the neighboring variables in the constraint graph
F	Constraint Graph	J	Select the value that results in fewest conflicts with other variables

****** THIS IS THE END OF THE MID-TERM EXAM ******