CSci 4511
Final

Name: ______________________________

Student ID: ______________________

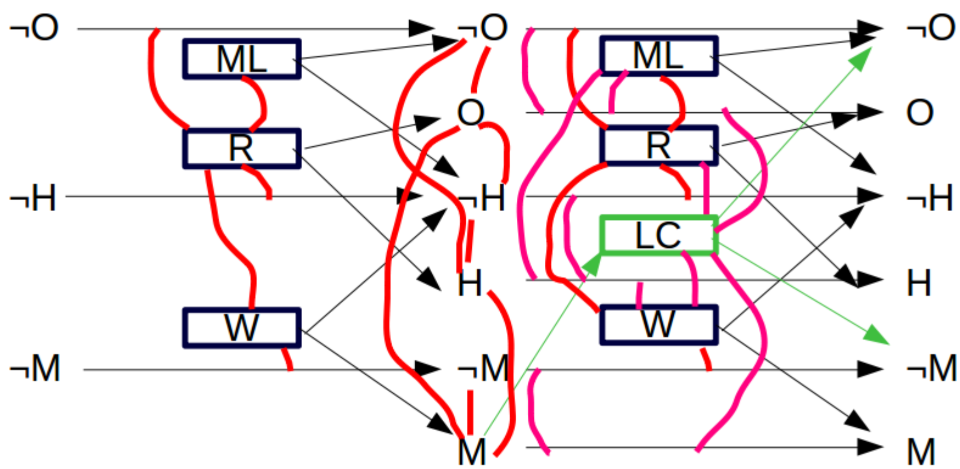
Instructions: The time limit is 120 minutes. Please write your answers in the space below. If you need more space, write on the back of the paper. The exam is open book and notes. You may not use the internet or any other outside resources. Usage of phones during the test is not allowed. For all questions you must **show work** to receive full credit.

Problem (1) [15 points]
Convert these English sentences into equivalent first order logic:
(1) "Moons are large natural objects that orbit a planet."
(2) "Moons can only orbit a single planet, not multiple."

Convert this sentence into conjunctive normal form (CNF):
$\forall x\Big(A(x) \wedge [\exists y\ B(x,y)] \Rightarrow [\exists y\ C(x,y)]\Big)$

Problem (2) [20 points]
Consider the graphplan shown below (might look familiar). All mutexes have been filled in except for the final state level on the right. The mutexes in the middle level are the pairs: $[(\neg O, O), (\neg O, H), (O, \neg H), (O, M), (\neg H, H), (H, M), (\neg M, M)]$. Add the missing state mutexes on the final state level (far right side).

Actions:	$MowLawn()$	$Lawncare()$	$Relax()$	$Work()$
Preconditions:		$Money$		
Effects:	$\neg Overgrown \wedge \neg Happy$	$\neg Overgrown \wedge \neg Money$	$Happy \wedge Overgrown$	$Money \wedge \neg Happy$



Problem (3) [25 points]
Consider the following knowledge base (KB):
$(A \vee \neg B), (B \vee C \vee D), (A \vee C \vee \neg D), (\neg A \vee D), (\neg A \vee B \vee \neg D), (\neg B \vee \neg C \vee \neg D)$

(1) Is the following entailed? $KB \vDash (\neg A \vee C)$
(2) Is the following entailed? $KB \vDash (C \vee D)$

Problem (4) [15 points]
Generate all the possible sentences using forward-chaining (there is no "goal for entailment"). You must show clearly your unification/substitution for full credit. (Though, you should show work on all problems.)

1.$\forall x, \exists y\ [A(x) \Rightarrow B(x) \vee D(y)]$
2.$\exists x, \forall y\ [B(x) \wedge C(y) \Rightarrow E(y,x)]$
3.$\forall x\ [A(x) \wedge B(x) \Rightarrow C(x)]$
4.$\forall x, y\ [A(x) \wedge D(y) \Rightarrow E(x,y)$
5.$\forall x, y, \exists z\ [E(x,y) \Rightarrow E(y,z)]$
6.$\forall x\ [A(x) \Rightarrow D(x)]$
7.$\exists x\ [A(x)]$
8.$\exists x\ [B(x)]$
9.$\forall x\ [C(x) \vee D(x)]$

Problem (5) [15 points]
Suppose you had two actions from a planning problem that quite often done one after another. For example, when you go to the bathroom you then immediately wash your hands (... right?!). If you wanted to combine these actions into a single action, describe what the resulting combined preconditions and effects would be. So in the example if you combined “bathroom” + “wash hands” to give the “sanitary bathroom” action, discuss what the resulting preconditions and and effects of ”sanitary bathroom” are (based on “bathroom” and “wash hands”).
You must discuss these preconditions and effects as a (clear) general rule and **not** for just a specific example. You should also state and justify under what conditions you could not combine two actions.

Problem (6) [10 points]
The basic breadth-first-search algorithm is exponential (b^d) runtime. By running bi-directional search instead, you essentially half the search depth, giving a runtime of ($b^{d/2}$). If you are doing mini-max you need to search the whole tree and the runtime we said was (b^d). If alpha-beta pruning is done effectively, you can get a runtime of ($b^{d/2}$).

Identify the two ways in which bi-directional search and alpha-beta pruning are different (along with justification). Then identify one way (in which they are most similar), again with justification. (Other than the obvious things like they are both trees or have the same runtime.)

Problem (7) [20 points]
For the following cases, cite a specific algorithm you think would be most efficient/effective to solve the problem (along with the type of problem this algorithm falls under, such as "search" or "propositional logic"). State what you think the second-best algorithm is (no restrictions on choice) as well and give a critical analysis of specifically why your "best" algorithm is in fact better.

(1) You are part of a chemist's laboratory testing compositions of medals trying to find a new cheap and durable metal. You have a bunch of abundant materials and need to figure out the ratios between them to get the best type of metal.

(2) You design the layout of the inner parts of a laptop product. You know the specs for the machine (CPU speed, memory amount, graphics card, keyboard size, touch-pad, etc.), but need to decide where to put each part. Many parts are fixed (such as the keyboard layout) and others cannot be put somewhere stupid (like the touch-pad on the bottom of the laptop or the fans blowing burning hot air towards the user). Obviously all the parts need to fit inside the case as well, with the "hot" items (like CPU and graphics card) near the edge of the computer and not next to each other.