Lista III - Parte I

Construção de árvores filogenéticas; implementação do método Agglomerative method for ultrametric trees (UPGMA).

Pedro Foletto Pimenta - 00229778
INF05018 - Biologia Computacional - Turma: U (2019/2)
Prof. Márcio Dorn

UPGMA (Unweighted Pair Group Method with Arithmetic mean)

- Método hierárquico de clustering para inferência de árvores filogenéticas
- **Entrada:** matriz de distâncias
- **Saída:** árvore ultramétrica filogenética estimada
- **Critério de agrupamento:** a cada passo é escolhido o par de clusters com menor média das distâncias de seus elementos.

$$\frac{1}{|\mathcal{T}_1|.|\mathcal{T}_2|} \sum_{i \in \mathcal{T}_1} \sum_{j \in \mathcal{T}_2} d_{ij}$$

UPGMA: Pseudo-código

1. inicializar lista de clusters
2. enquanto a arvore nao tiver completa:
   a. achar par de cluster com menor média de distâncias
   b. estimar da distância das bifurcações
   c. atualizar matriz de distâncias
   d. fusionar os dois clusters e atualiza lista de clusters
3. retornar a árvore resultante

Entrada: Matriz de distâncias

	Gorila	*Orangotango*	*Humano*	*Chimpanzé*	*Gibão*
Gorila	0	0.1890	0.1100	0.1130	0.2150
Orangotango	0.1890	0	0.1790	0.1920	0.2110
Humano	0.1100	0.1790	0	0.09405	0.2050
Chimpanzé	0.1130	0.1920	0.0940	0	0.2140
Gibão	0.2150	0.2110	0.2050	0.2140	0

Dados obtidos do artigo J Mol Evol. 1982;18(4):225-39. Mitochondrial DNA sequences of primates: tempo and mode of evolution. Brown WM, Prager EM, Wang A, Wilson AC.

Implementação

```
# ultrametric tree class
class U_Tree:

    # initialization
    def __init__(self):
            self.left = None
            self.right = None
            self.left_dist = None
            self.right_dist = None

    # verify if tree is ultrametric
    def is_ultrametric(self): ...

    # distance from this point to the leaves
    def get_leaves_dist(self): ...

    # printing function ( CALL THIS )
    def print_tree(self): ...

    # auxiliary printing function
    def print_subtree(self, subtree, level, dist): ...
```

Implementação

```
105    # agglomerative method for ultrametric trees (UPGMA)
106    def upgma(dist_matrix):
107        # dist_matrix : dicionario com as distancias entre as OTUs
108
109        # inicializacao da lista de OTUs
110        otu_list = []
111        for key in dist_matrix:
112            if(key[0] not in otu_list):
113                otu_list.append(key[0])
114            if(key[1] not in otu_list):
115                otu_list.append(key[1])
116
117        # initialize tree
118        tree_clusters = {}
119        for otu in otu_list:
120            tree_clusters[otu] = otu
121
122        # enquanto a arvore nao tiver completa
123        while(len(otu_list)>1):
124
125            # find smallest distance for clustering
126            otu_a, otu_b = min(dist_matrix, key=dist_matrix.get)
127
128            # branch lenght estimation
129            branch_lenght = dist_matrix[(otu_a, otu_b)]/2
130
131            # update distance matrix
132            dist_matrix = merge_matrix_otus(dist_matrix, otu_list, otu_a, otu_b)
133
134            # update OTU list
135            new_otu = otu_a + '-' + otu_b
136            otu_list.append(new_otu)
137            otu_list.remove(otu_a)
138            otu_list.remove(otu_b)
139
```

```
122        # enquanto a arvore nao tiver completa
123        while(len(otu_list)>1):
124
125            # find smallest distance for clustering
126            otu_a, otu_b = min(dist_matrix, key=dist_matrix.get)
127
128            # branch lenght estimation
129            branch_lenght = dist_matrix[(otu_a, otu_b)]/2
130
131            # update distance matrix
132            dist_matrix = merge_matrix_otus(dist_matrix, otu_list, otu_a, otu_b)
133
134            # update OTU list
135            new_otu = otu_a + '-' + otu_b
136            otu_list.append(new_otu)
137            otu_list.remove(otu_a)
138            otu_list.remove(otu_b)
139
140            # update tree : new tree node
141            new_tree_node = U_Tree()
142            new_tree_node.right = tree_clusters[otu_a]
143            if(isinstance(new_tree_node.right, U_Tree)):
144                new_tree_node.right_dist = branch_lenght - new_tree_node.right.get_leaves_dist()
145            else: # nodo folha
146                new_tree_node.right_dist = branch_lenght
147            new_tree_node.left = tree_clusters[otu_b]
148            new_tree_node.left_dist = branch_lenght
149            if(isinstance(new_tree_node.left, U_Tree)):
150                new_tree_node.left_dist = branch_lenght - new_tree_node.left.get_leaves_dist()
151            else: # nodo folha
152                new_tree_node.left_dist = branch_lenght
153
154            # update tree clusters
155            tree_clusters.pop(otu_a)
156            tree_clusters.pop(otu_b)
157            tree_clusters[new_otu] = new_tree_node
158
159            # DEBUG
160            #print("DEBUG otu list: " + str(otu_list))
161
162        return tree_clusters[otu_list[0]]
163
```

Implementação - "main"

```
# construcao de uma arvore ultrametrica filogenetica a partir da matriz de distancias usando UPGMA
tree = upgma(dist_matrix)

# printar resultados
print("printing resulting tree:")
tree.print_tree()
```

Resultado

```
pfpimenta@PC-pfpimenta:~/biocomp2019/lista3parte1$ python e3-1.py
printing resulting tree:
       - 0.1058125 -    gib
   -
            -   - 0.093625 -   -         ora
   -0.012187500000000004-
      -
                  -   -   - 0.05575 -   -   -      gor
      --0.037875--
         -
                           -   -   -   - 0.047025 -   -   -   -         chi
         ---0.008725000000000004---
                           -   -   -   - 0.047025 -   -   -   -         hum
         -
      -
   -
```

Resultado

A partir do resultado podemos ver quais as OTUs mais próximas:

Chimpanzé e Humano
seguidas por Gorila, Orangotango, e Gibão

Lista III - Parte I

Construção de árvores filogenéticas; implementação do método Agglomerative method for ultrametric trees (UPGMA).

Pedro Foletto Pimenta - 00229778
INF05018 - Biologia Computacional - Turma: U (2019/2)
Prof. Márcio Dorn