Awk one-liners for blast results manipulation Cheat Sheet

by melissamlwong via cheatography.com/22270/cs/5331/

Introduction

This cheatsheet contains 10 useful AWK one-liners for tab delimited blast results. It is created as part of a series to help graduate students and biologists in learning some simple programming scripts. Each oneliner is usually accompanied by additional comments which start with a hash ("#"). Runnabble codes is available on http://code.runnable.com/VfltWNXUYTcrUkwn/10-awk-one-liners-for-blast-results-manipulation-for-shell-bash-and-bioinformatics

Author: Melissa M.L. Wong; Date created: 6 Aug 2015; Date last modified:21 April 2016; Email: melissawongukm@gmail.com

Tab delimited blast results is a text-based files to show pairwise alignment between two sequences. It is generated using the option "-outfmt 6" or "-m 8". Each column is separated by a tab and represents queryId($1), subjectId($2), percIdentity($3), alnLength($4), mismatchCount($5), gapOpenCount($6), queryStart($7), queryEnd($8), subjectStart($9), subjectEnd($10), eValue($11) and bitScore($12) respectively

1. To filter alignment

```
awk '$1~/Medtr1g006460.1/' temp.blast #matching query name
awk '$2~/M edtr0/' temp.blast #matching reference name
awk '$12>= 1000' temp.blast #score
awk '$3>=80' temp.blast #identity percentage
awk '$11<1 e-30' temp.blast #e-value
```

2. To filter all against all blast results

```
#method 1 - remove blast results of the same sequence and apply filtering
blastn -task megablast -db database1 -query temp.fa -evalue 1E-10 -outfmt 6 | awk '$1!=$2 && $3>=40 &&
$4> =300'
#method 2 - remove blast results of the same sequence and apply filtering
blastn -task megablast -db database1 -query temp.fa -evalue 1E-10 -outfmt 6 | awk '{spli t($ 1,a ,".");
split( $1, b,"." ); if (a[1]! =b[1] && $3>=40 && $4> =300) print }'
#method 3 - remove redundant alignm ents. Any alignment in all-ag ain st-all blast can appear twice as
seq1\tseq2 and seq2\t seq1. Both alignments can sometimes vary in length by 1-2 bp, however, they always
share the same score.
awk '{c=$1 " \t"$ 2"\t "$12 ; b= $2" \t"$1 " \t"$12; if ($1!=$2 && a[c]==0 && a[b]==0) a[$1"\t " $2" -
\t"$1 2]= $0} END{for (i in a) print a[i]}' temp.txt > temp.blast #not so working well
```

3. To filter alignments based on sequence length

```
#method 1 - calculate sequence length, calculate percentage of alignment length against sequence length,
filter blast file
awk 'BEGIN {RS =">"; FS= " \n"} NR> 1{s eq= " ";for (i=2;i <=N F;i++) seq=se q""$i; print $1" \t"le ngt -
h(s eq)}' temp.fa > len1
awk 'NR==F NR{ a[N R]= $1" \t"$2 " \t"$ 4;d [NR ]=$ 0;s um+ =1} NR> FNR {b[ $1] =$2 }EN D{for (i=1;i <=s -
um;i++) {split (a[ i], c," \t"); if (c[3]/ b[c [1] ]>=0.8 && c[3]/b [c[ 2]] >=0.8) print d[i]}}'
temp.blast len1 len1
#method 2 - if length inform ation is included in fasta header
awk '{spli t($ 1,a ,"_"); split( $1, b,"_ "); c=a[2] ;d= b[2]; if ($4/c>=0.8 && $4/d>=0.8) print $0}'
temp.blast #if length in header and separated by " _"
```

4. To count the number of queries

```
awk '! a[$1]++' temp.blast | wc -l
awk '{a[$1 ]++ }EN D{for (i in a) sum+=1; print sum}' temp.blast #equiv alent script but faster
```

5. To count the number of alignments per query

```
awk '{a[$1]++}END{for (i in a) print i"\t"a[i]}' temp.blast
```

6. To find best hit for a query

```
#method 1 - Use the first alignment per sequence assuming the best hit is always listed first
awk '! a[$1]++' temp.blast
#method 2 - Use total score assuming each query can have multiple alignments to a reference sequence. In my opinion, this is the best way except in cases where multiple alignments to the same region of a pair of query and reference are reported.
awk '{b[$1 ]="0 "; e[$1]= " ";if (a[$1, $2] =="0 ") a[$1,$ 2]=$12; else {score =a[ $1, $2] +$12; a[$1,$ -
2]= sco re} }EN D{for (i in b) for (j in a) {split (j, c,S UBSEP); if (c[1]==i && a[j]>b[i]) {b[i]= -
a[j ];e [i] =c[ 2]}}; for (i in b) print i"\t " e[i ]"\t " b[i]}' temp.blast
```

7. To find reciprocal best hit for a query

```
#An extension of the finding best hit script by making sure that a query is a reference's best hit and vice versa
awk '{a[$1 ]="0 " ;b[ $1] ="";c [$2 ]="0 " ;d[ $2] ="";if (e[$1, $2]==0) e[$1,$ 2]=$12; else {score =e[ -
$1, $2] +$12; e[$1,$ 2]= sco re} }EN D{for (i in a) for (j in e) {split (j, f,S UBSEP); if (f[1]==i &&
e[j]>a[i]) {a[i]= e[j ];b [i] =f[ 2]}}; for (i in c) for (j in e) {split (j, f,S UBSEP); if (f[2]==i &&
e[j]>c[i]) {c[i]= e[j ];d [i] =f[ 1]}}; for (i in b) if (b[i] in d && d[b[i] ]==i) print i"\t " -
b[i ]"\t " a[i ]"\t " c[b [i]]}' temp.blast #need to debug
```

8. To extract one seqeunce

```
awk 'NR==FNR{if ($1~/Medtr1g006460.1/) a[$1]++}NR>FNR{if ($1 in a && $1!="") printf ">%s\n",$0}' RS="\n"
FS="\t" temp.blast RS=">" FS="\n" temp.fa
```

9. To reduce blast file size

```
#replace unnecessary columns by replacing them with empty string. For example, we are only interested in the query name, reference name and score.
awk '{print $1" \t"$2 " \t \t\t \t \t\t \t \t\t \t"$12}' temp.blast
```

10. To list all hits for each reference sequence

```
awk '{a[$1]++;b[$1,$2]++}END{for (i in a) {printf "%s", i; for (j in b) {split(j,c,SUBSEP); if (c[1]==i)
printf " %s", c[2]};printf "\n"}}' temp.blast
```