CS571: Artificial Intelligence

End Semester Quiz

Name: M Maheeth Reddy	**Roll No**. 1801CS31	**Date**: 24 November 2021

Ans 1:

Gini Index

- calculates the degree of probability of a specific variable that is wrongly being classified when chosen randomly and a variation of gini coefficient.
- works on categorical variables, provides outcomes either be "successful" or "failure" and hence conducts binary splitting only.

Information Gain

- used for determining the best features/attributes that render maximum information about a class.
- follows the concept of entropy while aiming at decreasing the level of entropy, beginning from the root node to the leaf nodes.
- computes the difference between entropy before and after split and specifies the impurity in class elements.

Diferences of Gini Index and Information Gain:

- Gini Index performs binary splitting to determine the efficiency of the split, Information Gain is robust.
- Reason: the Gini algorithm itself experiments with the data in a greedy fashion in-order to land at the best possible split using the concept of entropy.
- Though Information gain is computationally more heavy compared to Gini Index, we can be sure that it enables us to get the best possible split.
- However, if the data consists of large number of partitions, Information Gain will take lots of time to reach optimality. In such cases Gini Index could be preferred.
- Gini's maximum impurity is 0.5 and maximum purity is 0 while Entropy's maximum impurity is 1 and maximum purity is 0

Counterexample: If we have a dataset of 1000 clases, using information gain will take much time as compared to gini index.

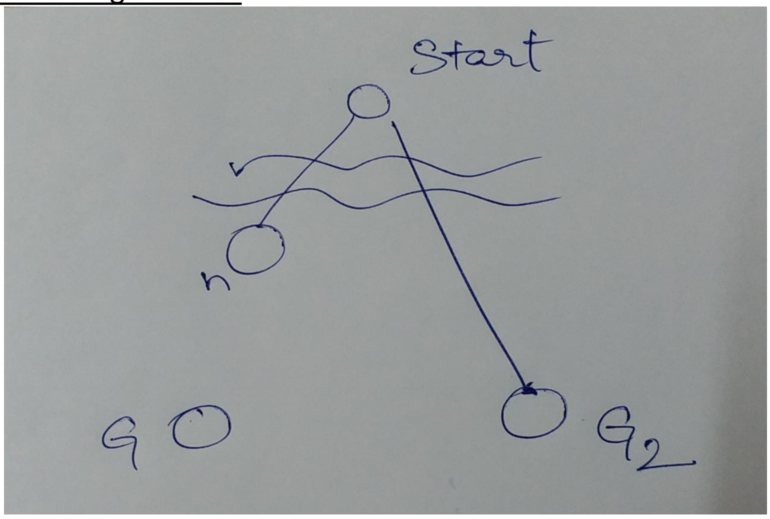
Ans 2:
A heuristic h(n) is said to be admissible if h(n)<=h*(n) where h*(n) is the true cost to the goal→ h(n) never overestimates the cost from node to goal state.

A* is guaranteed to always return the shortest path. All other algorithms return the optimal path must expand all other nodes whereas A* does not.

So, the given statement that "If A* knows the ACTUAL cost of the optimal path of any node to the goal, then no useless node is expanded" is **TRUE**

- We are using an admissible heuristic h(n), so there is no overestimation of the actual cost from a node **n** to goal.
- Since A* heuristic is calculated in such a way that it adds both the cost to that node and cost from that node to Goal state, A* is always guaranteed to return the shortest path and doesnt expand to an useless node

Consider the following scenario:



Node n is an unexpanded node on the shortest path to goal (G) and G_2 is a suboptimal goal.

h(G2) = 0 => f(G2) = g(G2)

As G2 is not optimal, g(G2) > g(G)

so, h(G) = 0 => f(G) = g(G) => f(G2) > f(G)
as n is an unexpanded node, f(n) = g(n) + h(n) <= f(G)
So, f(n) <= f(G). (Since h is admissible)
This implies f(G2) > f(n)

Hence, A* will never expand node G2.
Therefore, A* will never expand a useless node.

Ans 3:
Our knowledge base consists of the following first-order Horn clauses:

Ancestor (Mother(x), x)
Ancestor (x,y) ^ Ancestor (y,z) ==> Ancestor (x,z)

Clearly, we can see that the resolution is complete

We need to prove the following:
~ Ancestor (John, John)
The resolution cannot prove the above statement despite being complete or finished. A careful analysis will show that the statement to be proven doesn't follow the knowledge base.

Reason: There is nothing in the knowledge base that dismisses the possibility of everything being the ancestor of everything else.