Bioinformatics Assignment - LLLL76

QUESTION ONE

A - 25 MARKS

Given a set of leaf nodes, S, and a set of constraints on these nodes, C, build a tree such that all the constraints are true. A single constraint is in the form $(x, y) < (a, b)$ where (a, b) is a single node that is an ancestor of both a and b and has no descendants that are also an ancestor of both a and b. In the example constraint (x,y) is a descendent of (a,b), this forms a constraint. The build algorithm recursively splits the set of leaf nodes into sub-sets so as to create sub-trees in the tree that is created by the algorithms execution. The algorithm starts with a recursion end check, if S has length one then just return that node itself, a single node cannot be a tree. At the first execution of the function BUILD, compute π_c, this is the splitting of the input set given the constraints using three rules. Rule one is for each constraint in the form $(x, y) < (a, b)$, x and y are in the same block. Rule two is for each constraint in the form $(x, y) < (a, b)$, if a and b are in the same block then x,y,a,b are all in the same block. Rule three is that no two leaves are in the same block unless it is shown by rules one and two. Once these rules have been applied a number of subsets exist, 1 to r. If r is equal to one then a null tree is returned because the number of subsets is zero. For each of these numbers calculate C_m or the set of constants that apply to that subset and T_m or the tree that is created by that subset, this is done by recursively calling the build algorithm on the set of nodes that are in this subset and the set of constraints that apply to this subset. Once this has been done for all children of the root node then a tree exists, a new root node is created such that each of its children is the root node of the trees T_x where x is each of the numbers 1 to r corresponding to subsets of the set of leaf nodes. This tree is then returned.

B - 20 MARKS

```
$\pi_c(SetOfNodes):$
        $set = ()$
        $For\ child\ in\ Children(root):$
                $set.append(GetLeaves(child,\ SetOfNodes))$
        $Return(set)$

  $GetLeaves(node, SetOfNodes):$
        $If\ Children(node) = Null:$
                $Return(node)$
        $Else:$
                $set = ()$
                $For\ child\ in\ Children(node):$
                        $set.append(GetLeaves(child,\ SetOfNodes))$
                $Return(set)$
```

C - 25 MARKS

D - 25 MARKS

1 - (e, f) < (k, d)
2- (c, h) < (a, n)
3 - (j, n) < (j, l)
4 - (c, a) < (f, h)
5 - (j, l) < (e, n)

6 - (n, l) < (a, f)
7 - (d, i) < (k, n)
8 - (d, i) < (g, i)
9 - (c, l) < (g, k)
10 - (g, b) < (g, i)
11 - (g, i) < (d, m)
12 - (c, h) < (c, a)
13 - (e, f) < (h, l)
14 - (j, l) < (j, a)
15 - (k, m) < (e, i)
16 - (j, n) < (j, f)

(a b c d e f g h i j k l m n)

Apply 1
(e f)
(a b c d g h i j k l m n)

Apply 2
(e f) (c h)
(a b d g i j k l m n)

Apply 3
(e f) (c h) (j n)
(a b d g i k l m)

Apply 4
(e f) (a c h) (j n)
(b d g i k l m)

Apply 5 - Recurse Apply 3
(e f) (a c h) (j l n)
(b d g i k m)

Apply 6
(e f) (a c h) (j l n)
(b d g i k m)

Apply 7
(e f) (a c h) (j l n) (d i)
(b g k m)

Apply 8
(e f) (a c h) (j l n) (d i)
(b g k m)

Apply 9 - Recurse Apply 2
(e f) (a c h j l n) (d i)
(b g k m)

Apply 10
(e f) (a c h j l n) (d i) (b g)
(k m)

Apply 11 - Recurse Apply 10 and 8
(e f) (a c h j l n) (d i b g)
(k m)

Apply 12
(e f) (a c h j l n) (d i b g)
(k m)

Apply 13 - Recurse Apply 4,5,6,12
(e f a c h j l n) (d i b g)
(k m)

Apply 14
(e f a c h j l n) (d i b g)
(k m)

Apply 15
(e f a c h j l n) (d i b g) (k m)

Apply 16
(e f a c h j l n) (d i b g) (k m)

E - 30 Marks

Question Two

A - 15 Marks

B - 10 Marks