INDRAPRASTHA INSTITUTE of INFORMATION TECHNOLOGY DELHI

CSE643 – Artificial Intelligence

Monsoon 2020 session

End-sem exam

Max marks: 50 (will be scaled down to 25 marks) **09-Dec-2020**

Submission deadline: 09-Dec-20 at 19:00 hrs

INSTRUCTIONS:

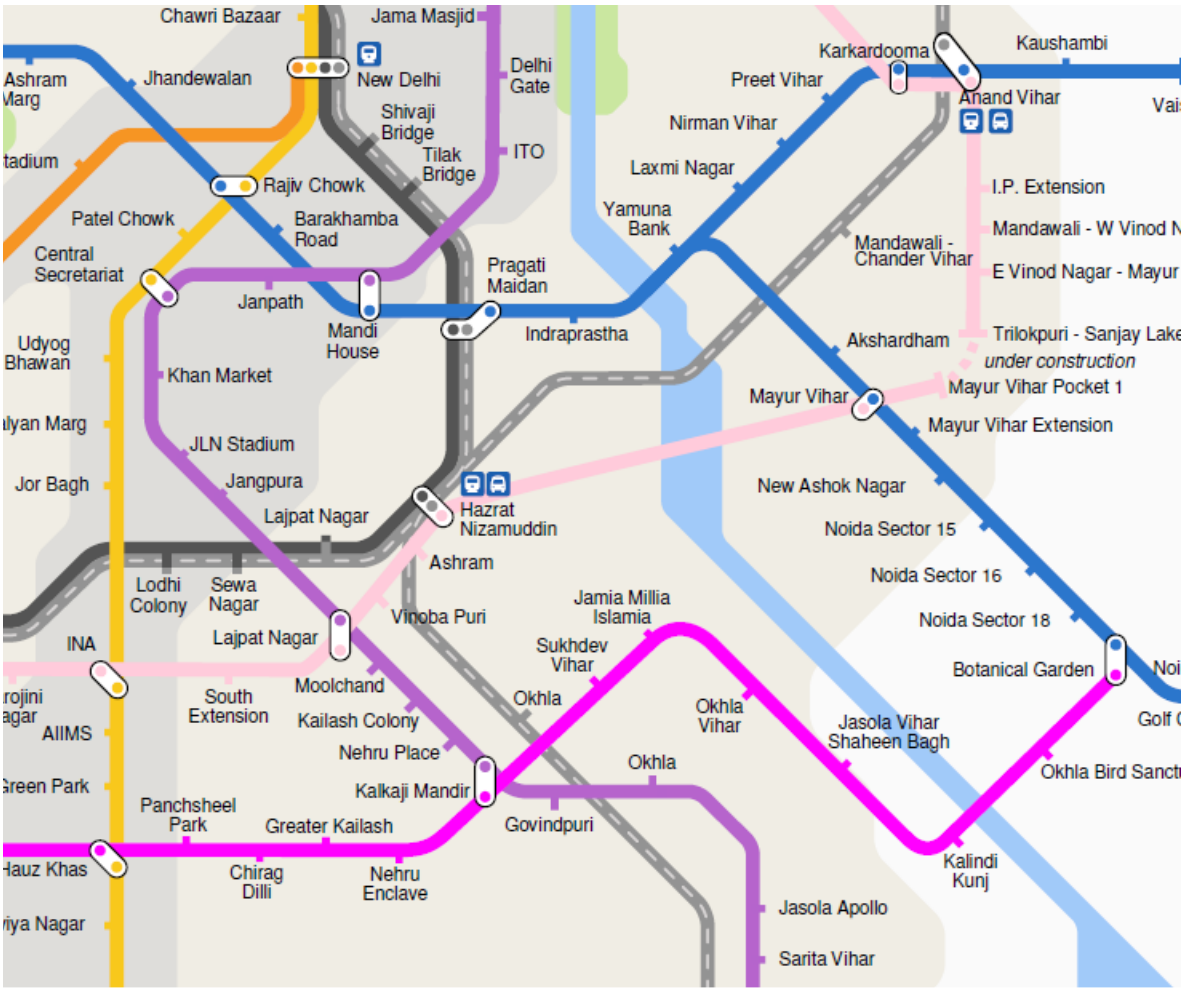
You will have to create a PDF file with your answers, name the file as AI-EndSem-<Name>-<RollNo> and upload it on the classroom page by 7:00 pm.

In the answer sheet write or type your name and roll number.

In case you choose to have hand-written answers then those pages can be scanned and uploaded (make sure that it is clearly readable). Make sure it has the name and roll number on it.

Q1: **(4 marks)**

You are building a system to tell users how to travel between two locations on Delhi Metro as quickly and as cheaply as possible for the metro map given below. Assume that Delhi Metro has a flat cost of Rs 20/- for any metro train ride from starting station to ending station irrespective of where you get off. If you change to an interconnecting train then you have to pay Rs 20/- again. Formulate this as a A* search problem of going from any given station to another station. Assume some heuristic and show that it is admissible and consistent.



Answers

We define the cost g(n) of getting to a node n as g(n) = 20 * Metrosboarded + NumSt, where Metrosboarded is the number of metro trains boarded so far, and NumSt is number of stations from Start station to get to n.

We define our heuristic function h(n) as: h(n) = Stns(n) + 20*Int(n), where Stns(n) is the minimum number of stations between any given station 'n' and the destination (goal) station. And Int(n) is the minimum number of interchange metro trains that need to be boarded to get to the destination (goal) station – if the given station and goal station are on the same line then Int(n) = 0 else it is the minimum number of trains to board to get to goal station.

A heuristic h(n) is admissible if, for every node n, it underestimates the cost to the goal, that is $h(n) \leq h^*(n)$; basically a lower bound on h*(n), where h*(n) is the cost of the optimal path to the goal. Since our heuristic is taking Stns(n) as the minimum number of stations between station 'n' and destination(goal) station, it will always be a lower bound.

A heuristic h(n) is consistent if, for every node n and every successor n' of n, the estimated cost of reaching the goal from n is no greater that the step cost of getting to n' plus the estimated cost of reaching the goal from n', i.e. $h(n) \leq c(n, n') + h(n')$. Since our heuristic is taking both Stns(n) and Int(n) as minimum number of stations and minimum number of interchange metro trains that need to be boarded between start and goal, the heuristic is also consistent.

Initial State: Start station

Actions: Calculate h(n) from Stns(n) and Int(n). Calculate f(n) = g(n) + h(n), where g(n) = 20 * Metrosboarded + NumSt, where Metrosboarded is the number of metro trains boarded so far, and NumSt is number of stations from Start station to get to n. Expand the minimum f(n) node.

State space: Stations travelled and Metro trains boarded.

Path: From Start to Destination station.

Goal test: Does CLOSED have Destination station or not. If yes Halt else expand node from OPEN.

Q2: **(4 marks)**

Represent the knowledge given below in FOPL and convert to CNF and then using resolution refutation (draw the graph) determine whether Macchli is frustrated or not.

Every tiger chases some deer. Every deer that jumps and runs is smart. No tiger catches any smart deer. Any tiger that chases some deer but does not catch it is frustrated. If all deers are smart then all tigers are frustrated. Macchli is a tiger. Josh is a deer and jumps and runs.

Answer

1. $\forall x \exists y$ (tiger(x) $\wedge$ deer(y) $\rightarrow$ chase(x,y))
2. $\forall x$ deer(x) $\wedge$ jumps(x) $\wedge$ runs(x) $\rightarrow$ smart(x)
3. $\forall x \forall y$ (tiger(x) $\wedge$ deer(y) $\wedge$ smart(y) $\rightarrow$ $\neg$catch(x,y))
4. $\forall x$ (tiger(x) $\wedge$ $\exists y$ (deer(y) $\wedge$ chase(x,y) $\wedge$ $\neg$catch(x,y)) $\rightarrow$ frustrated(x))
5. $\forall x \forall y$ (tiger(x) $\wedge$ deer(y) $\wedge$ smart(y) $\rightarrow$ frustrated(x))
6. tiger(Macchli)
7. deer(Josh)
8. jumps(Josh)
9. runs(Josh)

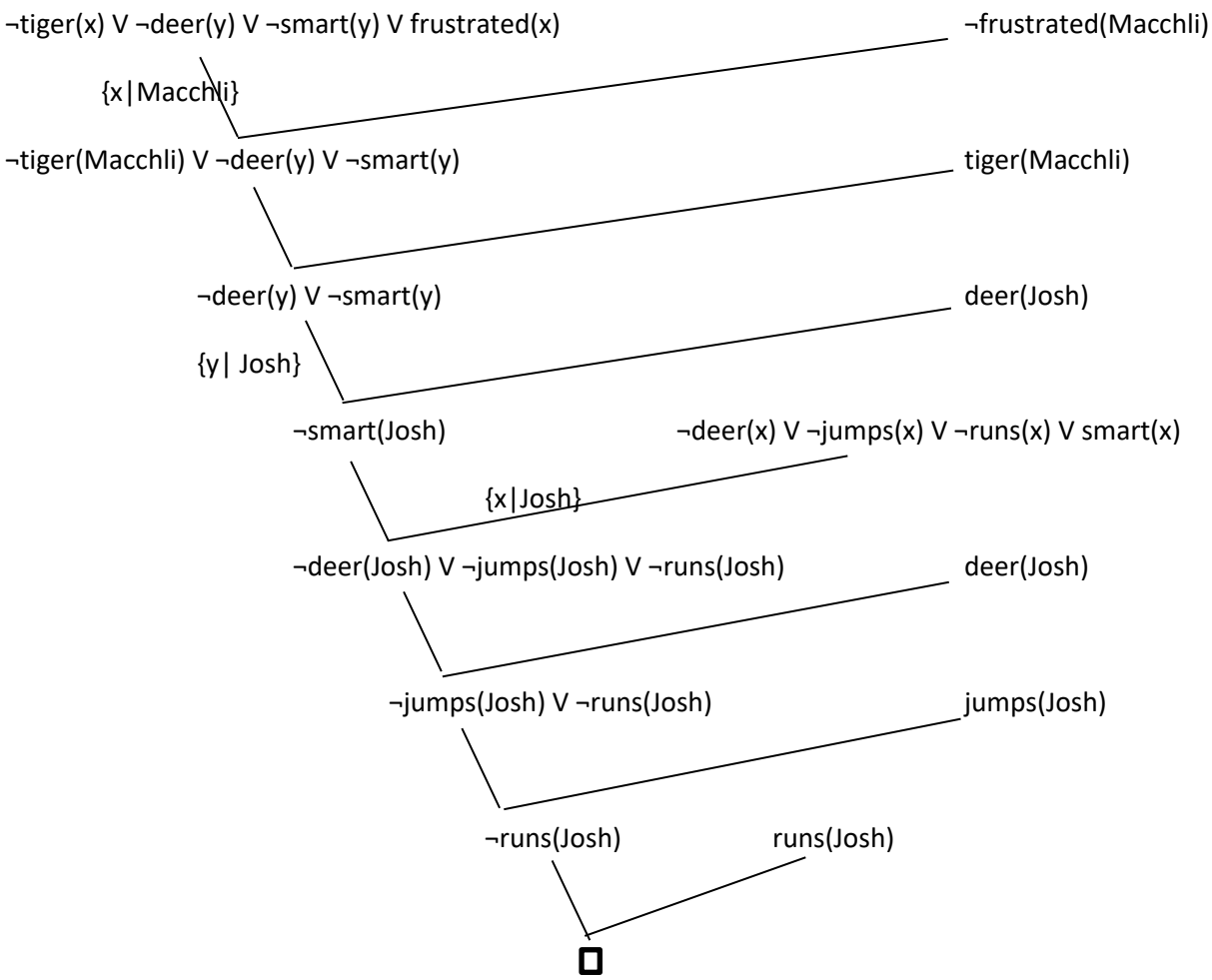
In CNF

10. $\forall x$ (tiger(x) $\wedge$ deer(C) $\rightarrow$ chase(x,C)) – introducing skolem constant C for existential quantifier
11. $\neg$tiger(x) $\vee$ $\neg$deer(C) $\vee$ chase(x,C) – (10) in clause form
12. $\neg$deer(x) $\vee$ $\neg$jumps(x) $\vee$ $\neg$runs(x) $\vee$ smart(x)

13. ¬tiger(x) V ¬deer(y) V ¬smart(y) V ¬catch(x,y)
14. ∀x (tiger(x) Λ deer(D) Λ chase(x,D) Λ ¬catch(x,D) → frustrated(x)) – introducing skolem constant
15. ¬tiger(x) V ¬deer(D) V ¬chase(x,D) V catch(x,D) V frustrated(x) – (14) in clause form
16. ¬tiger(x) V ¬deer(y) V ¬smart(y) V frustrated(x)

17. Hypothesis is frustrated(Macchli). Assume negation of hypothesis, that is ¬frustrated(Macchli). Add it to clauses.
18. From 16 and 17 we get ¬tiger(Macchli) V ¬deer(y) V ¬smart(y) with unification of {x|Macchli} and resolution
19. From 6 and 18 we get ¬deer(y) V ¬smart(y) – from resolution
20. From 7 and 19 we get ¬smart(Josh) -- from unification {y| Josh} and resolution
21. From 12 and 20 we get ¬deer(Josh) V ¬jumps(Josh) V ¬runs(Josh) -- from unification {x|Josh} and resolution
22. Using facts 7, 8, 9 and 21 we get a contradiction, thus the negation of the hypothesis is FALSE and so the hypothesis frustrated(Macchli) is TRUE.

Resolution Graph



We get Contradiction. So the hypothesis frustrated(Macchli) is TRUE.

Q3: **(6 marks)**

Represent the following IIITD academic course knowledge as rules. Explain where forward-chaining and backward-chaining will occur.

M.Tech (CSE) may be done with a thesis, or without a thesis but with a scholarly paper. In both options, students have to do certain amount of coursework. In addition, students doing M.Tech with thesis will have to do a thesis. Students in without thesis option have to do additional courses, and instead of a thesis will have to do a scholarly paper.

A student doing M.Tech with scholarly paper has multiple options for completing the scholarly paper requirement: a regular scholarly paper, industrial project and capstone project.

The overall requirements for M.Tech are as follows:

M.Tech with thesis: 32 credits of coursework + 16 credits of thesis. At most 4 credits may be earned by doing 300 and 400 level courses.

M.Tech without thesis: 40 or 44 credits of coursework + 8 or 4 credits for a scholarly paper. At most 8 credits may be earned through doing 300 and 400 level courses.

For the thesis or the scholarly paper credits, though the student has to register, he/ she need not be physically present and can do the work while being outside the Institute.

A student admitted to the M.Tech program will give his/ her choice regarding which of the two options he/she wants to pursue. However, this choice can be changed at any time during the program by suitably informing the PG Committee.

Within the course work requirement, each M.Tech(CSE) student has to earn 12 credits from core courses. The core courses comprises of one course (of 4 credits) each from the following groups (additional courses may be added later to these sets by consent of the faculty).

Theory bucket	Systems bucket	Software bucket
Modern Algorithm Design (CSE519)	Computer architecture (CSE511)	Program Analysis (CSE503)
Randomised algorithms (CSE523)	Mobile computing (CSE535)	Information Retrieval (CSE508)
Graduate Algorithms (CSE525)	Wireless Networks (CSE538)	Compiler (CSE601)

All other courses are electives. In electives, at most 4 credits of Independent Study and 4 credits of Minor Project can be taken.

Online course may be permitted to be registered as Independent study/minor project with permission of PGC.

Answers

Rule 1: If M.Tech thesis done then M.Tech(CSE) done.

Rule 2: If M.Tech without thesis done then M.Tech(CSE) done.

Rule 3: If CourseCreditsEarned ≥ 32 and ThesisCredits earned = 16 then M.Tech thesis done

Rule 4: If CourseCreditsEarned ≥ 40 and ScholarlyPaperDone and ScholarlyPaperCredits = 8 then M.Tech without thesis done

Rule 5: If CourseCreditsEarned ≥ 44 and ScholarlyPaperDone and ScholarlyPaperCredits = 4 then M.Tech without thesis done

Rule 6: If Regular scholarly paper done then ScholarlyPaperDone

Rule 7: If Industrial project done then ScholarlyPaperDone

Rule 8: If Capstone project done then ScholarlyPaperDone

Rule 9: If opted for M.Tech thesis then Max300400level = 4

Rule 10: If opted for M.Tech without thesis then Max300400level = 8

Rule 11: If opted for M.Tech thesis and doing Thesis then Register and NeedNotBePresentAtInstitute

Rule 12: If opted for M.Tech without thesis and doing ScholarlyPaper then Register and NeedNotBePhysicallyPresentAtInstitute and CanDoWorkOutsideInstitute

Rule 13: If StudentAdmittedToMTech then StudentSelectsThesisOrNoThesisOption

Rule 14: If StudentSelectsThesisOrNoThesisOption and WantsToChangeOption then InformPGCommittee.

Rule 15: If M.Tech(CSE) then StudentHasToEarnCoreCourseCredits = 12

Rule 16: If StudentHasToEarnCoreCourseCredits = 12 then One4CreditCourseFromTheoryBucket and One4CreditFromSystemsBucket and One4CreditFromSoftwareBucket

Rule 17: If CourseCreditsFromElectives then MaxIndependentStudyCredits =4 and MaxMinorProjectCredits = 4

Rule 18: If StudentEarnedCoreCredits and OtherElectiveCreditsEarned then CourseCreditsEarned

Rule 19: If OnlineCourse and PGC approval then IndependentStudyCredit

Rule 20: If OnlineCourse and PGC approval then MinorProjectCredit

Backward-chaining rules: Rule 1 and Rule 2 backward-chains to Rules 3, 4, 5. Rules 4 and 5 backward-chains to Rules 6,7, 8. Rule 15 backward-chains to Rule 16.

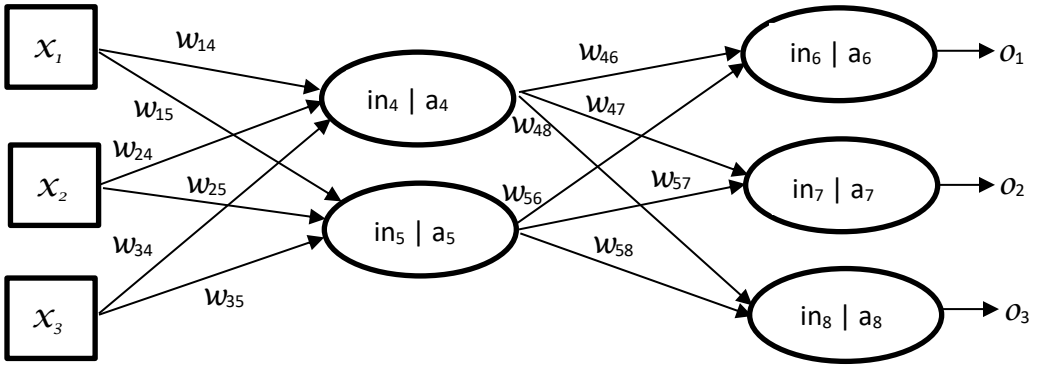
Forward-chaining rules: Rules 9, 10, 11, 12, 13, 14 forward-chains based on inputs.

Rules 17 to 20 can be forward-chaining or backward-chaining depending on the context.

Q4: **(4 marks)**

a) Suppose we have a dataset of n examples and in a Univariate regression we hypothesize the learning function as $h_w(x) = \ln(w^2 x_i)$. Assuming that the true output is represented by y_i, derive the gradient descent update for w. (Note: d/dx of ln(x) = 1/x)

b) In the Neural network given below all neurons have bias = 0 and the sigmoid activation function.



After training, the units of the network have the following weights were determined: w_{14} = -2; w_{15} = 1; w_{24} = 2; w_{25} = 1; w_{34} = -2; w_{35} = -1; w_{46} = 1; w_{47} = 0.5; w_{48} = 0.3; w_{56} = -3.5; w_{57} = -1.2; w_{58} = 0.6 .

i) If the input given is x_1 = 2; x_2 = 3; x_3 = 1; calculate the output of the hidden neurons and the output neurons.

ii) Further, if the actual outputs are o_1 = 0.83; o_2 = 0.26; o_3 = 0.56; while the target outputs are t_1 = 0.58; t_2 = 0.70; t_3 = 0.20. Calculate the error in the output neurons and the hidden neurons.

Answers

a) The L2 loss function is defined as: $\text{Loss}(h_w) = \sum_{i=1}^{n}\left(y_i - h_w(x_i)\right)^2$. Thus: $\text{Loss}(h_w) = \sum_{i=1}^{n}\left(y_i - ln(w^2 x_i)\right)^2$. The gradient descent of the of the weight w would be:

$w = w - \alpha \frac{\delta Loss(h_w)}{\delta w}$. That is $w = w - \alpha \frac{\delta}{\delta w}\left(\sum_{i=1}^{n}\left(y_i - ln(w^2 x_i)\right)^2\right)$ where α is the learning rate.

Using the summation rule of differentiation,

$\frac{\delta}{\delta w}\left(\sum_{i=1}^{n}\left(y_i - ln(w^2 x_i)\right)^2\right) \equiv \sum_{i=1}^{n}\frac{\delta}{\delta w}\left(y_i - ln(w^2 x_i)\right)^2)$

Using the chain rule of differentiation,

$\equiv \sum_{i=1}^{n} 2 * \left(y_i - ln(w^2 x_i)\right) * \frac{-2wx_i}{w^2 x_i} \equiv \sum_{i=1}^{n} 2 * \left(y_i - ln(w^2 x_i)\right) * \frac{-2}{w} \equiv \sum_{i=1}^{n}\frac{-4}{w} * \left(y_i - ln(w^2 x_i)\right)$

Thus, the gradient descent update for w is $w \leftarrow w - \alpha(\sum_{i=1}^{n}\frac{-4}{w} * \left(y_i - ln(w^2 x_i)\right))$ where α is the step size.

b)

i)

in_4= -2*2 + 2*3 + -2*1 = 0. Thus sigmoid(in_4) = a_4 = 0.5

in_5= 1*2 + 1*3 + -1*1 = 4. Thus sigmoid(in_5) = a_5= 0.982

in_6= 1*0.5 + -3.5*0.982 = -2.937. Thus sigmoid(in_6) = o_1= 0.050

in_7= 0.5*0.5 + -1.2*0.982 = -0.926. Thus sigmoid(in_7) = o_2= 0.284

in_8= 0.3*0.5 + 0.6*0.982 = 0.739. Thus sigmoid(in_8) = o_3= 0.677

ii)

Given o_1 = 0.83; o_2 = 0.26; o_3 = 0.56; and target outputs t_1 = 0.58; t_2 = 0.70; t_3 = 0.20 then error is calculated as:

loss(o_1) = $(0.58-0.83)^2$ = 0.0625; loss(o_2) = $(0.70-0.26)^2$ = 0.1936; loss(o_3) = $(0.20-0.56)^2$ = 0.1296 and the error is calculated via gradients of the loss function where:

$\delta_j = f'(H_j) \Sigma_k \delta_k w_{kj}$ and $f'(H_j) = f(H_j)(1 - f(H_j))$ where $f'(H_j)$ is the derivative of the activation function $f(H_j)$. Thus we get

For output nodes we get δ_{o1} = 0.1411*(0.58-0.83) = -0.0352. Similarly we get δ_{o2} = 0.0847 and δ_{o3} = -0.0887.

For hidden nodes we get:

δ_{a4} = 0.25 *((-0.0352*1) + (0.0847*0.5) + (-0.0887 * 0.3)) = -0.01946

δ_{a5} = 0.0176 * ((-3.5 * -0.0352) + (-1.2 * 0.0847) + (0.6 * -0.00887) = -0.0005572

Q5: **(4 marks)**

Describe the objective of Inductive logic programming with dataset characteristic, goal of ILP and stopping criterion for the induction process. Use an example to elucidate.

Answer

Objective of Inductive Logic Programming is to learn a generalized hypothesis from given background knowledge and a set of examples, in such a way that the generalized hypothesis entails all the positive examples and none of the negative examples.

Dataset characteristic

1. B is the background knowledge that is provided and consists of logical clauses (FOPL). These are the seed facts that are provided for the learning process.
2. E^+ is the set of positive examples from which the hypothesis should learn to generalize.
3. E^- is the set of negative examples which the hypothesis should learn not to cover in its generalization.
4. $B \wedge E^- \nvDash \square$ E^- are the negative examples and these examples should not derive a contradiction with the background knowledge.
5. $B \nvDash E^+$ Background knowledge should not already subsume the positive examples

Goal of ILP

1. On the basis of B , E^+ and E^- learn a generalized hypothesis $H = \{C_1 \vee C_2 \vee \dots \vee C_n\}$ to classify positive and negative examples correctly in such a way that it generalizes and works correctly on the test data.
2. Start with empty H.
3. Take the most constrained positive example in E^+ and create clause C_x with a new literal instead of a constant in that example, such that C_x covers that positive example with variables in place of constants.
4. Test if C_x covers a negative example, if yes drop and go to Step 3 until all positive examples are covered.
5. If C_x does not cover a negative example, add to H. Go to Step 3 until all positive examples are covered.

Stopping criterion

1. $B \wedge H \models E^+$ E^+ are the positive examples and the hypothesis H learnt together with the background knowledge B should logically imply all positive examples.
2. $B \wedge H \wedge E^- \nvDash \square$ E^- are the negative examples and the hypothesis H learnt together with the background knowledge B should not derive a contradiction.

Example

Given B: father(Ram, Vinay), father(Vinay, Shyam), father(Ram, Rajiv), father(Rahul, Hari)

And E^+ : grandfather(Ram, Shyam)

And E^- : grandfather(Ram, Hari)

H is learnt as: grandfather(X,Y) :- father(X,Z), father(Z, Y).

Q6: **(4 marks)**

Describe at a high level how Meta-learning is implemented and the three types of Meta-learning techniques that can used. Answer based on what has been taught in the course.

Answer

Meta-learning is learning to learn a function given input and output datasets. Basically, it is a process of improving a learning algorithm over multiple learning episodes.

A conventional ML improves model predictions over multiple data instances. During base learning, a learning algorithm learns the task to classify data instances given in a dataset with respect to an objective. That is given D = $\{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$ it learns a function f_w to predict y_i given x_i, that is we try to *min loss(D | f_w)*.

During meta-learning, we are given many such D_i datasets and the function f_{wi} it has learnt together with the weights w_i. Meta-learning tries to learn from a training set (D_i , f_{wi} | w_i) to find optimal weights such that loss(D_{test} | w_j) is minimal.

Types of Meta-learning

1. Metric-based meta learning: In this type of meta-learning we try to minimize the distance between two dataset samples. A Siamese network is used for this learning to detect the similarity between two datasets. Based on the similarity the weights can be learnt.
2. Optimization-based meta learning: We try to find the best possible weights by choosing n examples at random, do some learning using D_{train} to update w, and then compute loss on D_{test} and then change the initial weights w_0.
3. Model-based meta learning: We try to model it as a sequence of dataset points, $\{D_1,\ldots,D_n\}$ and its corresponding weights, we try to learn weights for a new dataset point D_{n+1}. A Recurrent Neural Net can be used for this kind of meta learning.

Q7: **(4 marks)**

i) What are the two major approaches of Natural Language Processing? Explain the advantages and disadvantages of both these ways.

ii) In the NL sentence “Raju saw a dog yesterday which was an Alsatian dog”, identify parts of the sentence that are dependent on other parts of the sentence. Can a feed-forward neural network be built to process this sentence? Explain your answer.

Answers

i) First approach is Classical Parsing-based approach, and the Second approach is the Machine Learning/ Deep Learning based approach. In the Parsing-based approach the advantage is that we are able to parse the syntactic structure of the sentences and do semantic processing. Due to this we are able to identify parts-of-speech and chunks and use that to meaningfully analyze the input sentences and return meaningful answers to the user. The disadvantage is that since natural language has tremendous variations, the parsing-based approach can be very brittle when people express their sentences in various ways. In the Machine learning/ Deep learning-based approach we create a big corpus of natural language text and the kind of output expected from it and learn the mapping from input to output. In this way we ‘sort-of’ learn the natural language processing that is to be done. The advantage is that this approach we do not have to extensively code the syntactic and semantic rules as the system is able to learn the variations in the input sentence from the corpus and predict the possible outcome. Thus, the system is less brittle as compared to the parsing-based approach. However, the disadvantage is that the system requires a huge corpus which may be difficult to build and at times misses out the semantic context of the input sentences.

ii) In the sentence “Raju saw a dog yesterday which was an Alsatian dog”, the word ‘yesterday’ is dependent on a notion of time and the word ‘which was’ is dependent on the context of ‘dog’ and clarifies the object that Raju saw. In a feed-forward neural network there is no concept of ‘recurrence’ or notion of time and thus this sentence cannot be processed by a feed-forward neural network.

Q8: **(6 marks)**

An intelligent robot needs to devise a plan to achieve the goal of having its socks, shirt, pants, shoes and coat on, starting from an initial, empty state. The pants can be put on only after the shirt is put on, the socks can be put on only after the pants has been put on, the shoes can be put on only after the socks are put on, and the coat can be put on only after the shoes. Devise a plan using Action, Precondition, Add and Delete effects. Assume that socks, shirt, pants, shoes, coat are available. Draw the plan graph. Note: There is the left side socks and right side socks, similarly there is a left shoe and a right shoe.

Answers

Action: Put-Shirt-On

Precondition: Nothing-on ^ Available(Shirt)

Add: Has-Shirt-On

Delete: Nothing-on ^ Available(Shirt)

Action: Put-Pants-On

Precondition: Has-Shirt-On ^ Available(Pants)

Add: Has-Pants-On

Delete: Available(Pants)

Action: Put-Socks-On(LeftSide)

Precondition: Has-Shirt-On ^ Has-Pants-On ^ Available(LeftSideSocks)

Add: Has-Socks-On(LeftSide)

Delete: Available(LeftSideSocks)

Action: Put-Socks-On(RightSide)

Precondition: Has-Shirt-On ^ Has-Pants-On ^ Available(RightSideSocks)

Add: Has-Socks-On(RightSide)

Delete: Available(RightSideSocks)

Action: Put-Shoes-On(LeftSide)

Precondition: Has-Shirt-On ^ Has-Pants-On ^ Has-Socks-On(LeftSide) ^ Available(LeftSideShoes)

Add: Has-Shoes-On(LeftSide)

Delete: Available(LeftSideShoes)

Action: Put-Shoes-On(RightSide)

Precondition: Has-Shirt-On ^ Has-Pants-On^Has-Socks-On(RightSide)

^ Available(RightSideShoes)

Add: Has-Shoes-On(RightSide)

Delete: Available(RightSideShoes)

Action: Put-Coat-On

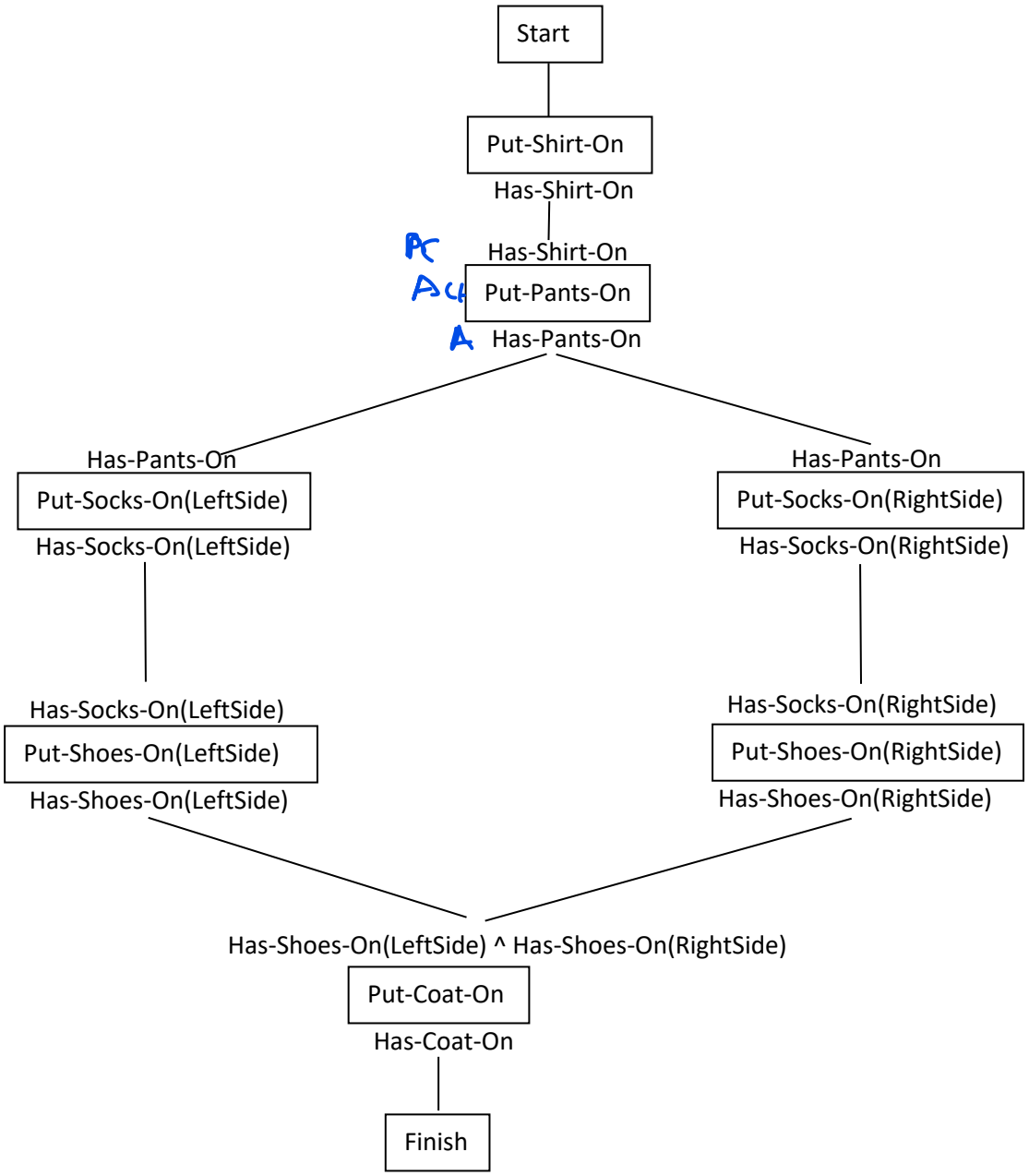
Precondition: Has-Shirt-On ^ Has-Pants-On ^ Has-Socks-On(RightSide) ^ Has-Shoes-On(LeftSide)

^ Has-Shoes-On(RightSide) ^ Available(Coat)

Add: Has-Coat-On

Delete: Available(Coat)

Plan Graph



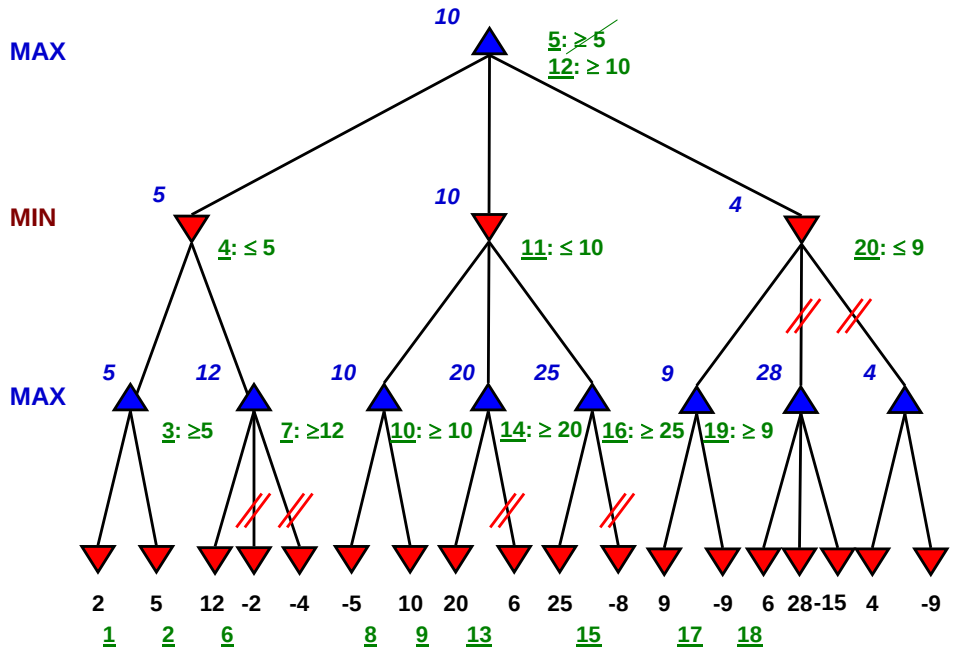
Q9: **(4 marks)**

Ram and Vinay decide to play a game with Ram starting first. Ram realizes that he has 3 possible options called {(a), (b), (c)}. If Ram chooses move (a) then Vinay has 2 possible options {(d), (e)}. If Ram chooses (b) or (c) then Vinay has three options {(f), (g), (h)} or {(i), (j), (k)} respectively. If Vinay chooses (d) then Ram has two options {(l) with gain value 2, (m) with gain value 5}. If Vinay chooses (e) then Ram has three options {(n) with gain value 12, (o) with loss value 2, (p) with loss value 4}. If Vinay choses (f) then Ram has two options {(q) with loss 5, (r) with gain 10}. If Vinay choses (g) then Ram has two options {(s) with gain 20, (t) with gain 6}. If Vinay choses (h) then Ram has two options {(u) with gain 25, (v) with loss 8}. If Vinay choses (i) then Ram has two choices {(w) with gain 9, (x) with loss 9}. If Vinay choses (j) then Ram has three choices {(y) with gain 6, (z) with gain 28, (z1) with loss 15}. If Vinay choses (k) then Ram has two choices {(z2) with gain 4, (z3) with loss 9}.

i) Draw a game tree and show the minmax evaluation on that tree.
ii) Do alpha-beta pruning and show the result of the game tree.

Answer

Let Ram's choices be represented by Blue upright triangles (Max nodes) and Vinay's choices be represented by Red inverted triangles (Min nodes).



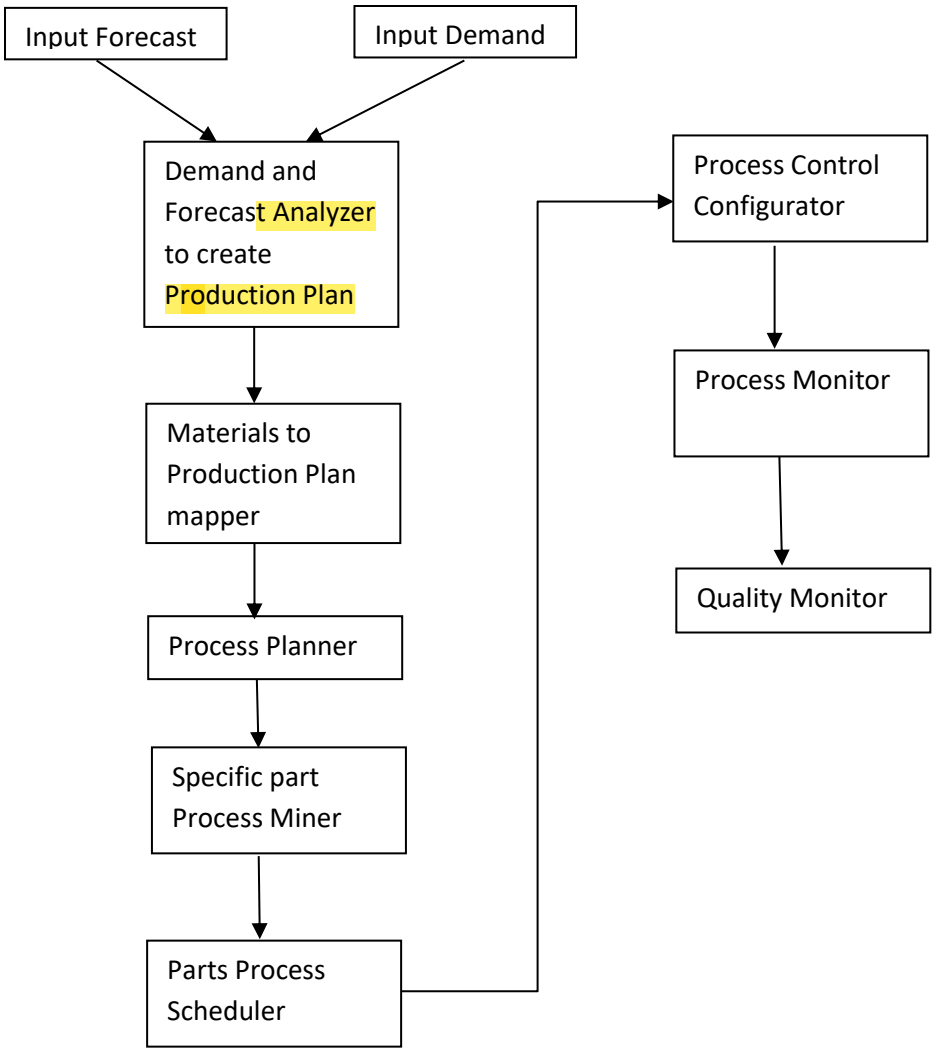
i) We can see that Ram will choose option (b) as that has the Max value of 10 for his play.
ii) The Alpha-beta pruning has pruned out nodes {o, p, t, v, j, k}.

Q10: **(10 marks)**

Lucas TVS manufactures vehicle parts, like alternators, starter motors, wiper motors, compressor motors, engine cooling systems, blower motors, filters, headlamp, horn, bulbs etc. With same raw material one or more parts can be manufactured depending on the demand and forecast. Suppose you have been asked to design an Intelligent Production scheduling and monitoring system for Lucas TVS, explain where all will you use Search, Logic, Knowledge representation (explicit and implicit), Reasoning (explicit and implicit), Planning, and Machine Learning in such a system? Draw an AI architecture of the system and explain each module with one example.

Answers

Vehicle Parts: {Alternators, Starter motors, Wiper motors, Compressor motors, Engine cooling systems, Blower motors, Filters, Headlamp, Horn, Bulb}. We create the architecture given below.



Description of each Module

1. Demand and Forecast Analyzer to create Production Plan
   This module analyzes Input Demand and Input Forecast to create a Production Plan. Since these inputs are based on various market forces, the demand and forecast will be estimates with some probability / score. From such estimates we try to construct the production plan of the various parts. Each part can also vary based on size, shape and type that is required. It may so happen that some parts are always manufactured together, for example, headlamps and bulbs may need to be manufactured together. Since it will be good to use prior history of demand-forecast to production plan, we use *machine learning* models to find the best possible production plan that can possibly meet the demand/ forecast.

2. Materials to Production Plan mapper
   Once the Production Plan is determined, we use Search and Logic with explicit rules to decide which materials are required for each part in the Production Plan. For example, we may have rules like
   if Headlamp for Tata Trucks then Bulb = 90Watt and material-required = aluminum
   if Headlamp for Bajaj Pulsar then Bulb = 35Watt and material-required = plastic
   if Headlamp for Volvo Buses then Bulb = 70Watt and material-required = steel
   etc.
   This helps plan the production process by determining if the materials are available.

3. Process Planner
   This module creates the plan to produce the parts based on the previous module. It uses Planning algorithms to check for Preconditions, Actions, and Post-conditions, the inter-relationships to achieve the Production Plan with Material plan. For example, it may create a plan as START → Precondition: Lamp materials available ^ Wattage determined; Action: Add Part to Manufacture Order; Delete: Lamp materials available by consumed amount → FINISH

4. Specific part Process Miner
   This module will extract out all Process specific aspects for a particular part from various history recorded, since there may be changes / similarities in conditions and situations. It will use Case Based reasoning and find similar cases. For example: CBR may be used to decide that 90-grade aluminum can be used instead of 98-grade aluminum since some material is out-of-stock.

5. Parts Process Scheduler
   This part will schedule the parts production based on explicit rules. For example, If Headlamp to be produced then Produce Lamp-holder-part first and then the Lamp-casing.

6. Process Control Configurator

This part will set the process control parameters based on machine learning. For example, for Truck Headlamp it will predict that the pressure needs to be 90psi with probability 0.9 etc.

7. Process Monitor
   This module will be based on explicit and implicit reasoning. There will be parameters that decide if the manufacturing process is going as per plan. Thus, there will be explicit reasoning to determine if there are any major violations, while implicit reasoning will determine if any possible cause of concern may arise. For example, if the bulb is being manufactured for a cabin-light using materials that could possibly heat-up then implicit reasoning may cause an alarm to be raised.

8. Quality Monitor
   This module will determine the quality of the part manufactured and will be based on the image capture of the part that is produced as well as its test report. Thus, the image processing submodule will be based on deep learning to identify defective parts from defect identification images, and the test report will be processed through an explicit rule-based system to check if the test of the part falls within the acceptable range.