UNIVERSIDADE DE SÃO PAULO
INSTITUTO DE FÍSICA DE SÃO CARLOS
BACHARELADO EM CIÊNCIAS FÍSICAS E BIOMOLECULARES

JADER OLIVEIRA LEITE FILHO

Estudos biocomputacionais de elementos transponíveis de *Taenia solium*

São Carlos
2018

TRABALHO DE CONCLUSÃO DE CURSO - MONOGRAFIA

ESTUDOS BIOCOMPUTACIONAIS DE ELEMENTOS TRANSPONÍVEIS DE *Taenia solium*

Aluno: Jader Oliveira Leite Filho
Curso: Bacharelado em Ciências Físicas e Biomoleculares
Orientador: Profº Dr Ricardo de Marco

São Carlos
2018

SUMÁRIO

1. Resumo

Partindo do fato de que organismos eucariotos possuem uma quantidade de DNA repetíveis, derivados de elementos transponíveis e que seu estudo tem trazido bastante informação sobre evolução e origem de organismos [1], decidiu-se estudar o genoma, já conhecido, do platelminto *Taenia solium* a fim de identificar elementos repetíveis no mesmo. Análises de quadro de leitura, em inglês open reading frames, árvore filogenética, alinhamento, entre outros permitiram que se pudesse ser retiradas informações importantes acerca de tais elementos. Este Trabalho de Conclusão de Curso descreve a atividade de Iniciação Científica realizada pelo aluno durante o ano de 2018 com a orientação do Professor Ricardo de Marco.

2. Introdução

2.1. Elementos Transponíveis, transposons

Muito estudados porém ainda pouco conhecidos [2], os transposons são elementos repetíveis lineares que se movem dentro do genoma dos organismos, podendo se replicar ou não. Existem diferentes tipos de classificação de transposons. Os de classe I são os que transpõe via RNA, através da codificação de integrases e transcriptase reversa, proteína gag e RNAse-H para a transposição, os chamados retrotransposons, possuem LTR (Long Terminal Repeats) nas duas extremidades, temos também o que não possuem essas extremidades terminais longas, os chamados non-LTR também conhecidos como “long interspersed element” (LINE) que normalmente produzem transcriptase reversa e endonuclease para a sua inserção.

O de classe II, são aqueles que transpõe do DNA, esses têm sua transposição por duas formas diferentes, a conservativa também chamada de corta e cola, no qual deixa o fragmento de DNA inicial para se inserir em um outro, e a replicativa também chamada de copia e cola, nesse o transposon é replicado, e inserido no mesmo cromossomo ou em um outro [3][4]. Nessa classe temos que toda a transposição é dada pela maquinaria da célula gerida pela enzima transposase que identifica as TIR’s, “terminal inverted repeats” encontradas nas terminações do DNA dos elementos transponíveis de classe II [5].

3. Métodos

3.1. Identificação dos elementos repetíveis

A primeira parte do projeto constitui-se na determinação dos possíveis elementos transponíveis através da identificação de elementos que se repetem dentro do genoma, para tal determinação realizou-se dois procedimentos, o primeiro através do pacote REPET e o segundo método utilizado foi através do programa Repeat Scout. O pacote REPET é um software cujo o propósito é identificar, classificar e anotar elementos repetíveis através de duas pipelines diferentes (TEdenovo e TEannot). TEdenovo é constituída de 8 passos distintos: separar o genoma em lotes, alinhar o genoma contra ele mesmo, agrupamento dos pares de maiores pontuação, definição de uma sequência consenso para cada agrupamento, classificação de cada sequência consenso, filtragem da sequências consenso que não foram classificadas, agrupamento das sequências consenso em famílias. TEannot explora o genoma com uma biblioteca de sequências de elementos transponíveis gerados pela primeira pipeline, os falsos positivos são descartados através filtros estatísticos e os SSR's (shorts simple repeats) são anotados com programas como RepeatMasker, TRF, e outros. Por fim realiza um processo que identifica os pequenos fragmentos pertencentes ao mesmo elemento e os une. [6]

Repeat Scout tem como propósito identificar em genomas famílias de sequências no qual banco de dados de elementos repetíveis ainda não foram identificados, funciona em quatro estágios: criação da tabela que identifica a frequência dos chamados lmers, comparação da sequência a ser analisada e a tabela criada a fim de gerar uma lista com todos os elementos repetidos encontrados, filtragem dos elementos e por fim a mascaração dos elementos [7]. Como o objetivo foi a identificação dos elementos repetidos parou-se no segundo estágio do programa. Para a instalação do mesmo foi necessário instalar também o Blast N e o Repeat Masker.

3.2. Classificação dos elementos repetíveis

Com as sequências de elementos repetidos identificados no passo anterior, o próximo passo foi identificar se nessas tais sequências existiam RNAs que executavam papéis similares a proteínas.

Para isso utilizou-se o Rfam [8], que contém uma coleção de famílias de RNA representados por alinhamento múltiplo, modelos de covariância e estruturas de consenso secundário convertendo-os em modelos computacionais utilizados para anotação de sequência de DNA e/ou RNA [9]. Como tinha-se uma quantidade inputs muito grande ficou-se inviável, comparar um por um via website, por isso optou-se pelo envio do tipo "batch search" no qual se envia arquivos multifasta para o site e ele retorna os resultados por email. Em seguida rodou-se um blast local no qual colocou-se o resultado do Rfam contra o genoma a fim de quantificar as repetições, ou seja, quantas vezes esses elementos se repetiam dentro do genoma.

3.3. Anotação dos elementos repetíveis

Pegou-se o arquivo resultante do Repeat Scout e rodou como query no programa Sma3s [10][11], sendo esse uma ferramenta que associa informações biológicas da sequência usada como query para extrair informações referentes a potencial função, localização celular ou estrutura proteica. Possui precisão de +/- 80%, e por mais que exija pouco computacionalmente, como o número de inputs do query era muito grande optou-se por rodar através do cluster utilizando 16 threads.

3.4. Classificação filogenética

Após a anotação via Sma3s pode-se identificar uma sequência com a qual podia se desconfiar que era um retrotransposon, para então confirmar-se tal premissa partiu-se para a classificação filogenética para isso utilizou-se sequências de retrotransposons já conhecidos e também trabalhados em [12]. Ou seja pegou-se a proteína codificada pelo possível elemento e fez-se um alinhamento múltiplo com as proteínas do artigo do professor De Marco, com o programa Clustal X e montou-se a árvore filogenética com o programa MEGA X.

3.5. Gráfico de região codificante

Pegou-se a sequência do retrotransposon encontrado e rodou-se o blast dele contra o arquivo SRA do genoma a fim de localizar as regiões codificantes e a frequência da mesma. O SRA é um arquivo

constituído por informações de alinhamento e sequenciamento de três grandes instituições: NCBI Sequence Read Archive (SRA), The European Bioinformatics Institute (EBI), e DNA Database of Japan (DDBJ), fazendo com que as informações sejam trocadas de forma rápida, permitindo assim que seja possível reproduzir experimentos e realizar novas descobertas comparando os banco de dados preexistentes [13]. Para a identificação da frequência da região codificante foi escrito um script em python que lê o resultado do blast e exporta as coordenadas para a criação do gráfico. O programa escrito está descrito na imagem abaixo.

```
a1 = open("qstart", "r")
a2 = open("qend", "r")


qstart = []
qend = []

for linha in a1:
    qstart.append(int(linha))
for linha in a2:
    qend.append(int(linha))


tupla = zip(qstart, qend)

a1.close()
a2.close()

grafico = []
count = 0
valor = 0

for i in range (len(tupla)):
    for qstart, qend in tupla:
        if valor >= qstart and valor <= qend:
            count += 1
    grafico.append(count)
    valor += 1
    count = 0
a3 = open("grafico", "w")

for j in grafico:
    a3.write(str(j))
    a3.write("\n")

a3.close()
```

Figura 1: *Programa escrito pelo aluno, para a criação do gráfico da região codificante*

No programa acima, a1 e a2 recebem os arquivos que saem de resultado do blastn do retrotransposon encontrado contra o SRA, "qstart" possui o inicio do alinhamento e "qend" possui o final do alinhamento, abre-se então duas listas separadas com os respectivos nomes para facilitar o entendimento, e com o comando zip, une-se as duas listas em uma tupla, abre-se uma lista vazia que

receberá o eixo y do gráfico. É feito então um for que percorre toda a tupla e para cada par encontrado nessa tupla é comparado se o número desejado está contido naquele intervalo, ou seja, naquele alinhamento (qstart, qend) se sim a variável auxiliar "count", que conta quantas vezes o valor está nos alinhamentos, incrementa em 1, por fim escreve-se o valor de count na lista do gráfico, o valor é incrementado em 1 para reiniciar o loop e o valor de count é resetado. Finalmente é escrito a lista em um arquivo para ser feito o gráfico. Para a montagem do gráfico utilizou-se o programa OriginLab e o eixo x recebeu os números de 1 a 4329 e o eixo y é o resultado do programa acima.

3.6. Identificação se o transposon é interno, externo ou sobreposto

Nessa parte do estudo, no wormbase pegou-se o arquivo gff3 da anotação da *Taenia solium,* selecionou-se todas coordenadas referentes aos genes dos determinados "contigs" e comparou com resultado do blast do retrotransposon encontrado contra o genoma inteiro a fim de identificar se os alinhamentos dele estariam internos, externos ou sobrepostos aos genes, nesse caso também fora escrito um script em python para tal análise. O programa escrito está descrito na imagem abaixo.

```
a1 = open("resultado_blast_transposonxgenoma_qstart", "r"); a2 = open("resultado_blast_transposonxgenoma_qend", "r")
a3 = open("inicio_gff3","r"); a4 = open("final_gff3","r"): a5 = open("resultado_blast_transposonxgenoma_contig", "r")
a6 = open("contig_gff3","r")

contiggff = []; contigtransposon = []; qstart = []; qend = []; inicio = []; fim = []


for linha in a1:
    qstart.append((linha))
for linha in a2:
    qend.append((linha))

for linha in a3:
    inicio.append(linha)
for linha in a4:
    fim.append(linha)

for linha in a5:
    contigtransposon.append(linha)
for linha in a6:
    contiggff.append(linha)


tupla = zip(contigtransposon,qstart, qend); tupla2 = zip(contiggff,inicio, fim)

a1.close(); a2.close(); a3.close(); a4.close(); a5.close(); a6.close()


fora_antes = 0
parcial_antes = 0
dentro = 0
parcial_depois = 0
fora_depois = 0

for contiggff,inicio, fim in tupla2:
    for contigtransposon,qstart, qend in tupla:
        if contigtransposon == contiggff:
            if  qend <= inicio:
                fora_antes += 1
            elif qstart <= inicio and qend >= fim:
                parcial_antes += 1
            elif qstart >= inicio and qend <= fim:
                dentro += 1
            elif qstart >= inicio and qend >= fim:
                parcial_depois += 1
            elif qstart >= fim:
                fora_depois += 1

print fora_antes, parcial_antes, dentro, parcial_depois, fora_depois
```

Figura 2: *Programa descrito para identificação se os alinhamentos estão internos, externos ou sobrepostos aos genes da anotação*

Nesse programa, pega-se as colunas saccver, qstart e qend resultado do blastn do retrotransposon contra o genome em arquivos separados e os armazena nas variáveis a5, a1 e a2, respectivamente, enquanto as colunas, contig, início e fim do arquivo de anotação de *Taenia solium* referente a gene são armazenados nas variáveis a6, a3 e a4 respectivamente. Então as respectivas listas são preenchidas com os valores dos arquivos e as tuplas são formadas. As variáveis auxiliares que conta

quantos estão externos, internos ou sobrepostos são iniciados em zero. O primeiro for percorre a primeira tupla, ou seja a tupla referente ao arquivo gff, enquanto o segundo for percorre o resultado do blastn, se os contigs batem, ou sejam é o mesmo gene daí compara-se se é interno, externo ou sobreposto, a variável auxiliar incrementa em 1 e o ciclo se reinicia com o próximo valor da tupla.

4. Resultados Obtidos e discussão

4.1. Identificação de elementos repetíveis

No primeiro software utilizado, REPET, não obtivemos resultado, uma vez que a instalação do mesmo foi extremamente complicada e o mesmo possui dependências antigas não mais encontradas nas novas versões do Ubuntu, por isso optou-se pelo uso do Repeat Scout, nesse o resultado obtido foi um arquivo em formato fasta com 2927 sequências que se repetem em todo o genoma da *Taenia solium*, esse arquivo será usado durante quase todo o trabalho como query de outros programas.

4.2. Classificação dos elementos repetíveis

É sabido que o RNA ainda catalisa várias reações nas células agindo como proteínas [14], nessa parte do trabalho separou-se o arquivo anterior do Repeat Scout em três outros arquivos, pois a "batch search" só suporta arquivos com até 1000 sequências diferentes, e os enviou para o Rfam para ver se alguma dessas sequências eram retrotransposons que atuam como proteína na célula, o resultado obtido foram doze sequências que atuavam como RNA transportador (tRNA), para confirmar a importância dessas sequências, rodou-se o blastn local dessas doze sequências encontradas contra todo o genoma, com um corte de e-value de 10^{-5}, para ver o quanto elas se repetem, é esperado para um transposon no mínimo cinquenta repetições, nesse caso o resultado

obtido fora três sequências que se repetiam tantas vezes. Rodou-se um Blastx, via web, e pôde-se chegar a conclusão de que nenhuma delas eram RNA exercendo papel de proteína.

4.3. Anotação do elementos repetíveis

Uma vez que não chegamos ao resultado desejado, de se encontrar um retrotransposon exercendo papel de proteína, retrocedemos ao arquivo do Repeat Scout e partimos para a anotação dessas sequências a fim de se obter informações sobre elas, para isso rodou-se o programa Sma3s, o resultado obtido foram quatro sequências que possuíam domínios de transcriptase reversa, após análise básica de sequências, chegou-se à conclusão de que todas faziam parte de uma mesma sequência, sendo essa um retrotransposon possuindo um domínio de transcriptase reversa, possuindo também LTR nas duas extremidades. A sequência obtida está descrita abaixo, sendo as partes destacadas em amarelo o LTR.

TGCAGCGGGATCCCCCCATTCCTCCTCTTCCTTCTTTCCCATCCTGCTGTTAGAAGGCGGAAT
*ATTACGGGATATTCTTTATTGTAAGACTGTTGGGCAATAGCGGTAGGC*TCATATCTACGACGGG
GAAGTTCCCGACTCCTGAGAGGAAGCATGCGTGTCGGCGTCCAAGTTAAAATTGTCCTCCCG
CCTCTGACCTACAAAGCACATTTAAACACTACGATTATTACACAGCCCCCTCATTTTTTGAGCG
CCGCCCCCGGTGCTACTCTCGTTCGTTGGCCTTAGTTATGCCATCGAGCGGTTTTCCTTCCTC
CATAATTCCGCAGACTTCGTCCTCGTAGCCAACCGGTGAGGGACCTTTGTAAGGTCGCATCTT
ATTATGATGGACCGTTATAGGCTGGGTACGGAGTTCCGCATTTCGGACAAGGTATTTCGTGGG
GAAGAGAACTTTTACTACGCAGAAGTGATCTTTGCTCCAAGGACGGTAAAATTTATGGTGAGTT
CCAGGTGGGGAAATCGGCTTGTATATTCGGACGAGATCTCCTTCGTGGTAGGTGTTGGGTCG
AATGTGCTTGGCATAGTACTTCTTTTGTCTACTATGAGGCATTCGTAAGTACCGCTGGGCCATAT
TGAACGTCTTTCTTATCCCTTCTTTTAACTCAGTACGTATTCCGGCACATTGTCGGT
TGCTGCTTCTTTGCTCGATAGAAAGATATCGGACGGAACTCTCATTTTACAGCCTGTGAGCATT
TTGAACAGAGAAACACCCGTTGACGTGTGGGTTGTGTCTTTATACGCTAGAAGGGCGCGTCCT
AGGCTTAGATCCCAGTCTTCCGGCTTTGCCCCTTTGGTAAAAAAATTCAGCAGTCCTACCAGA
GTGAGATTCGTTTTCTACCTGTCCGCTGCCTTGTGGGTGCCCTGGTGTCGTGCGCGTCTTTG
CGATTCCAAAAGTCTTGCGTAGTTTAACAAAAAGGCGGCTTTCAAAGTTAGAACCTTGATTGCT
ATGAATTGATTCGGGCACTCCGTGTTGACAGATCCAACGGTTGAAGAAGGCTGACGCGACTG
TCTCGGCGTCTTGAGACTTCATGGCTTCTGTTTCGGCTACTTTCGTGAAGTAGTCAACCATGA
CTAGGATACACCGGTTTCCCCTTTTGGTAAGTGACAGAGGACCAATAATGTTAATCTCCACCCC
TTTGCCCCGGAATCCTGTAGGCATGGGTGAGTGAGGCTTCTTGAAACTGCTGCAAGTGATGTA
GGTTCGGCAGAAATCGATCGCGTCCGGGATAAGAGAGGGCCACCACTATCGCTTGCTCGAG
GCTTCCACCATCTTCTTCTCACCTACATGCCCCAATTGCTCGTGCAACTCTTGAAGGACCGTTT
GTATCAGCGAACCTGGAACAACTAGACTTTTGAGAGACGTGGCATCTTCTTGATACCATAGAAC
TTCATCCTCCAGGTGAGTTTGGACCACTGCCGCCATATCCGTTTGGCTGTTTTGCTGGACGAG
TTCATCTCTTCTGCAGTGGGTTTGTGCGAGGATGCCAGAAACGTCTTGCATACCAAAGCCGTA

TCGGGATAGGTACTTTGTGTGTTTCTCCACTGGTGTCTAGTAGGCTCGCTCAAGAATAGCGTA
CCCGCTATACCGCTTTCCCTCTCTGCTGAGAGAGGTCTGCGAGACAATGCGTCCGTGTTGCT
GTGGGTAGTCCCTTTCCTGTACTGCACTGTAAAGTCGTATTGCTGCAGTTCCTCGTACCAGCG
CGCCACTGAGTGGTCAATCTCCTTCATTGTTCTTAGCCAGGTAAGCGCTTGGTGGTCCGTTTT
CACAATGGACTGTTTCGCTATCAGATAGTGTTTAAAGTGCCGAACCATTGTAAAGATGGCGAAC
AATTCACGTTCCATCGCACTTCTCTGTCTCATCTTTTTGTTGAGCCTGATACTGGCGTATGCGA
TGACGTGTTCCCTACCTTCTTTATCCCGTTGGGATAGCACACCATCCACGGCTACGTCGCTGG
CGTCCATGTCTAGTACAAAAGGAGGTGCATCGTTCTCGAAGTTGGGTAAGGCCAAAATCGGG
GCAGAGCAGAGCATTCGCTTTAGTTCCTTAAAGGCCTCATCATGTTCGCTCTCCCACTTGAAAT
TCTTTTTCGCCTGCTTCTCTGTTAGTTTGTGCAGAGGACCGGCGATCTTGGCGAAGTTTTTGA
CGAAACGTCGGTAGTAGTTTGCCAAGCTGAGAAAACTGCGGAGCTCCGTCTGATTGGTCGGT
GTCGGCCATGTTCTCACTCGGTTTGTACGGTCTTCTGTGACCGCCATCCCATCTGACAAGACT
GTATGCCTCAGGAACGTAACCGAACGTTGAAGGAAGTGGCATTTCTTCGGGTCCAGGGTCAA
TCCAGCGTCTCGAAGACGATCCAGCACTAGTTTCAGATTGGCATTATGCTCCTGCGTGTCTTTA
CCAAAGACAAGAATGTCATCGAGGTAGATTATGCAGTGCTTTGGAAAAAGCTCTATCAGTGCC
GTCTGCATCAGGCGCTGGAAAGTAGCTGCGGCGTTACATAGTCCAAACGGCATTGTCTGGAA
TTCATAGAGTCCGTTTGGCACTATGAAAGCTGTCTTCTCTCTGCCCGTCTCCGCTACTTCCACT
TGCCAGTAGTCGGATTTTAAGTCTAGAGTGGAAAACCACTTTGCCCTCGTGCAGGGAATCCAG
TGAGTCATTGATGTGAGGTAGTGGAAAAGCGTCCTTTAGTCACGTCATTTAGTTTTCTGTAGTC
AATGCACAGTTGTAGGCTCCCGTCACTTTTCTTTACTAGTGCGATCGGGTAGGCCCACGGTGA
CTTGGATGGCCGTATAACGTTGTCGTTGATCATCTCATTCACTAAACAATTTACACCTTCCAAGA
GAGGAGGCGGTATGCGTCTCGGGGGCTGCCATATTGGTCTAGCTTCACTTGTGTCGATGGCA
TATTTGATGATGTTCGTCCGCCCGAGTTTGGTTCCCTGCCAAGCGAACATTTTGGAGTATCTG
CCCAGCAATGAGTTCAGTTCCTTCCTTTCGCTGTCGGTGATGTAATTTAACTGTGAGCAAAGCT
CATCTGGGTTGTTAACAGAAATACCCGCTGCTTCGAAAAGGGCACTGCAAATTTCGTCAGCGT
CCTTTTCCGGGGAAGACGCGACTGAATATGTTGTTTTGTGCTGCTGGGCCGTAAAGGTACCCT
CTGCAAAGTTCAGGATTGCTTTAGTCTTTCGAAGGAAATCAGCGCCCAGTATCACGTCCCATA
CTAATTCTGGACACATAATAAATTGTACTGTCCAGGTTTCTTTCCCCACTGTGACCTTTAACGAC
GTCCCTCCGATCGCTTTCATCTTCCTCCCTTCATCTGAGAGTAGTTTTATAGAGGAAAGGCCG
GCACGAAACTTGCAGAGCAAATCTAAGAGCGCTTTGGGATTCACTAGGGACTTGACTGCTCC
CGAGTCCAGGAGAAAACGACAAGGATCTCCTTCCAACTTGCCGAGGGCTTGTAAAAATGGAC
GAGAGGCATTTGGGGGTTGGCAAGATAACGAATAGTAGGCCTCACCTGCTCTTGGGATATGAC
GTGATGGAGTCCAACGAGGGTGTGGCGGGTTTGATCGGCGGTGGGCCGTCTGGTGCGTCG
ATGTCAGGTGGAGTTCCTGGCGCTTCCTGGTCGCAGCCTCGACCGACTGATTGAGATCGTTG
GTCTTCATGGCATTGGGCTTTGCGGCGATGGTCAGTGGTCGGACTCCTGCGCAGAATTGGAC
GGCCACCCAGTTGGTCACTCTGGTTGGAGGGCAACCTCTGAAAGCTGGTTCCGCTAGGTGTT
GTAGGTTCCTAGCGTATTCTTCGTCGTTTTCGCCCACATTTTGGTCTCGGTGGAAAAAATCTCT
AGCCAGCGATTGTTTTCGCTGATCGATGGCCAGTTGAGAGAGAATCTCGCAGCAGTGGTCGA
TGTCGGAGTCAGCAGTGACTCCTGCGTCTATGGCTACCAGGAATAGCTCTTGCGAGAGGGCG
TGGAGGATCAGCGGGACTCTCTGCCGCTGAGGATATAAATGGATGTACAGCCAAGCAGTTTTC
ATCCAGTCTTCAAAGTTCATACCGTCTTTGAAGTGGGGGAG
GTCCGCCAACGCTGCTACCAGTGTAGCGGGATCTTCCTATTCCTCCTCTTCCTTCTTTCCCAC
CCTGCTATTAGAAGGCGGAATATTACGGGACATTCTTTATTGTAAGACTGTTGGGCAATAGTGG
TAGGCGTCCAAGTTAAAATTGTCCTCCCGCCTCTGACCTACAAAGCACATTTAAACACCACATT
TGTTACAC.

Analisou-se também o open reading frame (ORF), ou seja, quadro de leitura da sequência acima e obteve-se o seguinte resultado:

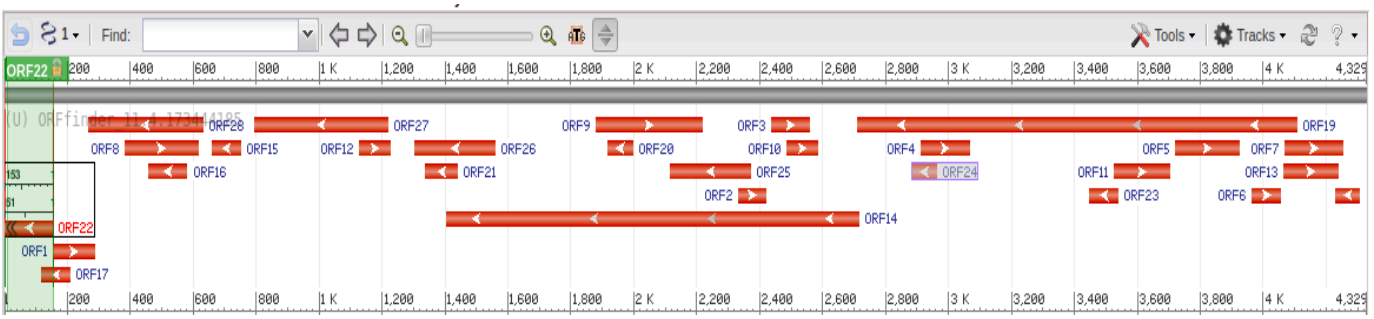

Figura 3: *Resultado obtido para o quadro de leitura da sequência obtida através do site https://www.ncbi.nlm.nih.gov/orffinder/ [15]*

Como pode-se observar existem vinte e oito orf's na sequência acima, no entanto pode-se notar que existem duas em específico que apresentam um tamanho mais considerável, analisando-as chegou-se à conclusão de que ambas fazem parte do mesmo quadro de leitura, o do domínio de transcriptase reversa, o que pode ter ocorrido para que ela se separasse é que durante a clivagem e reinserção do retrotransposon criou-se um frame shift degradado. Desenhou-se também o retrotransposon encontrado de acordo com os seus domínios conservados, encontrados no CDD do ncbi. O resultado está descrito na imagem abaixo.

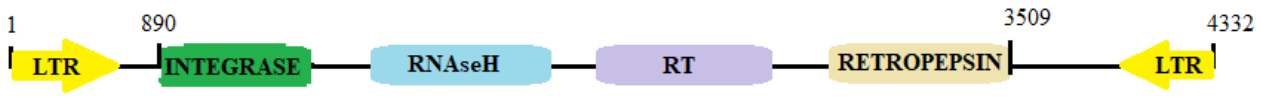

Figura 4: *Retrotransposon encontrado com os domínios conservados, de acordo com análise no CDD ncbi.*

Sendo que a integrase é responsável pela inserção do retrotransposon no novo lócus do DNA, a RNAseH é responsável pela degradação do RNA original, RT é a transcriptase reversa que converte o DNA em RNA para inserção e por último a retropepsina é a protease. Essas quatro enzimas fazem parte da POL, poliproteina.

4.4. Classificação filogenética

Pegou-se então a tradução dos dois quadros de leitura sobrepostos, sendo que um começa 1405 e termina em 2712 e a outra parte começa em 2712 e acaba em 4109, portanto o orf principal começa em 1405 e acaba em 4109 e o resultado traduzido fica:

"MNFEDWMKTAWLYIHLYPQRQRVPLILHALSQELFLVAIDAGVTADSDIDHCCEILSQLAIDQRK QSLARDFFHRDQNVGENDEEYARNLQHLAEPAFRGCPPTRVTNWVAVQFCAGVRPLTIAAKPNA MKTNDLNQSVEAATRKRQELHLTSTHQTAHRRSNPPHPRWTPSRHIPRAGEAYYSLSCQPPNASR PFLQALGKLEGDPCRFLLDSGAVKSLVNPKALLDLLCKFRAGLSSIKLLSDEGRKMKAIGGTSLK VTVGKETWTVQFIMCPELVWDVILGADFLRKTKAILNFAEGTFTAQQHKTTYSVASSPEKDADEI CSALFEAAGISVNNPDELCSQLNYITDSERKELNSLLGRYSKMFAWQGTKLGRTNIIKYAIDTSEAR PIWQPPRRIPPPLLEGVNCLVNEMINDNVIRPSKSPWAYPIALVKKSDGSLQLCIDYRKLNDVTKG RFSTTSHQTHWIPCTRAKWFSTLDLKSDYWQVEVAETGREKTAFIVPNGLYEFQTMPFGLCNAA ATFQRLMQTALIELFPKHCIIYLDDILVFGKDTQEHNANLKLVLDRLRDAGLTLDPKKCHFLQRS VTFLRHTVLSDGMAVTEDRTNRVRTWPTPTNQTELRSFLSLANYYRRFVKNFAKIAGPLHKLTEK QAKKNFKWESEHDEAFKELKRMLCSAPILALPNFENDAPPFVLDMDASDVAVDGVLSQRDKEG REHVIAYASIRLNKKMRQRSAMERELFAIFTMVRHFKHYLIAKQSIVKTDHQALTWLRTMKEIDHS VARWYEELQQYDFTVQYRKGTTHSNTDALSRRPLSAERESGIAGTLFLSEPTRHQWRNTQSTYPD TALVCKTFLASSHKPTAEEMNSSSKTAKRIWRQWSKLTWRMKFYGIKKMPRLSKV", e jogou essa tradução no Pfam [16], e o resultado obtido foi que nessa tradução apresenta-se uma família de transcriptase reversa (RVT_1) e um domínio de RNaseH. Com isso o próximo passo seria caracterizar que tipo de família de transcriptase reversa o retrotransposon encontrado faria parte. Para tal foi construída, com auxílio do programa MEGA X e Clustal X, uma árvore filogenética com o retrotransposon encontrado e várias outras sequências traduzidas de transcriptases reversas conhecidas. A árvore foi construída pelo método de " *maximum likelihood estimation*", sendo "Perdi" o retrotransposon encontrado, o resultado obtido está descrito na imagem abaixo.

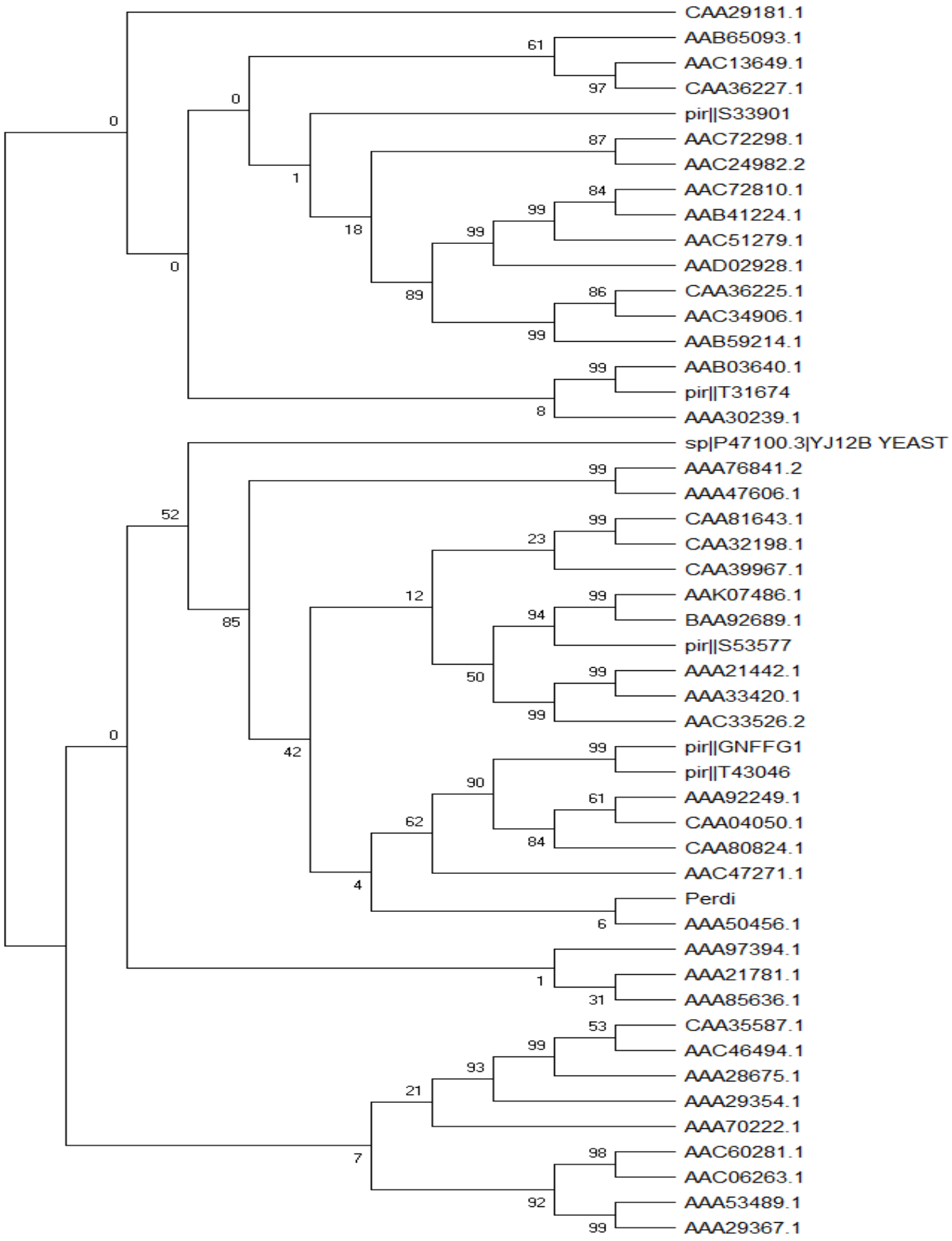

Figura 5: *Resultado obtido da árvore filogenética construída pelo método de maximum likelihood estimation.*

Como podemos observar na figura acima não é possível agrupar o domínio de transcriptase reversa do retrotransposon obtido com os das famílias selecionadas pois o valor de corte é muito pequeno, para poder chegar a uma conclusão positiva esperava-se uma equivalência de no mínimo 90. Para corroborar tal afirmação, foi feita uma segunda árvore, dessa vez com o método de neighbor joining e o resultado obtido está descrito na figura abaixo.

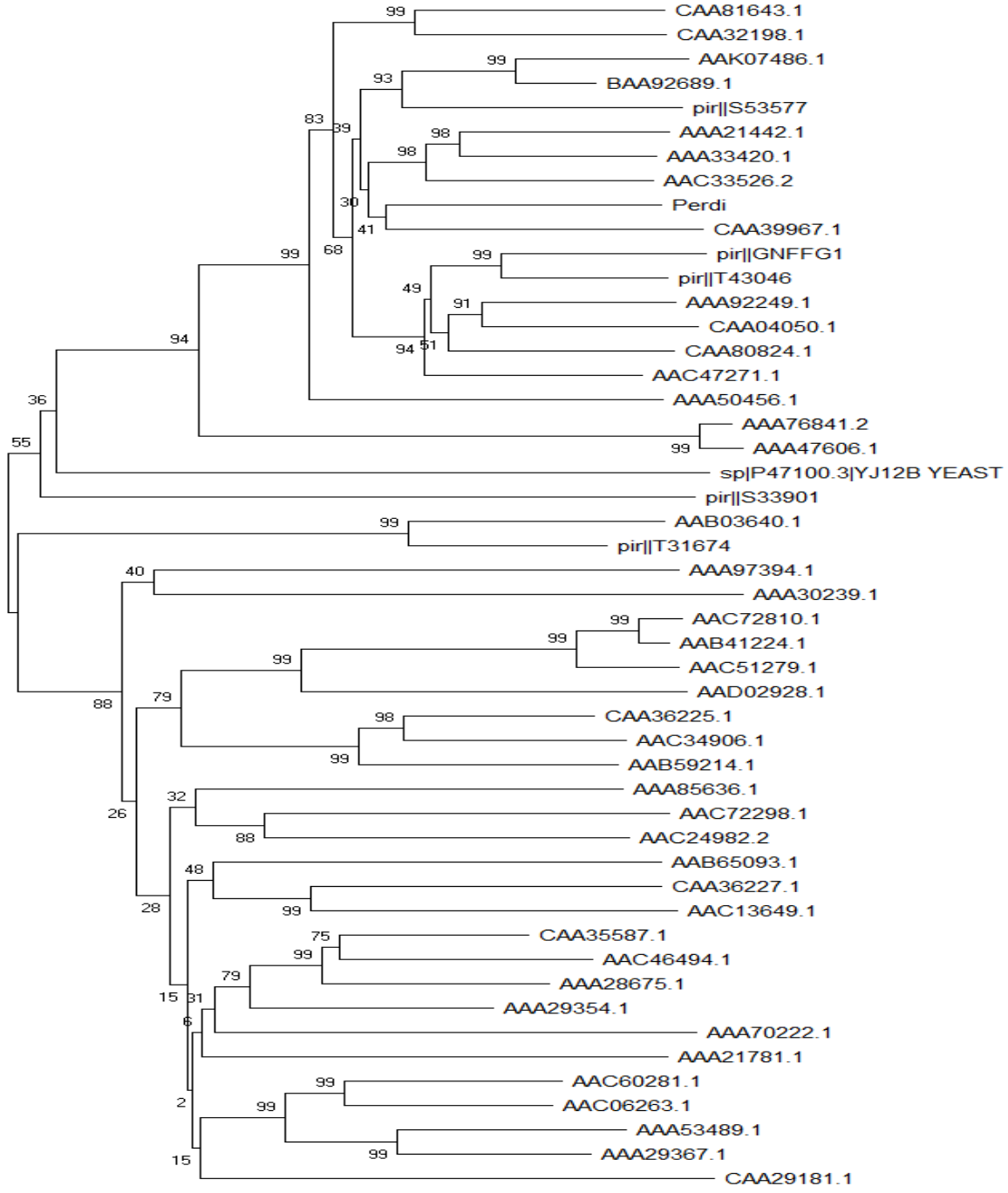

Figura 6: *Árvore filogenética construída pelo método de neighbor joining*

Como pode se notar o valor de bootstrap mais uma vez não consegue classificar satisfatoriamente, sendo este muito baixo.

4.5. Gráfico de frequência de região codificante

Uma outra análise interessante de se fazer é a frequência com que as regiões codificantes do retrotransposon encontrado aparecem, o resultado obtido do processo descrito em 3.5. foi:

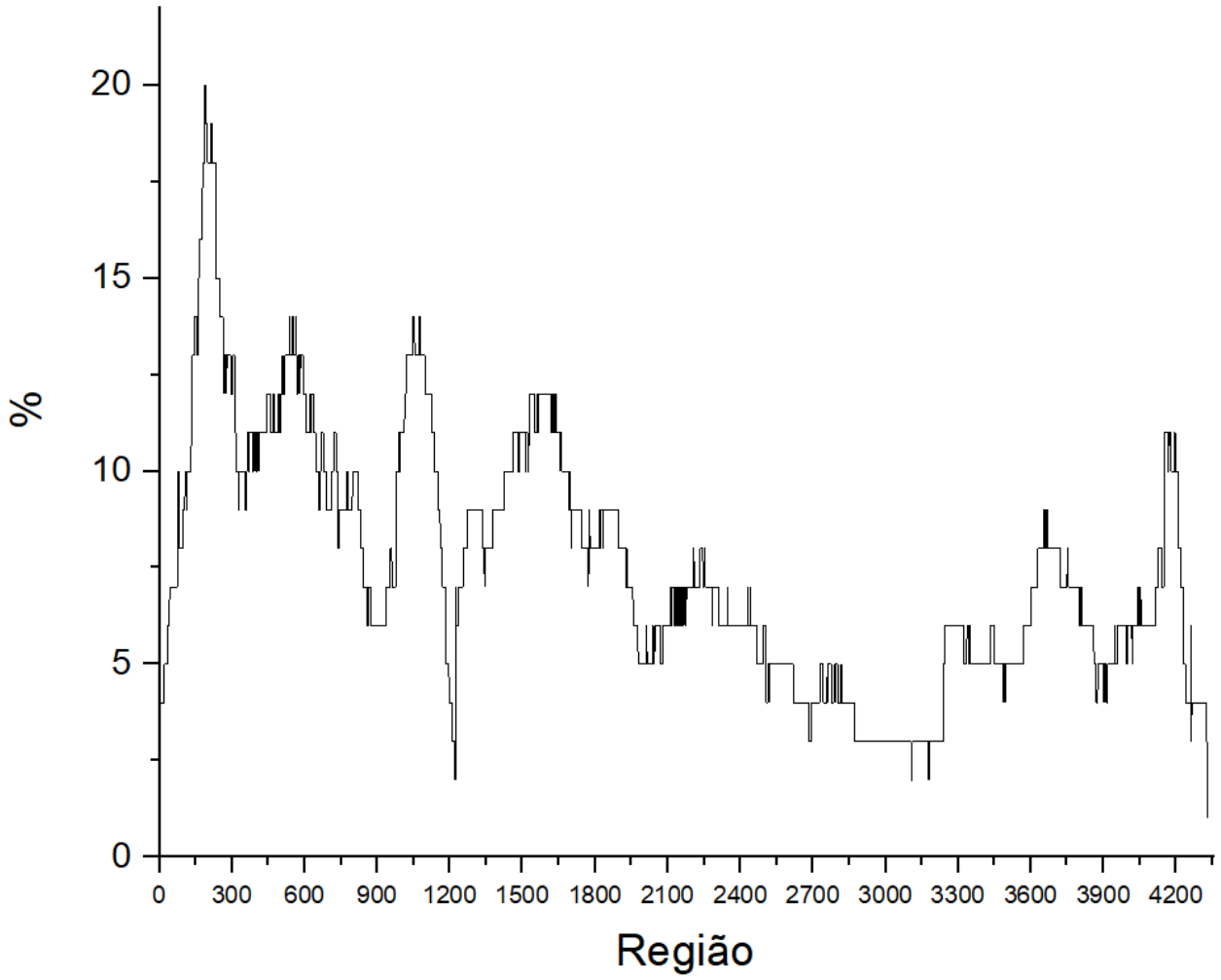

Figura 7: *Gráfico obtido como resultado do programa descrito em 3.5*

4.6. Identificação se o transposon é interno, externo ou sobreposto

Ao rodar o programa descrito em 3.6 o que obtém-se como resultado que 557 transposons, de um total de 1705, são intergênicos.

5. Conclusão

Este trabalho de conclusão de curso descreve as atividades exercidas pelo aluno ao longo do ano com a orientação do Professor Doutor Ricardo de Marco, nele tem-se por objetivo identificar e caracterizar prováveis elementos repetíveis, transposons, no genoma do organismo *Taenia solium.* Por mais que sejam pouco conhecidos [1] a importância dos transposon é real, uma vez que podem regular mudanças no genoma dos organismos. Após as análises feitas neste trabalho pode se encontrar ao menos um transposon da classe dos retrotransposons LTR com domínios de integrases, transcriptase reversa e retropepsina. Este trabalho pode ser estendido para novas análises futuras de classificação, uma vez que foi impossível classificar filogeneticamente.

6. Referências Bibliográficas

[1] JURKA, Jerzy et al. Repetitive sequences in complex genomes: structure and evolution. Annu. Rev. Genomics Hum. Genet., v. 8, p. 241-259, 2007.

[2] PACE, John K.; FESCHOTTE, Cédric. The evolutionary history of human DNA transposons: evidence for intense activity in the primate lineage. **Genome research**, v. 17, n. 4, p. 000-000, 2007.

[3] Autor Desconhecido. European Bioinformatics Institute, **Transposase**, 2018. Disponível em: <https://www.ebi.ac.uk/interpro/potm/2006_12/Page2.htm>. Acesso em: 24 out. 2018.

[4] KAZAZIAN, H. H.; SCOTT, A. F. " Copy and paste" transposable elements in the human genome. **The Journal of clinical investigation**, v. 91, n. 5, p. 1859-1860, 1993.

[5] MUÑOZ-LÓPEZ, Martín; GARCÍA-PÉREZ, José L. DNA transposons: nature and applications in genomics. **Current genomics**, v. 11, n. 2, p. 115-128, 2010.

[6] Flutre T, Duprat E, Feuillet C, Quesneville H (2011) **Considering transposable element diversification in de novo annotation approaches.** PLoS ONE 6(1): e16526. doi:10.1371/journal.pone.0016526

[7] PRICE, Alkes L.; JONES, Neil C.; PEVZNER, Pavel A. **De novo identification of repeat families in large genomes**. Bioinformatics, v. 21, n. suppl_1, p. i351-i358, 2005.

[8] Ioanna Kalvari, Joanna Argasinska, Natalia Quinones-Olvera, Eric P. Nawrocki, Elena Rivas, Sean R. Eddy, Alex Bateman, Robert D. Finn, Anton I. **Rfam 13.0: shifting to a genome-centric resource for non-coding RNA families** PetrovNucleic Acids Research (2017) 10.1093/nar/gkx1038

[9] KALVARI, Ioanna et al. Non‐Coding RNA Analysis Using the Rfam Database. **Current protocols in bioinformatics**, p. e51, 2018.

[10] MUNOZ-MÉRIDA, ANTONIO et al. Sma3s: a three-step modular annotator for large sequence datasets. **DNA research**, v. 21, n. 4, p. 341-353, 2014.

[11] CASIMIRO‐SORIGUER, Carlos S.; MUÑOZ‐MÉRIDA, Antonio; PÉREZ‐PULIDO, Antonio J. Sma3s: a universal tool for easy functional annotation of proteomes and transcriptomes. **Proteomics**, v. 17, n. 12, p. 1700071, 2017.

[12] DEMARCO, Ricardo et al. Saci-1,-2, and-3 and Perere, four novel retrotransposons with high transcriptional activities from the human parasite Schistosoma mansoni. **Journal of Virology**, v. 78, n. 6, p. 2967-2978, 2004.

[13] Autor desconhecido. **Sequence read archive.** Disponível em: <https://trace.ncbi.nlm.nih.gov/Traces/sra/sra.cgi?>. Acesso em: 02 nov. 2018

[14] Alberts B, Johnson A, Lewis J, et al. Molecular Biology of the Cell. 4th edition. New York: Garland Science; 2002. **The RNA World and the Origins of Life**. Available from: https://www.ncbi.nlm.nih.gov/books/NBK26876/

[15] ROMBEL, Irene T. et al. **ORF-FINDER: a vector for high-throughput gene identification.** Gene, v. 282, n. 1, p. 33-41, 2002.

[16] Robert D. Finn, Penelope Coggill, Ruth Y. Eberhardt, Sean R. Eddy, Jaina Mistry, Alex L. Mitchell, Simon C. Potter, Marco Punta, Matloob Qureshi, Amaia Sangrador-Vegas, Gustavo A. Salazar, John Tate, Alex Bateman; **The Pfam protein families database: towards a more sustainable future**, *Nucleic Acids Research*, Volume 44, Issue D1, 4 January 2016, Pages D279–D285, https://doi.org/10.1093/nar/gkv1344