ANSWERS to TDT4136-2020 Fall exam

SEARCH

a)

- Initial_state: The initial state is the starting star system S
- States: The states are the set of all states reachable from the initial. state by a sequence of actions.
- Actions: The actions are the possible action available in a state. Given a state s, ACTIONS(s) returns all actions that can be performed in s. For instance ACTIONS(C) = {Go(S); Go(B); Go(H); Go(G)}

• Path_cost: The path cost is the sum of costs of the individual actions along the path. They are shown on the graph. For instance COST(In(S);GO(B)) = 1

• Transition_model: Given a state and action, this returns the resulting planet: RESULT(In(B); Go(C)) = In(C)

• Goal_test: The goal test checks whether a given state is the goal state. Here the goal is the singleton set {In(G)}

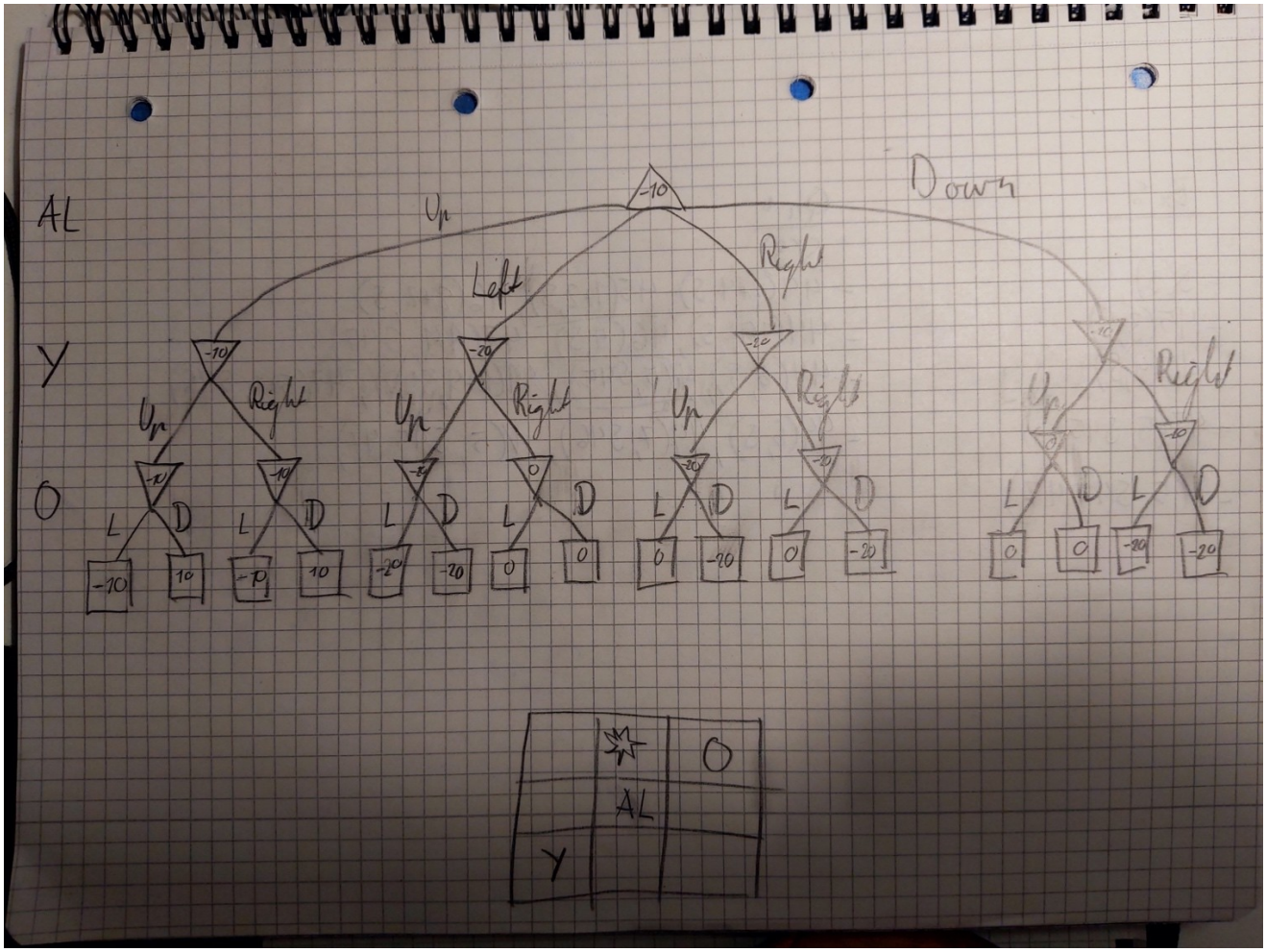
b)

Explored	**Frontier**
	S(5)
S(5)	C(4,SC), B(5,SB)
S(5), C(4, SC)	H(6, SCH), G(8, SCG), B(5, SB)
S(5), C(4, SC), B(5, SB)	H(6, SCH), G(8, SCG)
S(5), C(4, SC), B(5, SB), H(6, SCH)	~~G(8, SCG)~~, G(6, SCHG)
S(5), C(4, SC), B(5, SB), H(6, SCH), **G(6, SCHG)**	

c)

• Path found: SCHG, cost = 6. Not optimal.
• Reason: Heuristic is inconsistent.
• Fix: Change e.g h(C) = 3, then A* will find SBCHG which has a cost of 5 and is optimal.

d)



e). See Figure 2 below.
a: No nodes are pruned.
b: Search the tree from a right to left manner instead.
This corresponds to rearranging the nodes so that the three looks like. See figure 3

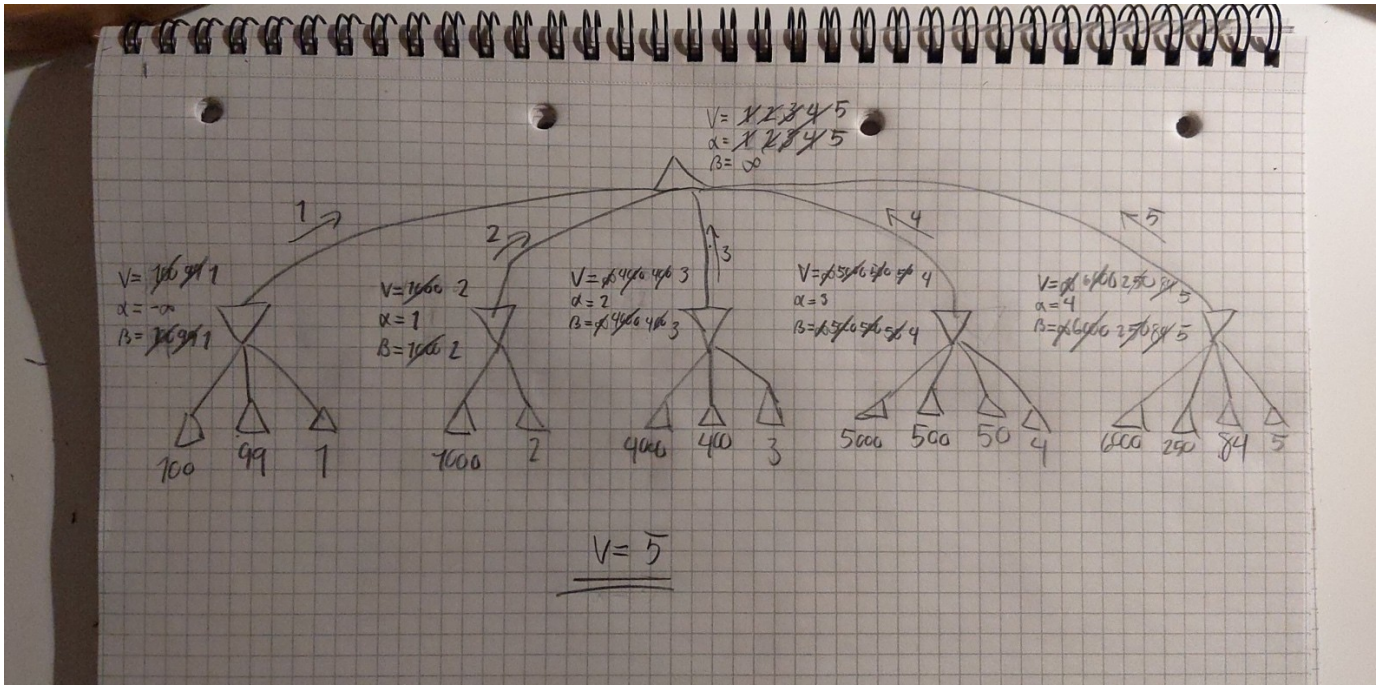


Figure 2

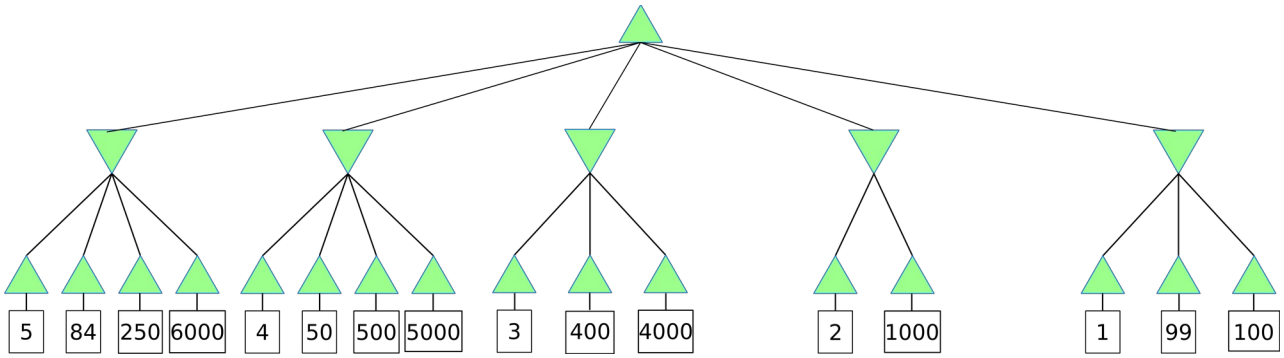


Figure 3

f). Found solution is not optimal, optimal solution would be SCDG

Explored	Frontier	Current Node
	S	
S	A(S), B(S), C(S)	S
S, A(S)	B(S), C(S)	A(S)
S, A(S), B(S)	C(S), D(S, B), E(S, B)	B(S)
S, A(S), B(S), C(S)	D(S, B), E(S, B)	C(S)
S, A(S), B(S), C(S), D(S, B)	E(S, B), G(S, B, D)	D(S, B)

Path found: SBDG
It is NOT the optimal path.
The optimal path is: SBEG

LOGIC

1) In this case it doesn't really matter which representation one picks, both can represent the board with the same amount of symbols. Because it doesn't matter I just picked propositional logic.

$Sparkle\,s_{x,y} = ¿$ Square (X, Y) has sparkles
$Diamond\,s_{x,y}$ = Square X Y has diamonds
$RedGlo\,w_{x,y}$ = Square X Y has a red glow
… and so on for Lava and Player
Board state is then:
$Sparkle\,s_{1,1} \wedge Diamond\,s_{1,2} \wedge RedGlo\,w_{1,2} \wedge Sparke\,s_{1,3} \wedge Lav\,a_{1,3} \wedge RedGlo\,w_{2,1} \wedge Sparkle\,s_{2,2} \wedge Sparkle\,s_{2,3} \wedge Re$

2)
1) $\forall x \forall y (Square(y) \wedge Square(x) \wedge Adjacent(x, y) \wedge Has(y, Diamonds) \Rightarrow Sparkles(x))$
2) $\forall x \forall y (Square(y) \wedge Square(x) \wedge Adjacent(x, y) \wedge Has(y, Lava) \Rightarrow RedGlow(x))$
3) $\forall x (Square(x) \wedge Has(x, Player) \wedge Has(x, Lava) \Rightarrow GameOver)$
4) $\neg \exists y (Diamond(y) \wedge \neg Has(Player, y)) \Rightarrow GameWo\,n$
5) $\forall x \forall z (Square(x) \wedge Player \wedge Diamond(z) \wedge Has(x, Player) \wedge Has(x, z) \Rightarrow Ha\,s(Player, z))$
6) $\forall x ((Square(x) \wedge CanMove(Player, x) \wedge \exists y (Square(y) \wedge Adjacent(y, x) \wedge Has(y, RedGlow))) \Rightarrow \exists y (S$
7) $\forall x \forall y (Square(x) \wedge Square(y) \wedge CanMove(Player, x) \wedge Has(x, Player) \Rightarrow Adjacent(x, y))$

3) No, the player can not win. This is because of rule 6. The player can never move to (2, 1) because it has a red glow but does not sparkle, which means that the player can never move to (2, -1)

4)
a, b, d already in CNF
c: $\forall x \forall y (Person(x) \wedge Afraid(x, y) \Rightarrow \neg Close(x, y))$
gives $\forall x \forall y (\neg Person(x) \vee \neg Afraid(x, y) \vee \neg Close(x, y))$ because implication is equivalent to this
gives $\neg Person(x_1) \vee \neg Afraid(x_1, y_1) \vee \neg Close(x_1, y_1)$ by removing universal quantifiers.
E: Same steps as c gives

$$\neg Close(x_2, y_2) \vee \neg Close(y_2, z_2) \vee Close(x_2, z_2)¿$$

F: Same steps as c gives

$$\neg Close(x_3, y_3) \vee Close(x_3, y_3)$$

G: Same steps as c gives

$$\neg CanMine(x_4, Diamonds) \vee Close(x_4, Diamonds)$$

“Richard does not mine diamonds” is

$$\neg CanMine(Richard, Diamonds)$$

So refutation is:
H: $CanMine(Richard, Diamonds)$
H and G gives I: Close(Richard, Diamonds)
B and D gives J: $\neg Afraid(Richard, y_1) \vee \neg Close(Richard, y_1)$
A and J gives K: $\neg Close(Richard, Lava)$
K and E gives L: $\neg Close(Richard, y_2) \wedge \neg Close(y_2, Lava)$
L and I gives M: $\neg Close(Diamonds, Lava)$
M and D gives empty, concluding proof by resolution refutation.

5)
Here RedGlow = RG, Diamonds = D
a. Valid, RedGlow may be false
b. Neither, RedGlow can’t be false and true at the same time
c. Valid, RedGlow is always either true or false
d. Valid, equivalent to c
e. Satisfiable, true when RedGlow and Diamonds have same truth value
f. Satisfiable, true when RedGlow and Diamonds do not have same truth value
g. Satisfiable, equivalent to $(\neg RedGlow \wedge \neg Diamonds) \vee (RedGlow \wedge Diamonds) \vee RedGlow$ which is true f.ex. when RedGlow is true and false when RedGlow is false and Diamonds is true.

Constraint Satisfaction Problem:

1. Domains based on the favorite colours and plant types of persons :
   Peter: {LG-Calat1, LG-Calat2, LG-Phil1}
   Anette: { V-Calat3, B-Phil2, DG-Calat3 }
   Rudolf: {V-Phil1, V-Calat1, B-Phil2 }
   Daisy: { V-Calat1, B-Phil2}
   Femke: { V-Phil1, V-Calat1, V-Calat2, V-Calat3, DG-Phil1, DG-Calat1, DG-Calat2, DG-Calat3, Y-Phil1, Y-calat1, Y-Calat2, Y-Calat3}

2. Search tree for “backtracking with forward checking”.

   Assign Peter: LG-Calat1, do inference:
   Some domains change because Peter now has “reserved” LG colour and Calat1 plan.t Nobody else can take them. Below are all the domains:

   A={V-Calat3, B-Phil2, DG-Calat3} ,
   R={V-Phil1, B-Phil2}
   D has a constraint with P. Plant(P)= Plant(D). Then
   D={-) Empty. Backtrack.

   Assign Per: **LG-Calat2**
   Anette: { V-Calat3, B-Phil2, DG-Calat3 }
   Rudolf: {V-Phil1, V-Calat1, B-Phil2 }
   Daisy: { V-Calat1}
   Femke: { V-Phil1, V-Calat1, ,V-Calat3, DG-Phil1, DG-Calat1, DG-Calat3, Y-Phil1, Y-calat1, Y-Calat3}

   Assign Anette= V-Calat3. Do forward check
   Rudolf: {B-Phil2 }
   Daisy: { } Empty. Stop backtrack

   Assign Anette= B-Phil2
   Rudolf: {V-Callat1, V-Phil1 }
   Daisy: {V-Calat1}
   Femke: {V-Phil1, DG-Phil1, Y-Phil1}

   Assign Rudolf= V-Calat1
   Daisy={} Empty . Backtract

   Assign Rudolf= V-Phil1
   Daisy={} Empty . Backtract

Assign Anette= DG-Calat3
Rudolf: {V-Phil1, V-Calat1, B-Phil2 }
Daisy: { V-Calat1 }
Femke: { V-Calat1 }. Must have same plant category as Anette, i.e., Calat.

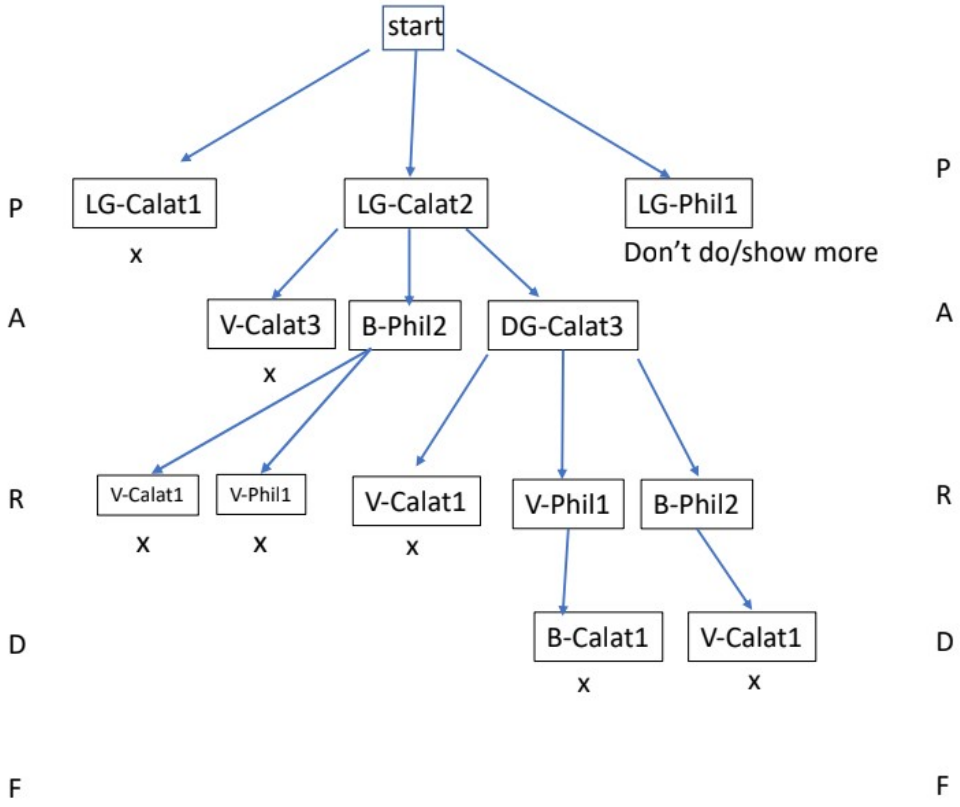
Assign Rudolf= V-Calat1
Daisy: {- }Empty, backtrack

Assign Rudolf= V-Phil1
Daisy: { B-Calat1} must have same same plant category as P, which is Calat
Femke: { Y-Calat1} . must have same plant category as Anette

Assign Daisy= B-Calat1
Femke: { -} Empty, Backtrack

Assign Rudolf= B-Phil2
D: {V-Calat1}
Femke: {Y-calat1, Y-Calat1} same category as Anette.

Assign D= V-Calat1
Femke: {-} Empty backtrack



3. Search tree for "**backtracking search with forward checking and propagating through domains that are reduced to singleton domains.**"

Assign P: LG-Calat1.
Do inference: Some domains change because person P now has “reserved” LG colour and Calat1 plan. Nobody else can take them. Below are all the domains:

A={V-Calat3, B-Phil2, DG-Calat3} ,
R={V-Phil1, B-Phil2}
D has a constraint with P. It cannot have Calat1 but must have a Calat plant because Plant-category(P) = Plant-category(D). Then
D={ -}. Empty domain. Backtrack.

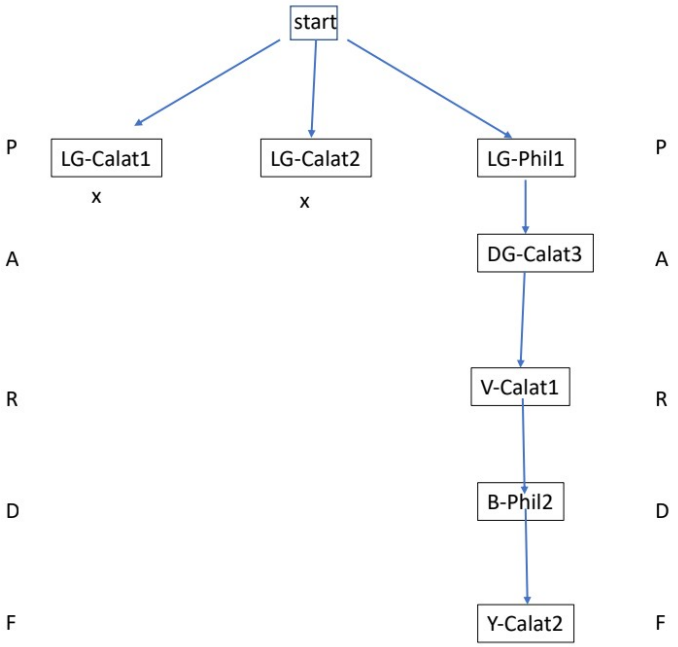
Assign P= LG-Calat2
Anette: { V-Calat3, B-Phil2, DG-Calat3 }
Rudolf: {V-Phil1, V-Calat1, B-Phil2 }
Daisy: { **V-Calat1**}. Singleton domain, propagate - Constraint on plant category with P.

Anette: { B-Phil2, DG-Calat3 }
Rudolf: {**B-Phil2** } Singleton domain, propagate

Anette: {**DG-Calat3**}. Singleton propagate
Femke cannot have LG, V, B, DG. Cannot have Calat2,Calat1,Calat3, and Phil2. Must have the same plant category as Anette which is Calat.
Femke:{Empty} Backtrack

Assign **P= LG-Phil1**
Anette: {V-Calat3, B-Phil2, DG-Calat3 }
Rudolf: {V-Calat1, B-Phil2}
Daisy: **{ B-Phil2**}-since must have plant in same category as Peter

Anette: { V-Calat3, DG-Calat3 }
Rudolf: {**V-Calat1}** singleton
Anette: {**DG-Calat3** }. Cannot have V because of Rudolf propagation.
Femke: {Y-Calat2}



PlANNING

1)

a) Flaw1. Precondition Smooth(B) of paint(B) is not satisfied
Flaw2. putOn(A,B) threatens paint(B) –because it makes A not free
Flaw3. putOn(A,B) threatens paint(A) - –because it makes B not free
Flaw4. putOn(A,B) threatens sand(A) - because it makes A not free
So, need for ordering links.

b) Flaw 1: add action. Sand(B)
Flaw 2: paint B must be executed before putOn(A,B), i.e., an ordering link is added
Flaw 3: paint B must be executed before putOn(A,B), i.e., an ordering link is added
Flaw 4: putOn(A,B) comes after sand(A) as well as newly added sand(B)

c) (sand(A) OR sand(B)) ; (Paint(A) OR paint(B)) ; putOn(A,B)

2) No constraints, they can be executed in parallel.
3) Both return TIK as the plan. Its postconditions (C and D) satisfies the goal $C \wedge D$.

GAME THEORY :

1) The Nash equilibrium is when all students chose the same integer. No student will have an incentive to move away from that number because moving to any other number will end up in zero payoff. There is no other Nash because any student who has chosen a less-often-chosen number would benefit from switching to the most-often one. Also if there are two numbers that are most often, students who have chosen these will benefit from switching to the other most-often number.

2) All strategy profiles except (B,B) are pareto optimal. (B,A) is also social optimum.
3) We eliminate a and b because c dominates them. Then on the remaining matrix, we can eliminate S and R as they both are dominated by T. Then the solution is (c,T).

SHORT QUESTIONS

1) LOCAL SEARCH QUEST

(a) Eval(a)= 1- # attacking pairs(a)= 1-5= -4
Eval(b) = 1-# attacking pairs(b)= 1-9= -8

(b) See textbook p.126, Figure 4.5. Simulated Annealing algorithm. The alg decides to move if Delta-E> 0. If smaller, then moves with probability $e^{(Delta-E/T).}$
Delta E= (1-9)-(1-5)= -4. Moves with probability 1/e.

2) Ethical Issues

This is an open question and I wanted to see how students think/reason. Any justified answer is accepted. Student connect consequentialism to the whole

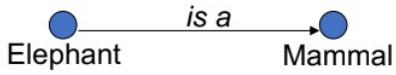
agent notion, rationalism, search strategies, more specific search problems such as adversarial search, as well as game theory. The connection with the game theory and the utilities is particularly interesting.

3) PEAS and Characteristics of the Environment

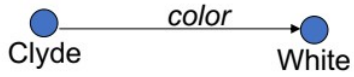
Environment: partially observable, deterministic (but may be stochastic if the sensor is not good and the robot may be blown away by the mine), continuous, static, sequential (mines are most probably places with a some distance between them. If one found, then the next one will be at least in 30 cm distance). The answers may differ based on the assumptions. So, various answers are accepted here.

4) Translation from Semantic Networks to Logic

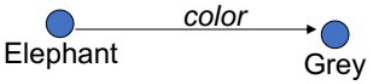
Translate the sematic network representations shown in the figure to logic representations. The following is the outset. I accepted also variations such as Isa(elephant,mammal), color(elephant, grey) type of answers without quantification.




```
(∀x)(elephant(x) -> mammal(x))
```




```
color(clyde, white)
```




```
(∀x)(elephant(x) -> color(x, grey))
```