AI notes

Saturday, November 9, 2024 9:09 PM

15/7/24

Artificial Intelligence

Mendane Task - Analog

- Common sense reasoning
- NLP

Formal Tasks -

- Game playing
- Theorem proving

Expert Tasks - analysis -
↳ expert system

stock predict", medical analysis

knowledge - change, hard to characterise

AI Technique -
Properties -
1) Generalisa^n
2) Work like human being.

<u>Tic-tac-toe game algo:</u>

Board - 3x3 - 2d array
1 player user.

2 - ϕ
3 - X
5 - O

1	2	3
4	5	6
7	8	9

posswin (P) ⇒ Return 0 if P aint win
else return value

$3 \times 3 \times 2 = 18 >$

$5 \times 5 \times 2 = 50$

go(n)

makemove(): 2, 4, 6, 8

Posswin not needed till atleast two moves are made by anyone.

X - ① 1st move - center sq. is best
↳ go(5)

O - ② if Board[5] == ϕ then go(5)
else go(1)

X - ③ if Board[1] == ϕ then go(1)
else go(3)

winning possibility of X starts from here, so posswin will be needed.

∴ now O will have to chk & call posswin before directly playing.

O - ④ If (posswin (X) ≠ 0) then
go(posswin(X))
else go(7)

X - ⑤ if Board[7] = ϕ & posswin (O) ≠ 0
~~posswin (X)~~ go(7)
else go(9)

⑥ If posswin(X) ≠ 0 then
go(posswin(X))

Modifying the board config:

6	1	8
7	5	3
2	9	4

all row sum 15

→ this is called magic square

But this still isn't AI technique

Static evaluat^n fxn OR Heuristic for 1-ply search: Direct

fxn Range → +10 to -10

10 → X → max player win (starts game)

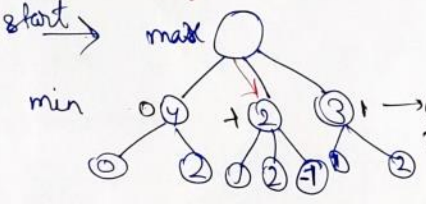
-10 → O - win for min player

zero → draw

2ply: " consider's opponent's choices & its next moves.

Min-Max Algorithm

start →

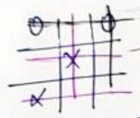


→ among these for next player min val is -1 & max is 1

So as next is max then if 1 is chosen it would be more benefitial for him & if -1 is choosen it would be harmful

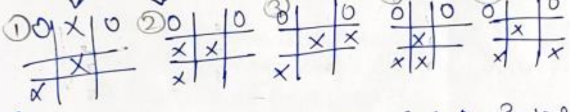
∴ max का worst here is -1 so playing /following -1 path will be most benefital for min.

X = max player.



two win possibilities for O
→ 3 win possibilities for X
→ score = 3 − 2 = 1

Heuristic fxn = score = open win possibility of max − open win possibility of min



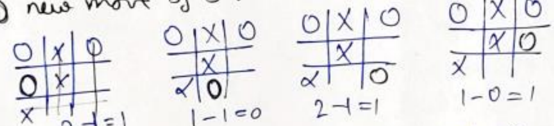
① ② ③ ④ ⑤

3 − 1 = 2   3 − 2 = 1   3 − 1 = 2   3 − 2 = 1   3 − 1 = 2

If we consider 1-ply search 2 score वाले will be best i.e. ①, ③, ⑤

→ max val 2
→ min val 1

for ① new move of O :

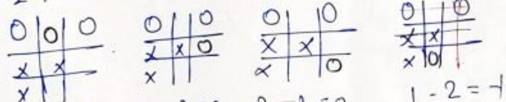


2 − 1 = 1   1 − 1 = 0   2 − 1 = 1   1 − 0 = 1

1
0

O की turn में इतना min return

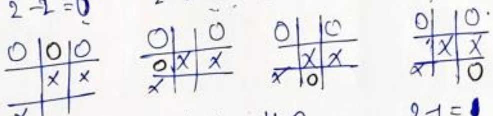
②:



2 − 2 = 0   2 − 2 = 0   2 − 2 = 0   1 − 2 = −1

0
−1

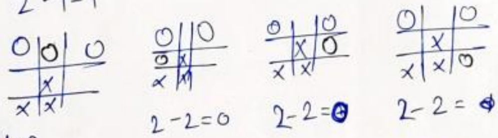
③:



2 − 1 = 1   2 − 1 = 1   H: 0   2 − 1 = 1

1
0

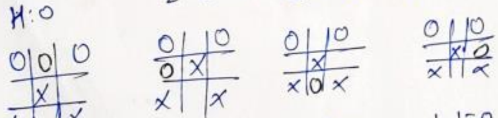
④:



H: 0   2 − 2 = 0   2 − 2 = 0   2 − 2 = 0

0

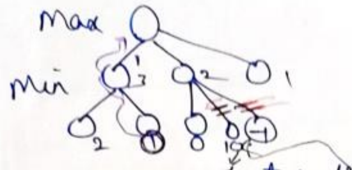
⑤:



2 − 1 = 1   2 − 1 = 1   1 − 1 = 0   1 − 1 = 0

1
0

22/7/24 α-β-prunning:

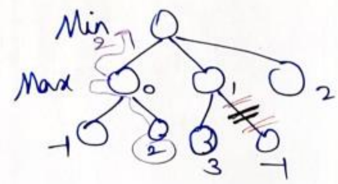


$\alpha = 1 =$

If getting val >1 then only we shld proceed.

If less than 1 is got then we can prun (stop) the search as min player will choose that.

why to explore here onwards as min will 0 or anything lesser than 0 or will be 0. so this seach needn't be done.



$\beta = 2 =$

3 is > 2 ∴ can prun.

-Min player choose 0 → then backtrack returns 2,

If > 2 then prun as max player will choose that so 3 choose, ∴ prun ~~ahead~~ next.

max

min → return max val.

max → return/backtrack smallest val.

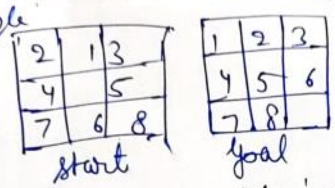
min

→ return min val

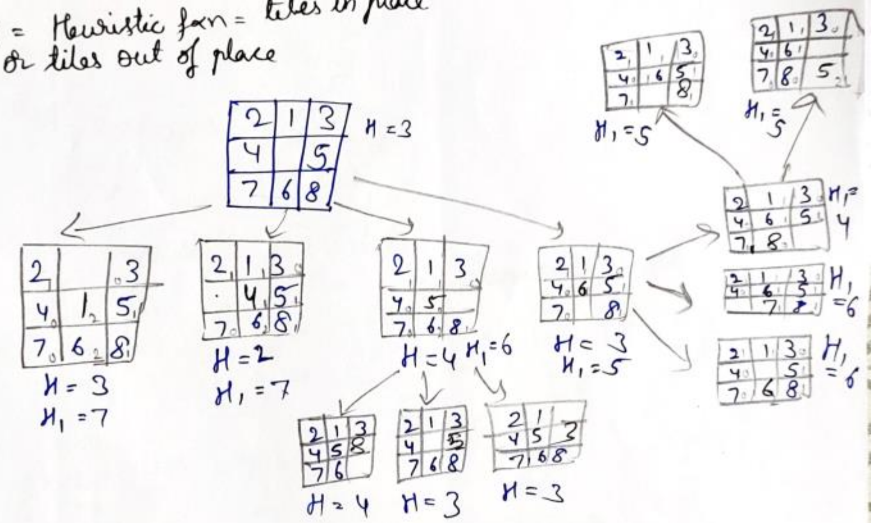
max

→ return max val

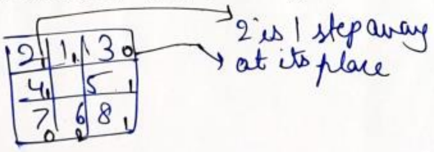
8 puzzle



H = Heuristic fxn = tiles in place
or tiles out of place

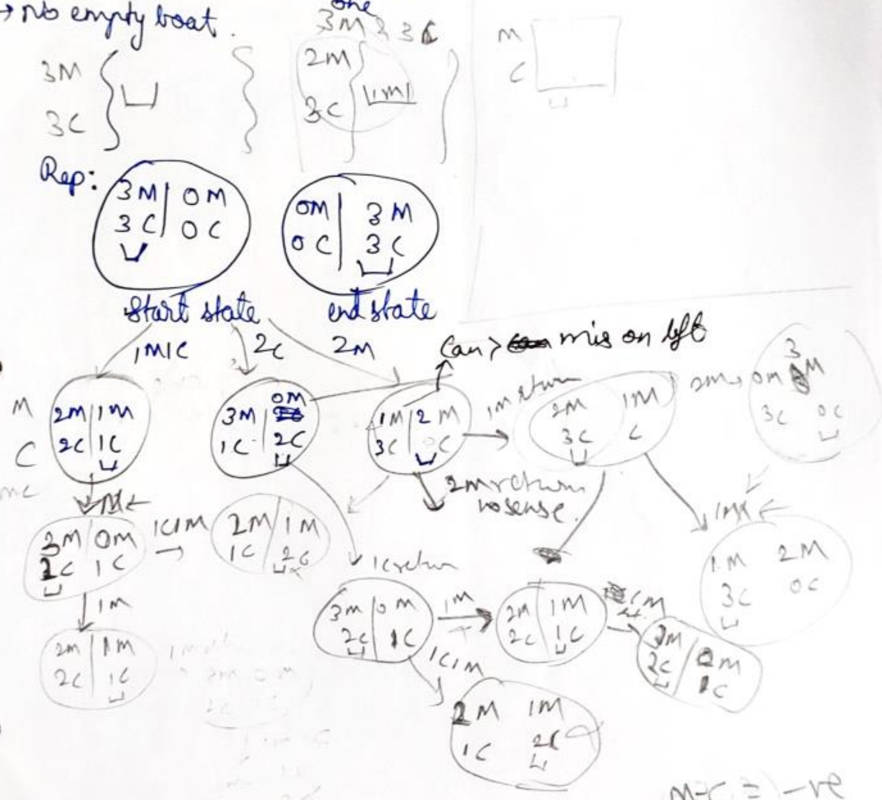


H_1 = manhattan dist sum of all tiles



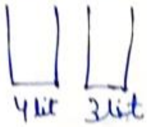
→ Max 2 ppl
→ No. of mis shld not be less than can on one side.
→ no empty boat.

Normal soln:



M-C = -ve
when M≠0
0M-2C = -2
is valid.

Water jug problem Q. – 2 lit in 4 lit jug.



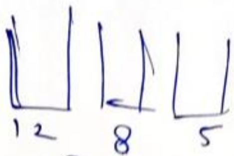
(0,0) start

(4,n) goal

(0,0) every successor gives solⁿ

(0,0) → (4,0) → (1,3) → (1,0) → (0,1) → (4,1) → (2,3)

(0,0) → (0,3) → (3,0) → (3,3) → (4,2) → (2,0)



(0,0,0) start

(1,0,0) end.

(0,8,5) → (12,1,0)

(12,0,0) → this successor can't get solⁿ.

Environment Types

~~At location~~ Discrete
limited, clearly defined percepts & acts.
eg. vaccuum cleaner

Continuous

Partially observable
eg. autonomous taxi driver.

Fully observable
eg. 8 puzzle chess. all toy games

Single Agent
- single player

Multiagent
- multiple player

Deterministic
Next state is determined by current state & the actn executed by agent.
eg.

Stochastic

Episodic

Sequential
Next state is determined by prev. state i.e. episode.

eg. shortest path
eg = 8 puzzle

Static

Dynamic

eg. magic sq.
eg. tic tac toe - all multiplayer games

Char of problems — Models part

Q1) Whether the problem is decomposable?
↳ Episodic.

Q2) Can sol^n step be ignored or undone if proven unwise?

Q3) Is the problem's universe is predicatable?
↳ single player

Q4) Is the sol^n absolute or relative?
↳ paragraph — ↳ shortest path
yes – no ans ↳ chess
↳ water – jug

Q5) Is the sol^n in single state or path to a state? — water jug
travelling salesman, shortest path.

Q6) Is explicit knowledge required?

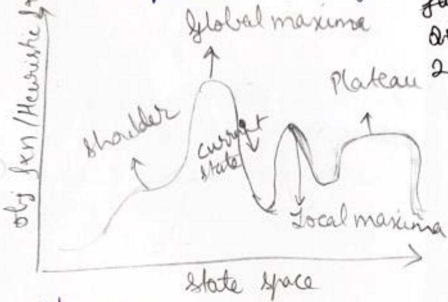
Q7) Does the problem require interact^n with human or needs human interface?

Hill Climbing Algorithm
↳ local search algorithm

Types ~~Simple - blockgame~~
↳ Steepest Ascent hill climbing
↳ First-choice hill climbing
↳ Simulated annealing

Plateau - flat, heuristic value is same
Ridge - Area higher than its surroundings but has a slope so it can't be reached in single move → any pt is low so looks like sol^n
Shoulder - plateau with uphill edge

overcome plateau - make bigger jumps
overcome ridge - make use of 2+ rules before testing



Adv -
↳ simple optimizat^n algo.
↳ Can be used to solve variety of optimizat^n problems having large search space & complex constraints
↳ efficient in finding local minima / optima where good sol^n is needed quickly.

Disadv -
↳ gets stuck @ local minima & can't find global maxima / minima
↳ Sensitive to initial choice of sol^n, if initial sol^n is poor final sol^n also may be poor.
↳ Doesn't explore search space thoroughly, which limit is ability to find better sol^n.
↳ Back-tracking isn't permitted.

5/8/24 Best-first-Search / A-algorithm / Admissible algorithm.

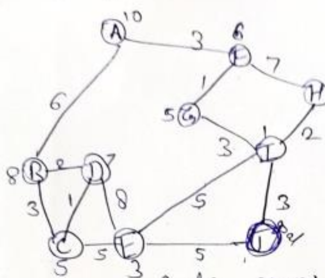
↳ Soln isn't optimal always.
↳ It always gives the soln for problem ~~if~~ if soln exits.
But the soln needn't neccesarily be optimal.

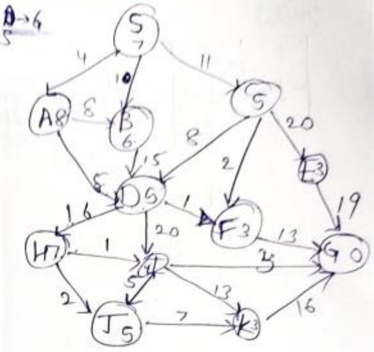
open – ~~E~~, D, F, ~~I~~, ~~J~~, ~~K~~, ~~L~~, ~~O~~, P
closed – A, B, C, E, G, H, ~~O~~ I, J, L

~~Open node~~ → node that isn't explored yet (successors aren't ~~seen~~ seen/added)
Closed: successors are explored & ~~nodes~~ successors are added to open node list

Best first & A*

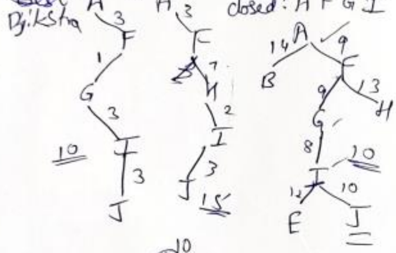
eg S→G



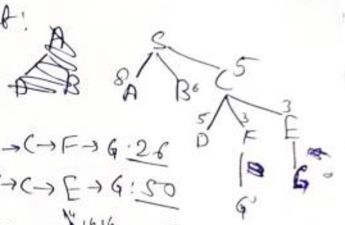


Best: open: B A B
closed: A F G I

Dijkstra



Best:

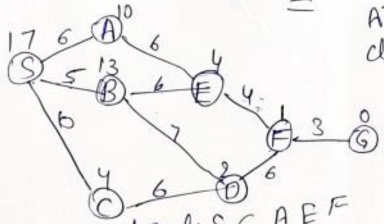


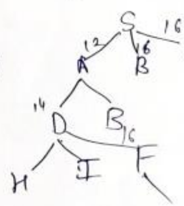
S→C→F→G : 26
S→C→E→G : 50

A*: open: B C D H I
closed: A D F

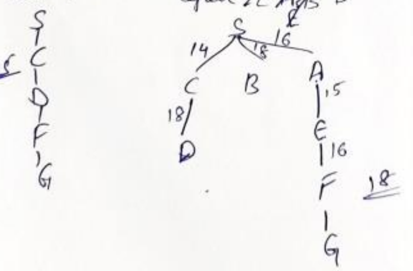
S-A-D

eg.





Best: A*: closed: S C A E F
open: C A B D



Path - S - C - D - A - E - F - G

CSP - Constraint Satisfaction Problem

0 & 1

definitely 1 as O is zero

$C_1\ C_2\ C_3$ 0 & 1

S E N D

\+ M O R E

M O N E Y

1 1

8 2 3 5 D

\+ 1 0 9 2

1 0 3 2 7 Y

$S = 9,\ C_1 = 0$

1 1

8 9

1 0

10 0

constraints

$M = 1$

$S + M >= 8$ & 9

$O = 0$

$E \neq N$

$N = E + 1$

$C_2 = 1$

$N + R \geq 11$

$S = 9,\ M = 1,$

$C_1 = 0,\ O = 0,$

$C_2 = 1,\ N + R > 11$

$N + R = E$ & $E+1$ $N = E + 1$

$R > 6$

$R = 7$

E = 4 1

9 4 5

1 0 7

1 0 5

E = 2

S + M = 8/9

N = 2

N + R ≥ 11

E = 3

N = 4

E = 4, E = 5, N = 6

N = 5

E = 6

N = 7

S >= 8

R > 8

R = 9

S = 7 & 8

$C_1\ C_2\ C_3\ C_4$

CROSS

\+ ROADS

DANGER

D = 1

S + 1 = E

2 + S = R

C + R ≥ 9

Means-Ends Analysis

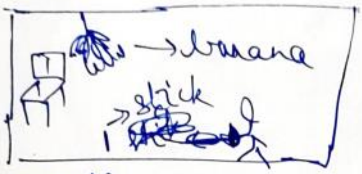
destinⁿ → move table & objects on it
↓ plane surface, no obstacle

obj 1 on floor
walk - pickup
putdown - obj 2 on
pickup - putdown floor
walk - pickup - table
carry - place
walk - pickup - obj 2
carry - palace.

Operator	Precondⁿ	Post Condⁿ
1) Push(obj, loc) there is nothing on top of it	large(obj) ∧ clear(obj) ∧ armempty(~~obj~~) ∧ at(robot, loc) robot shld be near the obj to push it.	at(obj, loc) ∧ at(robot, loc) ~~holding(obj)~~
2) PICKUP(obj1, obj2)	small(obj) ∧ clear(obj) ∧ at(robot, loc) ∧ armempty()	holding(obj1) ∧ clear(obj 2) no change in loc. req.
3) PUTDOWN(obj) on floor not table	holding(obj) → OR we can use — armempty() → arm not empty & (∧) small(obj) ⇒ this also means same as holding(obj).	armempty()
4) PLACE(obj1, obj 2)	large(obj 2) & holding(obj1) & at(robot, loc)	on(obj1, obj 2) ∧ armempty() → obj1 is on obj 2.
5) Walk(loc) emptyhanded walk - come back to source etc.	armempty()	At(robot, loc) locaⁿ is changed.
6) CARRY(obj)	holding(obj1)	at(robot, loc) → locaⁿ is changed.

Monkey banana Problem

Stick is place somewhere



~~operations~~ operator	Precondition	Post condition
1) findstick( )	nostick (!) or notempty ( ) stick	stick( ) ⟶ holding stick
2) walk(loc)	holding (~~stance~~) stick ( )	at(loc)
3) climb( )	holding ( )	holding ( ), at(loc) → chair
4) wave( )	climb( ) holding ( )	holding ( ), eat ( ), at(loc) → chair
5) eat( )		

Predicate logic

23/9/24

1) Spot is a dog.
dog(spot)

2) Spot has a tail
hastail (spot)

3) All dogs have tail
$\forall$-quantifier.
$\forall x : dog(x) \rightarrow hastail(x)$

4) Some dogs has tail
$\forall x : \exists y : dog(y) \rightarrow$ ~~had~~ $hastail(y)$

1) Marcus was a man
$\rightarrow$ man(marcus)

2) Marcus was a pompeian
$\rightarrow$ pompean(marcus)

3) All pompians were Romans
$\rightarrow \forall x : pompeians(x) \rightarrow Romans(x)$

4) Caeser was ruler
$\rightarrow$ Ruler(Caeser)

5) All Romans were either loyal to Caeser or hated him.
$\rightarrow \forall x : Roman(x) \rightarrow loyalto(x, caeser) \vee hate(x, caeser)$ OR

6) Everyone is loyal to someone
$\rightarrow \forall x ; \exists y : loyalto(x, y)$

7) People only try to assassinate rulers, they are not loyal to
$\rightarrow \forall x ; \forall y : person(x) \wedge ruler(y) \wedge tryassassinate(x, y) \rightarrow \neg loyalto(x, y)$ And negation

8) Marcus tried to assassinate Caeser.
$\rightarrow$ ~~try~~ Assassinate (markus, caeser)
markus assacinate caeser

9) Marcus was born in 40AD.
$\rightarrow$ born(Marcus, 40AD)

10) All men are mortal
$\rightarrow \forall x : men(x) \rightarrow mortal(x)$

11) All Pompeians died when the volcano errupted in 79 AD.
independent facts
$\rightarrow \forall x : Pompeians(x) \rightarrow died(x, 79AD)$
$\rightarrow$ erupted(Volano, 79 AD)

12) No mortal lives longer than 150 years.
$\rightarrow \forall x ; \forall t_1 ; \forall t_2 : mortal(x) \wedge born(x, t_1) \wedge gt(t_2 - t_1, 150) \rightarrow dead(x, t_2)$

13) It is now 2024.
$\rightarrow now = \underset{t_2 - t_1}{2024}$. greater than 150

14) If someone is dead he is dead at all later times.
$\rightarrow \forall x : \forall t_1 ; \forall t_2 ; dead(x, t_1) \wedge gt(t_2, t_1) \rightarrow dead(x, t_2)$

Prove $\neg$loyal to(Marcus, Caeser) - add - match 7 on RHS

15) All men are person.
$\forall x : man(x) \rightarrow person(x)$
thus ~~man~~ $\underset{(15)}{person(markus)} \wedge \underset{(4)}{ruler(caeser)} \wedge \underset{(8)}{tryassassinate(markus, caeser)}$
$\rightarrow \underset{(7)}{\neg loyalto(\text{~~}markus, caeser)}$

Hence proved.

Q. Is markus alive now? prove. Alive(Markus, now)

16) Alive means not dead.

17) Dead means not alive.

eg.

1) John likes all kinds of food.
→ ∀x : food(x) → ~~likes~~(John, x)
गलत → John(likes, x)

2) Apples are food.
→ food(apples)

3) Anything anyone eats & is not killed by is food.
→ ∀x : ∀y : eats(x, y) ∧ ¬ killed(x)
→ food(y)

4) Bill eats peanuts & is still alive.
→ Eats(Bill, peanuts) ∧ alive(Bill) ⇒ not one combined fact
→ Eats(Bill, peanuts) → Alive(Bill) ⇒ two independent facts

5) Merry eats everything Bill eats.
→ ∀x : Eats(bill, x) → Eats(Merry, x)
#4

eg. #
1) If x is on top of y, y supports x.
→ ∀x, ∀y : top(x, y) → supports(y, x)

2) If x is above y & they are touching each other, then x is on top of y.
→ above(x, y) ∧ touch(x, y) → top(x, y)

3) A cup is above a book.
→ above(cup, book)

4) A cup is touching a book.
→ touch(cup, book)
#5

#1
12) → dead(x) → ¬alive(x, now)

17) ⇒ dead(markus, now) (17)

↑

dead(markus, t1) (11) ∧ gt(now, t1)

↑

pompean(markus) (2) ∧ gt(now, 79)   t1 = 79

↑

gt(now, 79AD)

↑

gt(2024, 79AD)

↑

nil

OR we can prove as markus was born in 40AD & pompeans live for 150 yrs so now he's dead.

mortal(markus) ∧ born(markus, t1) ∧ gt(now − t1, 150) → died(m, now) (12)

↑

#2 man(markus) ∧ born(markus, t1) ∧ gt(now − t1, 150)

↑

born(markus, 40AD) (9) ∧ gt(now − t1, 150)

↑

gt(now − t1, 150)   t1 = 40

↑

gt(now − 40, 150)

↑

gt(2024 − 40 AD, 150)

#2 man(markus) (1) ⇒ man(markus) → ~~dead~~ mortal(markus) (10)
replace in (12) & match LHS to (12)

#3 apply (17)
dead(markus, now) → ¬alive(markus, now)
hence proved.

Prove: Does markus hate Caeser?

Can't prove unless we know
hate(x,y) → ¬ ~~the~~ loyalto(x, y)

#4 Prove: John likes peanuts.
Can't prove directly.
We need to prove 1st - peanut is food. Than can be done using (3) & (4)

add: Alive (Bill) → ¬ ~~not~~ killed(Bill)

then: Eats (Bill, peanuts) ∧ Alive (Bill)

↓

Eats (Bill, peanuts) ∧ ¬ killed (Bill)

↓ (3)

food(peanut)

↓ (1)

food(peanuts) → likes (John, peanuts)

#5 Prove: Supports (Book, cup)
top(~~Book~~, cup Book) → supports (Book, cup)

↑

above (cup, book) ∧ touch (cup, book)

eg.

1) Every bird sleeps in some tree.

→ ∀ (Bird ( ) → sleeps(tree)

$\forall x$: bird(x) $\land$ tree(y) → sleeps(x, y)

2) Every loon is a bird & every loon is aquatic.

→ $\forall x$: loon(x) → bird(x)

loon(x) → aquatic(x)

→ bird(loop) → aquatic(loop)

3) Every tree in which any aquatic bird sleeps is besides some lake.

→ $\forall x, \forall y$: aquaticbird(x) $\land$ sleeps(x, y) → besidelake(y)

4) Anything that sleeps in anything that is beside any lake, eats fish.

→ $\forall x$, $\forall y$: (bird, tree)

sleep(x, y) $\land$ beside-lake(y) → eats(x, fish)

1/10/24 Resolutⁿ

- → Proof by refutatⁿ
- → Conversion into CNF - (Conjuctive Normal Form)
- → Unificatⁿ (Combining statements to remove quantifiers)

★

① $A \to B = \neg A \lor B$
② move all quantifiers to LHS & drop the prefix.
③ Use standardize variables
④ distributive law
gt(2024, 40AD) - type can be removed directly in ∨ as it is true fact.

eg. $\underbrace{\quad}_{A} \to \underbrace{\quad}_{B}$

1) $\forall x : food(x) \to likes(John, x)$

$\neg food(x_1) \lor likes(John, x_1)$

2) food(apples) $\neg A$

3) $\forall x. \forall y : (eats(x, y) \land \neg killed(x)) \to food(y)$ B

$\neg A \to B \Rightarrow A \lor B$

4) a) eats(Bill, Peanuts)
b) Alive(Bill)

5) $\forall x$ eats(Bill, x_3) → eats(Mary, x_3) $\neg eats(Bill, x_3) \lor eats(Mary, x_3)$

To prove: likes(John, Peanuts)

Proof:-

x_1 = Peanuts in ①.

$\neg likes(John, peanuts)$ ① x_1 = peanuts

$\neg food(peanuts)$

③ y_1 = peanuts

$\neg eats(x_2, peanuts) \lor killed(x_2)$

4a x_2 = bill

killed(bill)

Alive means not killed & killed means not alive.

$\neg Alive(Bill)$

[ ]

eg.

1) $\forall x, \forall y : top(x,y) \to supports(y, x)$

$\neg top(x_1,y_1) \lor supports(y_1, x_1)$

2) $[above(x,y) \land touch(x,y)] \to top(x,y)$

$\neg above(x_2, y_2) \lor \neg touch(x_2, y_2) \lor top(x_2, y_2)$

3) above(cup, book)

4) touch(cup, book)

Prove: supports(Book, cup)

Proof:

$\neg supports(Book, cup)$ ① x_1 = cup, y_1 = book

$\neg top(cup, Book)$ ② x_2 = cup, y_2 = Book

$\neg above(cup, Book) \lor \neg touch(cup, Book)$

③, ④

supports(Book, cup)

eg. To prove: $\neg Alive(Markus, now)$

now = 2024

Solⁿ:

1> men (markus)

2> pompeian (markus)

3) $\forall x$: pompeian(x) ~~→~~ Romans(x)

$\neg$ pompiean $(x_1) \vee$ Roman (x_1)

4) Ruler (Caeser)

5> $\forall x$: Roman(x) → loyalto(x, Caeser) $\vee$ hate (x, Caeser)

Sol:

$\neg$ Roman$(x_2) \vee$ loyal to $(x_2$, Caeser) $\vee$ hate $(x_2$, Caeser).

6> $\forall x \forall y$: loyalto (x, y)

7> $\forall x, \forall y$: prison (x) $\wedge$ ruler (y) & (and try assassinate (x, y)

8) Assasinate (markus, Ceaser)

9> Born (markus, 40 AD)

10) All men are mortal.

→ $\forall x$: men (x) → mortal(x)

$\neg$ men (x) $\vee$ mortal (x)

11> $\forall x$: Pompeians (x) → died (x, 79 AD)

$\neg$ Pompieans (x) $\vee$ died (x, 79 AD)

12>

12) $\forall x : \forall t_1 \forall t_2$:

$\neg$(mortal (x) $\wedge$ born (x_2, t) $\wedge$ gt $(t_2 - t_1, 150)$ → dead (x, t_2)

12> ⇒ $\neg$ mortal $(x_1) \vee \neg$ born(x, t_1) $\vee \neg$gt $(t_2 - t_1)$ 4150 $\vee$ dead (x_2, t_2)

$\forall x, \forall t_1 : \forall t_2$:

dead $(x, t_1) \wedge$ gt (t_2, t_1) → dead

Ans. 13 $\neg$dead $(x_2, t_1) \vee$ Gt $(t_2, t_1) \vee$ dead (x, t_2)

14/10/24 Proove using resolution - every loon eats fish

Proof: ~~¬eats(fish, loon)~~

¬eats(loon, fish)

1> $\forall x : \exists y$ sleeps(x, y)

CNF:

~~2> $\forall x$: bird(x) → aquatic(x~~

2> bird(loon)

aquatic (loon)

CNF: same

3> $\forall x$: $\forall y$ sleeps(y, x) → besides(y)

tree aqua

CNF: ¬ sleeps(y_1, x_1) ∨ beside (y_1)

4> $\forall x$: $\forall y$: sleeps(x, y) ∧ besides(y) → eats(x, fish)

CNF: ¬sleeps (x_2, y_2) ∨ ¬ beside(y_2) ∨ eats(x_2, fish)

¬eats (loon, fish)

(4) x_2 = loon

14/10/24

1) Every women that can be burnt is witch. → $\forall x$: women(x) $\wedge$ burnt(x) → witch(x)

CNF: $\neg$woman(x_1) $\vee$ $\neg$burn(x_1) $\vee$ with(x_1)

2) Everything that is made of wood can be burnt → $\forall x$: madeofwood(x) → burnt(x)

CNF: $\neg$madeofwood(x_2) $\vee$ burnt(x_2)

3) Everything that floats is made of wood. → $\forall x$: floats(x) → madeofwood(x)

CNF: $\neg$floats(x_3) $\vee$ madeofwood(x_3)

4) Everything that weighs the same as something that floats, does float too. → $\forall x, y$: weighsame(x, y) $\wedge$ floats(x) → floats(y)

CNF: $\neg$weighsame(x_4, y_1) $\vee$ $\neg$floats(x_4) $\vee$ floats(y_1)

5) This girl is a woman → woman(girl)

6) This girl weights the same as this tub. → weighsame(girl, tub) = weighsame(tub, gir)

7) This tub floats. tub तरे duck → floats(tub)

Is the girl witch - witch(girl)?

- Prove using resolutn → must 1 ques in exam

proof: $\neg$ witch (girl)

① x_1/girl

② $\neg$woman(girl) $\vee$ $\neg$burn(girl)

x_2/girl

⑤ $\neg$madeofwood(girl) $\vee$ ~~burn~~ woman(girl)

$\neg$madeofwood(girl)

③ x_3/girl

y_1 = tub ④ x_4 = girl

$\neg$float(girl)

$\neg$weighsame(girl, tub)

Proof using p (regular)

6 & 7 से 4 → floats (girl)

4 से 3 → madeofwood (girl)

3 → 2 → burn (girl) } combine get

5 → woman (girl) }

∴ woman(girl) $\wedge$ burn(girl) → witch(girl)

hence proved.