Trabalho 3 de Algoritmos de Bioinformática: Análise Automática de Sequências

Edgar Carneiro
Faculdade de Ciências da Universidade do Porto
(Dated: 18 de Junho de 2019)

I. INTRODUÇÃO

No âmbito da disciplina de Bioinformática, disciplina esta que aborda temáticas atuais e relevantes na área da investigação, será analisada a implementação de um módulo capaz de implementar uma ***pipeline*** bioinformática responsável pela execução de forma automática de análise sequências.

A *pipeline* em questão será responsável por receber uma sequência que denominaremos de ***query*** e um conjunto de sequências que constituirão a **base de dados**. Todas as sequências são fornecidas no formato *FASTA*[1], significando que para além de cada sequência, sabemos também qual o seu identificador, o que nos casos trabalhados permitiu também identificar qual a espécie associada.

A *pipeline* é depois responsável por encontrar as dez sequências pertencentes à base de dados com maior semelhança à sequência *query*, sendo que sequências da mesma espécie são desconsideradas.

Após obter as dez sequências mais semelhantes, proceder-se-ia ao seu alinhamento múltiplo. Usando as mesmas sequências, posteriormente obter-se-ia a representação correspondente sobre forma de árvore filogenética, usando o método ***UPGMA*** - *unweighted pair group method with arithmetic mean*[2].

A finalizar a análise automática, é obtido um grafo, tendo como base para sua construção a matriz de distâncias entre sequências.

II. DESCRIÇÃO E ESTRATÉGIAS DE IMPLEMENTAÇÃO

A maior dificuldade encontrada no desenvolvimento deste projeto foi a interligação dos dois módulos desenvolvidos em trabalhos prévios com o código fornecido pelo docente para auxílio de implementação.

No entanto, essa interligação foi conseguida, tornando assim o código extremamente modular e capaz de suportar novas *pipelines* que possam vir a ser de interesse implementar.

Para o desenvolvimento da *pipeline* requisitada, cada um das diferentes fases foi abordada e ultrapassada da seguinte forma:

1. **Sequência a analisar**: A classe **Pipeline** guarda como seus atributos quer a sequência *query*, quer as sequências da base dados, mantendo também os seus identificadores. Esta classe possui também vários métodos que permitem extrair qual a espécie de uma dada sequência.

2. **Análise BLAST**: Nesta fase, é realizada uma cópia da base de dados apenas com sequências de diferente espécie da sequência *query*. Esta cópia é fornecida a uma instância da classe ***MyBlast***, onde é chamado método ***best_ alignments***. Este método é uma modificação do método *best_ alignment* fornecido pelo docente onde ao invés de retornar o melhor alinhamento, o método retorno uma lista de alinhamentos ordenados de forma descendente, com tamanha máximo igual ao parâmetro *top*. Por omissão, este parâmetro tem o valor um, retornando assim apenas o melhor alinhamento.

3. **Alinhmaneto Múltiplo**: Nesta fase, é criado uma instância da classe ***MultipleAlignment*** e chamado o seu método *align_ consensus*, passando como parâmetros a sequência *query*, as dez melhores sequências calculadas no passo anterior e configuração de alinhamento que é mantida pela classe *Pipeline*. Esta configuração é fornecida no constructor da classe, mas pode ser alterada através do método *change_ alignment_ settings*.

4. **Árvore**: Nesta fase, é criada uma instância da classe *UPGMA* e é através desta que obtemos a representação filogenética das onze sequências prévias, através do método *run*. Esta classe é também importante na medida em que faz a computação da matriz de distâncias das sequências. Para visualização da árvore filogenética foi alterado o método *print_ tree* fornecido pelo docente, passando este método a poder receber uma mapeamento entre valor - string. Quando este parâmetro é diferente de *None*, ao invés de nas folhas imprimir os seus valores é antes imprimida a string correspondente ao valor mapeado da folha.

5. **Grafo**: Para esta fase foi desenvolvido o método estático ***create_ from_ num_ matrix*** na classe ***MyGraph*** que recebendo como parâmetro um objecto do tipo *NumMatrix* e um valor de corte, cria um grafo, no qual dois nós só se encontram conectados quando o seu valor na matriz é inferior ao valor de corte. Assim, nesta fase, usa-se a matriz de distâncias calculada na fase anterior e uma lista de valores de corte passada no construtor, de forma a gerar um grafo para cada valor de corte fornecido.

A figura 1 representa de forma gráfica a interação entre os módulos desenvolvidos e a *Pipeline*.

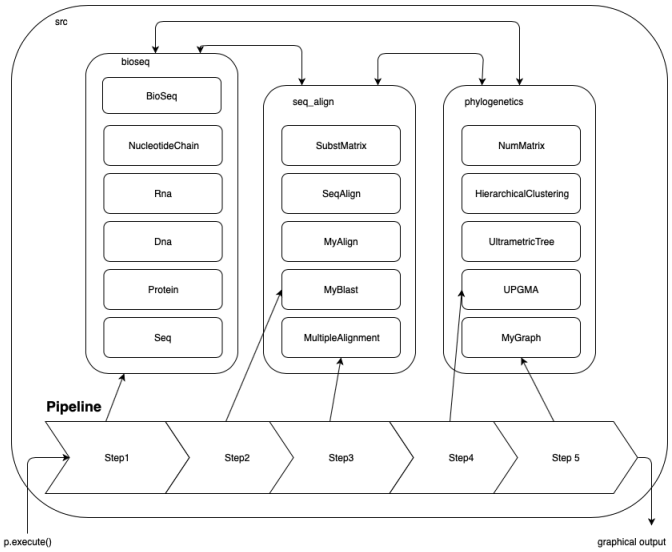

Figura 1. Representação gráfica da *Pipeline*

Em cada etapa da *pipeline* existem diversos parâmetros que podem ser customizados, através da classe *Pipeline*. Na tabela 1 é possível ver qual o valor por omissão desses parâmetros, sendo que todos eles podem ser alterados. Parâmetros sem valor de omissão têm ter os seus valores alterados ou no constructor ou com os métodos correspondentes.

Parâmetro	Valor por Omissão	Funcionalidade
WORD_SIZE	3	Tamanho de subsequências no BLAST
TOP	10	Número máximo de alinhamentos no BLAST
align_data	-	tuplo com Matriz de substituição e valor do gap
cut	-	Lista de cortes para gerar grafos

III. RESULTADOS

Foram implementadas as funcionalidades pedidas na sua totalidade.

Para possível extensão da *Pipeline*, seria necessário retornar os resultados de processamento que vão sendo acumulados, ao invés de apenas exibir estes.

A. Funcionalidades Extra

Foram desenvolvidas várias funcionalidades extra, para facilitar o uso e interligação dos módulos desenvolvidos, sendo estas:

- Para ultrapassar a dificuldade de nos módulos prévios serem aceites classes de Sequências (Dia, Rna e Protein) ao invés de *strings*, foi desenvolvido um método estático **infer_type(seq)**, capaz de inferir o tipo associado a uma *string* e instanciar um objecto da classe correspondente com a sequência fornecida.
- Foram desenvolvidos testes unitários para todas as classes, mesmo aquelas não usadas diretamente na Pipeline, perfazendo um total de setenta e dois testes unitários e noventa por cento de *coverage* do código, sendo que apenas os métodos de printing não foram testados.
- Documentação de todos as classes, mesmo aquelas não usadas diretamente na Pipeline.
- Funções para visualização mais agradável de estruturas e suas métricas associadas.

IV. COMENTÁRIOS E CONCLUSÕES

Acredito que este trabalho foi de todos os trabalhos até então desenvolvidos aquele que mais se assemelha aos projetos que são desenvolvidos nos dias de hoje na área da Bioinformática, ainda que numa perspetiva claramente mais complexa e desafiante. A componente informática encontra-se fortemente interligada à componente biológica, produzindo resultados extremamente interessantes que contribuem ainda mais para o interesse no desenvolvimento do projeto.

Em jeito de conclusão, o resultado obtido com este trabalho foi satisfatório, tendo implementado a totalidade das funcionalidades requisitadas. Considero também que o trabalho desenvolvido, pela estrutura que apresenta, revela-se uma base sólida para possíveis extensões e desenvolvimentos sobre esta, entre as quais novas e diferenciadas *pipelines*.

V. BIBLIOGRAFIA

[1] What is FASTA Format?, https://zhanglab.ccmb.med.umich.edu/FASTA/

[2] Cluster Analysis: UPGMA and WPGMA, Michael WeißMarkus Göker, in The Yeasts (Fifth Edition), 2011

VI. ANEXOS

```
       :::Step 1 - Create copy database without similar specie sequences:::

Correspondent species of sequences in copy Database:
        Pan_troglodytes
        Pan_paniscus
        Gorilla_gorilla_gorilla
        Pongo_abelii
        Colobus_angolensis_palliatus
        Rhinopithecus_bieti
        Rhinopithecus_roxellana
        Macaca_fascicularis
        Macaca_mulatta
        Macaca_nemestrina
        Theropithecus_gelada
        Cercocebus_atys
        Theropithecus_gelada
        Papio_anubis
        Mandrillus_leucophaeus
        Pongo_abelii
        Chlorocebus_sabaeus
        Cebus_capucinus_imitator
        Propithecus_coquereli
```

Figura 2. Exemplo de output gráfico da fase 1

```
        :::Step 2 - Running BLAST and getting top alignments:::

Top 10 Alignments obtained from Blast
(0, 8, 491, 484, 0)
(0, 0, 491, 484, 1)
(0, 0, 491, 479, 2)
(0, 0, 491, 477, 3)
(0, 0, 490, 457, 15)
(0, 0, 491, 453, 5)
(0, 0, 491, 451, 6)
(0, 0, 491, 451, 10)
(0, 0, 491, 450, 11)
(0, 0, 491, 450, 12)
```

Figura 3. Exemplo de output gráfico da fase 2

```
        :::Step 3 - Running MSA with the respective top alignments:::

This might take a little bit...
Multiple Sequence Alignment Result:

0. --------MELSVLLFLALLTGLLLLLVQRHPNTHDRLPPGPRPLPLLGNLLQMDRRGLLKSFLRFREKYGDVFTVHLGPRPVVMLCGVEAIREALVDKAEAFSGRGKIAMVDPF
FRGYGVIFANGNRWKVLRRFSVTTMRDFGMGKRSVEERIQEEAQCLIEELRKSKGALMDPTFLFQSITANIICSIVFGKRFHYQDQEFLKMLNLFYQTFSLISSVFGQLFELFSGFLK
YFPGAHRQVYKNLQEINAYIGHSVEKHRETLDPSAPKDLIDTYLLHMEKEKSNAHSEFSHQNLNLNTLSLFFAGTETTSTTLRYGFLLMLKYPHVAERVYREIEQVIGPHRPPELHDR
AKMPYTEAVIYEIQRFSDLLPMGVPHIVTQHTSFRGYIIPKDTEVFLILSTALHDPHYFEKPDAFNPDHFLDANGALKKTEAFIPFSLGKRICLGEGIARAELFLFFTTILQNFSMAS
PVAPEDIDLTPQECGVGKIPPTYQIRFLPR

1. MQGSQTRTMELSVLLFLALLTGLLLLLVQRHPNTHGRLPPGPRPLPLLGNLLQMDRRGLLKSFLRFREKYGDVFTVHLGPRPVVMLCGVEAIREALVDKAEAFSGRGKIAMVDPF
FRGYGVIFANGNRWKVLRRFSVTTMRDFGMGKRSVEERIQEEAQCLIEELRKSKGALMDPTFLFQSITANIICSIVFGKRFHYQDQEFLKMLNLFYQTFSLVSSVFGQLFELFSGFLK
YFPGAHRQVYKNLQEINAYIGHSVEKHRETLDPSAPKDLIDTYLLHMEKEKSDAHSEFSHQNLTLNTLSLFFAGTETTSTTLRYGFLLMLKYPHVAERVYREIEQVIGPHRPPELHDR
AKMPYTEAVIYEIQRFSDLLPMGVPHIVTQHTSFRGYIIPKDTEVFLILSTALHDPHYFEKPDAFNPDHFLDANGALKKNEAFIPFSLGKRICLGEGIARAELFLFFTTILQNFSVAS
PEAPEDIDLTPQECGVGKIPPTYQIRFLPR

2. --------MELSVLLFLALLTGLLLLLVQRHPNTHGRLPPGPRPLPLLGNLLQMDRRGLLKSFLRFREKYGDVFTVHLGPRPVVMLCGVEAIREALVDKAEAFSGRGKIAMVDPF
FRGYGVIFANGNRWKVLRRFSVTTMRDFGMGKRSVEERIQEEAQCLIEELRKSKGALMDPTFLFQSITANIICSIVFGKRFHYQDQEFLKMLNLFYQTFSLVSSVFGQLFELFSGFLK
YFPGAHRQVYKNLQEINAYIGHSVEKHRETLDPSAPKDLIDTYLLHMEKEKSDAHSEFSHQNLTLNTLSLFFAGTETTSTTLRYGFLLMLKYPHVAERVYREIEQVIGPHRPPELHDR
AKMPYTEAVIYEIQRFSDLLPMGVPHIVTQHTSFRGYIIPKDTEVFLILSTALHDPHYFEKPDAFNPDHFLDANGALKKNEAFIPFSLGKRICLGEGIARAELFLFFTTILQNFSVAS
PEAPEDIDLTPQECGVGKIPPTYQIRFLPR

3. --------MELSVLLFLALLTGLLLLLVQRHPNTHGRLPPGPRPLPLLGNLLQMDRRGLLKSFLRFREKYGDVFTVHLGPRPVVMLCGVEAIREALVDKAEAFSGRGKIAMVDPF
FRGYGVIFANGNRWKALRRFSVTTMRDFGMGKRSVEERIQEEAQCLIEELRKSKGALMDPTFLFQSITANIICSIVFGKRFHYQDQEFLKMLNLFYQTFSLVSSVFGQLFELFSGFLK
YFPGAHRQVYKNLQEINAYVGHSVEKHRETLDPSAPRDLIDTYLLHMEKEKSNPHSEFSHQNLTLNTLSLFFAGTETTSTTLRYGFLLMLKYPHVAERVYREIEQVIGPRRPPELHDR
AKMPYTEAVIHEIQRFSDLLPMGVPHIVTQHTSFRGYIIPKDTEVFLILSTALHDPHYFEKPDAFNPDHFLDANGALKKNEAFIPFSLGKRICLGEGIARAELFLFFTTILQNFSVAS
PVAPEDIDLTPQECGVGKIPPMYQIRFLPR

4. --------MELSVLLFLALLTGLLLLLVQGHPNTHGRLPPGPRPLPLLGNLLQMDRRGLLKSFLRFREKYGDVFTVHLGPRPVVMLCGVEAIREALVDKAEAFSGRGKIAMVDPV
FRGYGVIFANGNRWKVLRRFSVTTMRDFGMGKRSVEERIQEEAQCLIQELRKSKGALMDPTFLFQSITANIICSIVFGKRFHYQDQEFLKILNLFYQTFSLVSSVFGQLFELFSGFLK
YFPGAHRQVYKNLQEINAYIGHSVEKHRETLDPSTPKDLIDTYLLHMEKEKSNAHSEFSHQNLTLNTLSLFFAGTETTSTTLRYGFLLMLKYPHVAERVYREIEQVIGPHRPPELHDR
AKMPYTEAVIHEIQRFADLLPMGVPHIVTQHTSFRGYIIPKDTEVFLILSTALRDPHYFEKPDAFNPDHFLDASGALKKNEAFIPFSLGKRICLGEGIARAELFLFFTTILQNFSVAS
PVAPEDIDLTPQECGVGKIPPTYQIRFLPR

5. --------MELSVLLFLALLTGLLLLLVQRHPNTHGRLPPGPRPLPLLGNLLQMDRRGLLKSFLRFREKYGDVFTVHLGPRPVVVLCGVQAIREALVDKAEAFSGRGKIAIMDPV
YQGYGVIFANGNRWKVLRRFSVTTMRDFGMGKQSVEERIQEEAQCLIEELQKSKGALMDPTFLFHSITANIICSIVFGKRFHYQDQEFLKMLNLFCQSFSLISSISSQLFELFSGFLK
YFPGAHRQLYKNLQEINAYIGHSVEKHRETLDPSAPQDLIDTYLLHMEKEKSNPHSEFSHQNLIINTLSLFFAGTETTSTTLCYGFLLMLKYPHVAERVYKEIEQVVGPHCPPVLDDR
AKMPYTEAVIHEIQRFADLLPMGVPHIVTQHTRFRGYIIPKDTEVFLILSTALRDPHYFEKPDAFNPDHFLDANGALKKNEAFIPFSLGKRICLGEGIARAELFLFFTTILQNFSVAS
PVAPEDIDLTPQECGVGKIPPTYQIRFLPH

6. --------MELSVLLFFALLTGLLLLLVQHHPKAHGRLPPGPRPLPLLGNLLQMDRRGLLKSFQRFREKYGDVFTVHLGPRPVVMLCGVEAIREALVDNAEAFSGRGKIAIVDPV
FQGYGVVFANGNRWKVLRRFSLTTMRDFGMGKRSVEERIQEEAQCLIEELRKSKGALVDPTFLFHSITANIICSIVFGKRFHYQDQEFLKMLNLFYNTFSLTSSIFGQLFELLSGFLK
YFPGVHRQVYKNLQEINAFIGHSVEKHRETLDPSAPQDLIDSYLLQMEKEKSNPHSEFSHQNLILNTLSLFFAGTETTSTTLRYGFLLMLKYPHVAERVYKEIEQVIGPHRPPALDDR
AKMPYTEAVIHEIQRFADLLPMGVPHIVTQQTSFRGYIIPKDTEVYLILSTALHDPHYFEKPDAFNPDHFLDANGALKKNEAFIPFSLGRRICLGEGIARNKLFLFFTTILQNFSVAS
PVAPEDIDLTPQESGVGKIPPTYQIRFLPR

7. --------MELSVLLFFALLTGLLLLLVQHHPKAHGRLPPGPRPLPLLGNLLQMDRRGLLKSFQRFREKYGDVFTVHLGPRPVVMLCGVEAIREALVDNAEAFSGRGKIAIVDPV
FQGYGVVFANGNRWKVLRRFSLTTMRDFGMGKRSVEERIQEEAQCLIEELRKSKGALVDPTFLFHSITANIICSIVFGKRFHYQDQEFLKMLNLFYNTFSLTSSIFGQLFELLSGFLK
YFPGVHRQVYKNLQEINAFIGHSVEKHRETLDPSAPQDLIDSYLLQMEKEKSNPHSEFSHQNLILNTLSLFFAGTETTSTTLRYGFLLMLKYPHVAERVYKEIEQVIGPHRPPALDDR
AKMPYTEAVIHEIQRFADLLPMGVPHIVTQQTSFRGYIIPKDTEVFSLLSTALHDPHYFEKPDTFNPDHFLDANGALKKNEAFIPFSLGRRICLGEGIARNELFLFFTTILQNFSVAS
PVAPEDIDLTTQESGVGKIPPTYQIRFLPR

8. --------MELSVLLFLALLTGLLLLLVQRHPNAHGRLPPGPRPLPLLGNLLQMDRRGLLRSFLRFREKYGDVFTVYLGPRPVVMLCGVEAIREALVDNAEAFSGRGKIAITDPV
```

Figura 4. Exemplo de output gráfico da fase 3

```
       :::Step 4 - Obtaining the Ultrametric Tree from the top alignments, using UPGMA:::

Distances Matrix obtained by the UPGMA method:
0       0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0
15      0       0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0
7       8       0       0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0
12      17      9       0       0.0     0.0     0.0     0.0     0.0     0.0     0.0
14      19      11      14      0       0.0     0.0     0.0     0.0     0.0     0.0
34      42      34      34      33      0       0.0     0.0     0.0     0.0     0.0
38      45      37      37      37      41      0       0.0     0.0     0.0     0.0
40      47      39      39      39      43      6       0       0.0     0.0     0.0
40      47      39      39      38      41      21      18      0       0.0     0.0
41      48      40      40      39      43      21      18      2       0       0.0
41      48      40      39      39      42      22      19      1       3       0

Phylogenetics Tree created:

 Root - Dist.:  20.433333333333334
         Left - Dist.:  17.7
                 Left - Dist.:  7.375
                         Left - Dist.:  6.5
                                 Left - Dist.:  5.25
                                         Left - Dist.:  3.5
                                                 Left: Pan_paniscus
                                                 Right: Homo_sapiens
                                         Right: Gorilla_gorilla_gorilla
                                 Right: Pongo_abelii
                         Right: Pan_troglodytes
                 Right: Pongo_abelii
         Right - Dist.:  9.916666666666666
                 Left - Dist.:  3.0
                         Left: Rhinopithecus_roxellana
                         Right: Rhinopithecus_bieti
                 Right - Dist.:  1.25
                         Left - Dist.:  0.5
                                 Left: Theropithecus_gelada
                                 Right: Theropithecus_gelada
                         Right: Cercocebus_atys
```

Figura 5. Exemplo de output gráfico da fase 4

```
        :::Step 5 - Creating Graph using UPGMA distance matrix and cut values [10, 5, 15]:::

GRAPH:
Graph nodes:
         [2, 0, 1, 3, 7, 6, 9, 8, 10]

Graph edges:
         Edge
         (2, 0)
         (2, 1)
         (3, 2)
         (7, 6)
         (9, 8)
         (10, 8)
         (10, 9)

GRAPH METRICS:
Top 3 highest degree nodes:
         Node
         2
         9
         8

Mean degree: 1.5555555555555556

Mean distance: 1.2222222222222223

Clustering Coefficient of all nodes:
         Node - Clustering Coefficient
         2 - 0.0
         0 - 0.0
         1 - 0.0
         3 - 0.0
         7 - 0.0
         6 - 0.0
         9 - 1.0
         8 - 1.0
         10 - 1.0

Mean Clustering Coefficient: 0.3333333333333333
```

Figura 6. Exemplo de output gráfico da fase 5, para um dos valores de corte.