Clustering Matches using the program matchcluster Manual

Stefan Kurtz*

August 4, 2014

This short manual describes the program *matchcluster* which allows to cluster matches if they contain pairwise similar sequences, or if their positions are close together.

1 The program *matchcluster* and its options

The program is called as follows:

matchcluster options matchfile

And here is a description of the options:

`-erate` ϵ
: Specify the error percentage ϵ for clustering by sequence similarity. ϵ is an integer in the range 0 to 100.

`-gapsize` γ
: Specify the maximum gap size γ for clustering by gap size. γ is an non-negative integer.

`-overlap` ω
: Specify the overlap percentage ω for clustering by overlap. ω is an non-negative integer.

`-outprefix` *prefix*
: Specify that each cluster of matches is output into a separate file whose name starts with *prefix*.

*Zentrum für Bioinformatik, Universität Hamburg, Bundesstrasse 43, 20146 Hamburg, Germany, E-mail: kurtz@zbh.uni-hamburg.de

`-version`

Show the version of the *Vmatch* software package, the program is part of. Also report the compilation date and the compilation options.

`-help`

Show a summary of all options and terminate.

The option `-outprefix` is mandatory. Also exactly one of the options `-erate`, `-gapsize`, and `-overlap` must be used. *matchcluster* reads in the match file and applies single linkage clustering to the matches. The choice of the parameter `-erate`, `-gapsize`, and `-overlap` determines which kind of clustering is performed.

Each cluster is reported in a separate file named *prefix*`.`s`.`i`.match`. This stores all matches which were used to form cluster i. s is the size of the cluster with number i. The cluster numbers are consecutive numbers beginning with 0. The matches are reported in the standard `vmatch` format. Furthermore, each match is preceeded by a comment line showing the identification number of the match. At the end of the file, it is reported why the cluster was formed.

2 Examples

Suppose we have constructed an index for yeast chromosome III, using the program `mkvtree`:

```
$ mkvtree -dna -db ychrIII.fna -v -tis -ois -bwt -suf -lcp
reading file "ychrIII.fna"
total length of sequences: 315339
create file "ychrIII.fna.tis"
create file "ychrIII.fna.ois"
create file "ychrIII.fna.des"
create file "ychrIII.fna.sds"
create file "ychrIII.fna.lcp"
initializing data structures
sorting suffixes
create file "ychrIII.fna.llv"
create file "ychrIII.fna.suf"
create file "ychrIII.fna.bwt"
create file "ychrIII.fna.prj"
create file "ychrIII.fna.al1"
overall space peak: main=2.82 MB (9.38 bytes/symbol), secondary=0.31 MB
```

Now compute all repeats of length ≥ 100 in yeast chromosome III, using the program `vmatch`.

```
$ vmatch -l 100 ychrIII.fna
# args=-l 100 ychrIII.fna
  203    0 199591   D   203    0 293111   0   1.69e-112    406   100.00
  120    0  13690   D   120    0 293111   0    1.58e-62    240   100.00
```

```
  608    0  12324   D   608    0 291850   0    0.00e+00   1216   100.00
  608    0 198225   D   608    0 291850   0    0.00e+00   1216   100.00
 1624    0  12186   D  1624    0 198087   0    0.00e+00   3248   100.00
  117    0 123938   D   117    0 142657   0    1.01e-60    234   100.00
  195    0  13811   D   195    0 199712   0   1.11e-107    390   100.00
  280    0  83954   D   280    0  84469   0   7.41e-159    560   100.00
  239    0  83954   D   239    0  90099   0   3.58e-134    478   100.00
  210    0 267362   D   210    0 267689   0   1.03e-116    420   100.00
  276    0  11909   D   276    0 197810   0   1.90e-156    552   100.00
  143    0  11765   D   143    0 197666   0    2.25e-76    286   100.00
  286    0  84422   D   286    0  90052   0   1.81e-162    572   100.00
  126    0  83680   D   126    0  90052   0    3.86e-66    252   100.00
  126    0  83680   D   126    0  84422   0    3.86e-66    252   100.00
  266    0  11498   D   266    0 197399   0   1.99e-150    532   100.00
```

Suppose we have stored these matches in a matchfile `ychrIII-100.match`. The following program call performs clustering by similarity, using an error rate 35, producing clusters in files with the prefix `clout`:

```
$ matchcluster -erate 35 -outprefix clout ychrIII-100.match
# file=/vol/biodata/DNA-mix/Grumbach.fna/ychrIII.fna 320531 315339
# databaselength=315338
# alphabet "aAcCgGtTuUnsywrkvbdhmNSYWRKVBDHM" (size 32) mapped to "acgtn" (size 5)
# ychrIII.fna.tis read
# ychrIII.fna.ois read
# ychrIII.fna.des read
# ychrIII.fna.sds read
# read self matches
# cluster 16 matches
# create cluster 0 of size 2
# create cluster 1 of size 3
# create cluster 2 of size 2
```

Three files where generated by this program call:

```
$ ls -l clout.*
-rw-r-----    1 kurtz    gistaff       285 2005-02-21 16:41 clout.2.0.match
-rw-r-----    1 kurtz    gistaff       289 2005-02-21 16:41 clout.2.2.match
-rw-r-----    1 kurtz    gistaff       491 2005-02-21 16:41 clout.3.1.match
```

Consider the last file `clout.3.1.match` representing cluster 1 with three matches:

```
$ cat clout.3.1.match
# args=-l 100 ychrIII.fna
# id 7
  280      0  83954   D   280      0  84469   0   7.41e-159    560   100.00
# id 8
  239      0  83954   D   239      0  90099   0   3.58e-134    478   100.00
# id 12
  286      0  84422   D   286      0  90052   0   1.81e-162    572   100.00
# linked 8 and 12 with edit distance 47 (error rate 19.67%)
# linked 7 and 12 with edit distance 88 (error rate 31.43%)
# linked 7 and 8 with edit distance 41 (error rate 17.15%)
```

Cluster 1 thus contains the matches with identification numbers 7, 8, and 12. For example, match 7 and 12 achieve a distance of 88, which corresponds to an error rate of 31.43%, well below the maximum error rate of 35%. Note that the generated files are in a format that they can be read by `vmatchselect`. This, for example, allows to report the sequence content of the matches:

```
$ vmatchselect -s leftseq clout.3.1.match
# args=-l 100 ychrIII.fna
>  239    0  83954   D   239    0  90099   0   3.58e-134    478   100.00
CTAGTATATTATCATATACGGTGTTAGAAGATGACGCAAATGATGAGAAATAGTCATCTA
AATTAGTGGAAGCTGAAACGCAAGGATTGATAATGTAATAGGATCAATGAATATTAACAT
ATAAAATGATGATAATAATATTTATAGAATTGTGTAGAATTGCAGATTCCCTTTTATGGA
TTCCTAAATCCTCGAGGAGAACTTCTAGTATATCTACATACCTAATATTATTGCCTTAT

>  280    0  83954   D   280    0  84469   0   7.41e-159    560   100.00
CTAGTATATTATCATATACGGTGTTAGAAGATGACGCAAATGATGAGAAATAGTCATCTA
AATTAGTGGAAGCTGAAACGCAAGGATTGATAATGTAATAGGATCAATGAATATTAACAT
ATAAAATGATGATAATAATATTTATAGAATTGTGTAGAATTGCAGATTCCCTTTTATGGA
TTCCTAAATCCTCGAGGAGAACTTCTAGTATATCTACATACCTAATATTATTGCCTTATA
AAAAATGGAATCCCAACAATTACATCAAAATCCACGTTCT

>  286    0  84422   D   286    0  90052   0   1.81e-162    572   100.00
TGTTGGAATAAAAATCAACTATCATCTACTAACTAGTATTTACGTTACTAGTATATTATC
ATATACGGTGTTAGAAGATGACGCAAATGATGAGAAATAGTCATCTAAATTAGTGGAAGC
TGAAACGCAAGGATTGATAATGTAATAGGATCAATGAATATTAACATATAAAATGATGAT
AATAATATTTATAGAATTGTGTAGAATTGCAGATTCCCTTTTATGGATTCCTAAATCCTC
GAGGAGAACTTCTAGTATATCTACATACCTAATATTATTGCCTTAT
```

Alternatively, we cluster by gap size, allowing gaps of size up to 1000.

```
$ matchcluster -gapsize 1000 -outprefix clout ychrIII-100.match
# file=/vol/biodata/DNA-mix/Grumbach.fna/ychrIII.fna 320531 315339
# databaselength=315338
# alphabet "aAcCgGtTuUnsywrkvbdhmNSYWRKVBDHM" (size 32) mapped to "acgtn" (size 5)
# ychrIII.fna.tis read
# ychrIII.fna.ois read
# ychrIII.fna.des read
# ychrIII.fna.sds read
# read self matches
# cluster 16 matches
# create cluster 0 of size 9
# create cluster 1 of size 4
```

Consider the file `clout.4.1.match` representing cluster 1 with four matches: It contains the matches with identification numbers 14, 7, 8, and 12. For example, match 8 and 7 have a gap of $276 = 84469 - 83954 - 239$ which is well below the maximum gap size of 1000.

```
$ cat clout.4.1.match
# args=-l 100 ychrIII.fna
# id 14
```

```
  126      0  83680   D   126      0  84422   0    3.86e-66    252   100.00
# id 7
  280      0  83954   D   280      0  84469   0   7.41e-159    560   100.00
# id 8
  239      0  83954   D   239      0  90099   0   3.58e-134    478   100.00
# id 12
  286      0  84422   D   286      0  90052   0   1.81e-162    572   100.00
# linked 8 and 7 with gapsize 276
# linked 8 and 14 with gapsize 229
# linked 8 and 12 with gapsize 229
# linked 14 and 7 with gapsize 663
# linked 14 and 12 with gapsize 616
# linked 14 and 8 with gapsize 148
# linked 14 and 7 with gapsize 148
```

In a third run we cluster the matches by overlap, allowing overlaps of minimum 10%.

```
$ matchcluster -overlap 10 -outprefix clout ychrIII-100.match
# file=/vol/biodata/DNA-mix/Grumbach.fna/ychrIII.fna 320531 315339
# databaselength=315338
# alphabet "aAcCgGtTuUnsywrkvbdhmNSYWRKVBDHM" (size 32) mapped to "acgtn" (size 5)
# ychrIII.fna.tis read
# ychrIII.fna.ois read
# ychrIII.fna.des read
# ychrIII.fna.sds read
# read self matches
# cluster 16 matches
# create cluster 0 of size 3
# create cluster 1 of size 5
# create cluster 2 of size 3
```

Consider the file `clout.3.2.match` representing cluster 2 with three matches: It contains the matches with identification numbers 0, 6, and 1. For example, match 0 and 1, overlap by $82 = 199591 + 203 - 199712$, which is 40.39% of the longer match 203.

```
$ cat clout.3.2.match
# args=-l 100 ychrIII.fna
# id 0
  203      0 199591   D   203      0 293111   0   1.69e-112    406   100.00
# id 6
  195      0  13811   D   195      0 199712   0   1.11e-107    390   100.00
# id 1
  120      0  13690   D   120      0 293111   0    1.58e-62    240   100.00
# linked 0 and 1 with overlap percentage 100.00
# linked 0 and 6 with overlap percentage 40.39
```

A Single Linkage Clustering

Initially, the single linkage clustering strategy puts each match into its own cluster. In every clustering step, each match is in exactly one cluster. To compute the clusters, for each pair

(m, m') of matches it is verified, if the matches are related. Whenever two matches m and m' are related, the two clusters containing m and m' are joint to one cluster. The different clustering strategies differ by what they consider to be related sequences. There are basically two notions of relatedness: One is sequence based and the other is position based.

Clustering by sequence similarity For this clustering strategy, the user specifies an error rate ϵ and two matches m and m' are related if the sequences involved in the matches are sufficiently similar: To precisely define this, we have to refer to the sequences involved in a pairwise match. So let a match be a pair (u, v) of match instances, i.e. u and v are matching sequences. Let $m = (u, v)$ and $m' = (u', v')$ be the matches for which we want to check relatedness alias similarity. Let l be the minimal length of all four sequences u, v, u', and v'. Then $e = l \cdot \epsilon/100$ is the error threshold for this pair (m, m') of matches. Let $edist(x, y)$ denote the edit distance between two sequences x and y, i.e. the minimal number of insertions, deletions, and replacements required to transform sequence x into sequence y. We consider m and m' to be similar, if and only if at least one of the following four conditions holds:

- $edist(u, u') \leq e$
- $edist(u, v') \leq e$
- $edist(v, u') \leq e$
- $edist(v, v') \leq e$

Clustering by match positions There are two strategies of clustering by positions. These were originally described and evaluated by [1] in the context of repeat clustering.

- clustering by *gap size*: in this case the user specifies a parameter $\gamma \geq 0$, the maximal size of a gap.
- clustering by *overlap*: in this case the user specifies a parameter $\omega \in [0, 100]$, the overlap percentage.

Both clustering strategies ignore the sequence content of the matches and only consider the positions where the matches occur. Therefore, now consider a match to be a quadruple (l, i, r, j), where l and r are the length of the matching sequences and i and j are their start positions. Let M be the set of all matches. We mirror M to obtain a set P of partition points, defined as follows:

$$P = M \cup \{(r, j, l, i) \mid (l, i, r, j) \in M\}$$

Now sort the partition points in P according to their second component. Partition points with identical second components are sorted according to their fourth component. Now consider two partition points $p = (l, i, r, j)$ and $p' = (l', i', r', j')$ from P such that $i \leq i'$. We define

$d(p, p') = \max(0, i' - i - l)$ to be the gap size between p and p'. When clustering with gap size, we consider p and p' to be related, if $d(p, p') \leq \gamma$. We also define $o(p, p') = \max(0, i + l - i')$ to be the overlap size of p and p. The overlap percentage is relative to the length of the longer sequence, i.e. it is defined as $\frac{o(p,p')}{\max(l,l')}$. When clustering with overlap is used, we consider p and p' to be related, if $100 \cdot \frac{o(p,p')}{\max(l,l')} \geq \omega$.

References

[1] N. Volfovsky, B.J. Haas, and S.L. Salzberg. A Clustering Method for Repeat Analysis in DNA Sequences. *Genome Biology*, 2(8):research0027.1–0027.11, 2001.