NAME

clm dist − compute the distance between two or more partitions (clusterings).

The distance that is computed can be any of *split/join distance*, *variance of information*, or *Mirkin metric*.

clmdist is not in actual fact a program. This manual page documents the behaviour and options of the clm program when invoked in mode *dist*. The options **-h**, **--apropos**, **--version**, **-set**, **--nop** are accessible in all **clm** modes. They are described in the **clm** manual page.

SYNOPSIS

clm dist [options] <file name> <file name>+

clm dist [-mode <sj|vi|mk|sc> (*distance type*)**] [-o** fname (*output file*)**] [--chain** (*only compare consecutive clusterings*)**] [--one-to-many** (*compare first clustering to all others*)**] [--sort** (*sort clusterings based on coarseness*)**] [--index** (*output Rand, adjusted Rand and Jaccard indices*)**] [-digits** k (*output decimals*)**] [-h** (*print synopsis, exit*)**] [--apropos** (*print synopsis, exit*)**] [--version** (*print version, exit*)**]** <file name> <file name>+

DESCRIPTION

clm dist computes distances between clusterings. It can compute the *split/join distance* (described below), the *variance of information measure*, and the *Mirkin metric*. By default it computes the chosen distance for all pairs of distances in the clusterings provided. Clusterings must be in the mcl matrix format (cf. **mcxio(5)**), and are supplied on the command line as the names of the files in which they are stored. It is possible to compare only consecutive clusterings by using the **--chain** option.

Currently, **clm dist** cannot compute different distance types simultaneously.

The output is linewise, each line giving information about the distance between a pair of clusterings. A line has the following format:

```
d  d1  d2  N  v  name1  name2  [v50,v75,v90,v95,v99]
```

where `dT` is the distance between the two clusterings, `d1` is the distance from the first clustering to the greatest common subclustering (alternatively called GCS, intersection, or meet) of the two clusterings, `d2` is similarly the distance from the second clustering to the GCS, `N` is the number of nodes in the set over which the clusterings are defined, `name1` is the name of the file containing the first clustering, `name2` is the name of the file containing the second clustering, and `vXX` is the number of *volatile nodes* at stringency factor `0.XX` (i.e. 0.5 for `v50`). Refer to **clm vol** for a definition of *volatile node*.

OPTIONS

-mode <sj|vi|mk> (*distance type*)

Use **sj** for the *split/join distance* (described below), **vi** for the *variance of information measure* and **mk** for the *Mirkin metric*.

--chain (*only compare consecutive clusterings*)

This option can be used if you know that the clusterings are nested clusterings (or appoximately so) and ordered from coarse to fine-grained or vice versa. An example of this is the set of clusterings resulting from applying **mcl** with a range of inflation parameters.

--one-to-many (*compare first clustering to all others*)

Use this option for example to compare a gold standard classification to a collection of clusterings. Bear in mind that sub-clustering and super-clustering are also ways for a clustering to be compatible with a gold standard. This means that the simple numerical criterion of distance between clusters (by whatever method) is only partially informative. For the Mirkin, variation of information and split/join metrics it pays to take into account the constituent distances `d1` and `d2` (see above). Assuming that the first clustering given as argument represents a gold standard, a small value for `d1` implies that the second clustering is (nearly) a superclustering, and similarly a small value for `d2` implies that it is (nearly) a subclustering.

--sort (*sort clusterings based on coarseness*)
This option can be useful in conjunction with the **--chain** option, in case the list of clusterings supplied is not necessarily ordered by granularity.

--index (*output Rand, adjusted Rand and Jaccard indices*)
As described.

-o fname (*output file*)

-digits k (*output decimals*)
The number of decimals printed when using the variance of information measure.

SPLIT/JOIN DISTANCE

For each pair of clusterings **C1**, **C2**, two numbers are given, say **d1** and **d2**. Then **d1** + **d2** equals the number of nodes that have to be exchanged in order to transform any of the two clusterings into the other, and you can think of (**d1+d2**)/**2N** as the percentage that the two clusterings differ. The split/join distance has a linearity property with respect to the meet of **C1** and **C2**, see below.

The split/join distance **sjd** is very handy in computing the consistency of two or more clusterings of the same graph, or comparing clusterings made with different resource (but otherwise identical) parameters. The latter is for finding out whether you can settle for cheaper mcl settings, or whether you need to switch to more expensive settings. The former is for finding out whether clusterings are identical, conflicting, or whether one is (almost) a subclustering of the other - mostly for comparing a set of clusterings of different granularity, made by letting the mcl parameter **-I** vary. The **EXAMPLES** section contains examples of all these **clm dist** uses, and the use of **clm info** and **clm meet** is also discussed there.

sjd is a metric distance on the space of partitions of a set of a given fixed cardinality. It has the following linearity property. Let **P1** and **P2** be partitions, then

sjd(**P1**, **P2**) = **sjd**(**P1**, **D**) + **sjd**(**P2**, **D**)

where **D** (for Dutch Doorsnede) is the intersection of **P1** and **P2**, i.e. the unique clustering that is both a subclustering of **P1** and **P2** *and* a superclustering of all other subclusterings of **P1** and **P2**. Sloppily worded, **D** is the largest subclustering of both **P1** and **P2**. See the **REFERENCES** section for a pointer to the technical report in which **sjd** was first defined (and in which the non-trivial triangle inequality is proven).

Because it is useful to know whether one partition (or clustering) is almost a subclustering of the other, **clm dist** returns the two constituents **sjd**(**P1**,**D**) and **sjd**(**P2**,**D**).

Let **P1** and **P2** be two clusterings of a graph of cardinality **N**, and suppose **clm dist** returns the integers **d1** and **d2**. You can think of **100 * (d1 + d2) / N** as the percentage that **P1** and **P2** differ. This interpretation is in fact slightly conservative. The numerator is the number of nodes that need to be exchanged in order to transform one into the other. This number may grow as large as **2*N - 2*sqrt(N)**, so it would be justified to take 50 as a scaling factor rather than 100.

For example, if **A** and **B** are both clusterings of a graph on a set of 9058 nodes and **clm dist** returns [38, 2096], this conveys that **A** is almost a subclustering of **B** (by splitting 38 nodes in **A** we obtain a clustering **D** that is a subclustering of **B**), and that **B** is much less granular than **A**. The latter is because we can obtain **B** from **D** by *joining* 2096 nodes in some way.

EXAMPLES

The following is an example of several mcl validation tools applied to a set of clusterings on a protein graph of 9058 nodes. In the first experiment, six different clusterings were generated for different values of the inflation parameter, which was respectively set to 1.2, 1.6, 2.0, 2.4, 2.8, and 3.2. It should be noted that protein graphs seem somewhat special in that an inflation parameter setting as low as 1.2 still produces a very acceptable clustering. The six clusterings are scrutinized using **clm dist**, **clm info**, and **clm meet**. In the second experiment, four different clusterings were generated with identical flow (i.e. inflation) parameter, but with different resource parameters. **clm dist** is used to choose a sufficient resource level.

High **-P/-S/-R** values make **mcl** more accurate but also more time and memory consuming. Run **mcl** with different settings for these parameters, holding other parameters fixed. If the expensive and supposedly more accurate clusterings are very similar to the clusterings resulting from cheaper settings, the cheaper setting is sufficient. If the distances between cheaper clusterings and more expensive clusterings are large, this is an indication that you need the expensive settings. In that case, you may want to increase the **-P/-S/-R** parameters (or simply the **-scheme** parameter) until associated clusterings at nearby resource levels are very similar.

In this particular example, the validation tools do not reveal that one clustering in particular can be chosen as 'best', because all clusterings seem at least acceptable. They do aid however in showing the relative merits of each clusterings. The most important issue in this respect is cluster granularity. The table below shows the output of **clm info**.

```
     Efficiency  Mass frac  Area frac  Cl weight  Mx link weight
1.2   0.42364     0.98690    0.02616    52.06002    50.82800
1.6   0.58297     0.95441    0.01353    55.40282    50.82800
2.0   0.63279     0.92386    0.01171    58.09409    50.82800
2.4   0.65532     0.90702    0.01091    59.58283    50.82800
2.8   0.66854     0.84954    0.00940    63.19183    50.82800
3.2   0.67674     0.82275    0.00845    66.10831    50.82800
```

This data shows that there is exceptionally strong cluster structure present in the input graph. The 1.2 clustering captures almost all edge mass using only 2.5 percent of 'area'. The 3.2 clustering still captures 82 percent of the mass using less than 1 percent of area. We continue with looking at the mutual consistency of the six clusterings. Below is a table that shows all pairwise distances between the clusterings.

	1.6	2.0	2.4	2.8	3.2	3.6
1.2	2096,38	2728,41	3045,48	3404,45	3621,43	3800, 42
1.6		797,72	1204,76	1638,78	1919,70	2167, 69
2.0			477,68	936,78	1235,85	1504, 88
2.4				498,64	836,91	1124,103
2.8					384,95	688,119
3.2						350,110

The table shows that the different clusterings are pretty consistent with each other, because for two different clusterings it is generally true that one is almost a subclustering of the other. The interpretation for the distance between the 1.6 and the 3.2 clustering for example, is that by rearranging 43 nodes in the 3.2 clustering, we obtain a subclustering of the 1.6 clustering. The table shows that for any pair of clusterings, at most 119 entries need to be rearranged in order to make one a subclustering of the other.

The overall consistency becomes all the more clear by looking at the meet of all the clusterings:

```
clm meet -o meet out12 out16 out20 out24 out28 out32
clm dist meet out12 out16 out20 out24 out28 out32
```

results in the following distances between the respective clusterings and their meet.

	1.2	1.6	2.0	2.4	2.8	3.2
meet	0,3663	0,1972	0,1321	0,958	0,559	0,283

This shows that by rearranging only 283 nodes in the 3.2 clustering, one obtains a subclustering of all other clusterings.

In the last experiment, **mcl** was run with inflation parameter 1.4, for each of the four different preset pruning schemes `k=1,2,3,4`. The **clm dist** distances between the different clusterings are shown below.

	k=2	k=3	k=4
k=1	17,17	16,16	16,16
k=2		3,3	5,5
k=3			4,4

This example is a little boring in that the cheapest scheme seems adequate. If anything, the gaps between the `k=1` scheme and the rest are a little larger than the three gaps between the `k=2`, `k=3`, and `k=4` clusterings. Had all distances been much larger, then such an observation would be reason to choose the `k=2` setting.

Note that you need not feel uncomfortable with the clusterings still being different at high resource levels, if ever so slightly. In all likelihood, there are anyway nodes which are not in any core of attraction, and that are on the boundary between two or more clusterings. They may go one way or another, and these are the nodes which will go different ways even at high resource levels. Such nodes may be stable in clusterings obtained for lower inflation values (i.e. coarser clusterings), in which the different clusters to which they are attracted are merged.

AUTHOR

Stijn van Dongen.

SEE ALSO

mclfamily(7) for an overview of all the documentation and the utilities in the mcl family.

REFERENCES

Stijn van Dongen. *Performance criteria for graph clustering and Markov cluster experiments*. Technical Report INS-R0012, National Research Institute for Mathematics and Computer Science in the Netherlands, Amsterdam, May 2000.
http://www.cwi.nl/ftp/CWIreports/INS/INS-R0012.ps.Z

Marina Meila. *Comparing Clusterings − An Axiomatic View*. In *Proceedings of the 22nd International Conference on Machine Learning*, Bonn, Germany, 2005.

Marina Meila. *Comparing Clusterings*, UW Statistics Technical Report 418.
http://www.stat.washington.edu/www/research/reports/2002/tr418.ps