



## Clustering of highly homologous sequences to reduce the size of large protein databases

Weizhong Li<sup>1</sup>, Lukasz Jaroszewski<sup>2</sup> and Adam Godzik<sup>2,\*</sup>

<sup>1</sup>San Diego Supercomputer Center, La Jolla, CA 92093, USA and <sup>2</sup>The Burnham Institute, La Jolla, CA 92037, USA

Received on October 4, 2000; revised on November 1, 2000; accepted on November 6, 2000

### ABSTRACT

**Summary:** We present a fast and flexible program for clustering large protein databases at different sequence identity levels. It takes less than 2 h for the all-against-all sequence comparison and clustering of the non-redundant protein database of over 560 000 sequences on a high-end PC. The output database, including only the representative sequences, can be used for more efficient and sensitive database searches.

**Availability:** The program is available from <http://bioinformatics.burnham-inst.org/cd-hi>

**Contact:** [liwz@sdsc.edu](mailto:liwz@sdsc.edu) or [adam@burnham-inst.org](mailto:adam@burnham-inst.org)

### INTRODUCTION

Redundancy is very common in sequence databases. Since these redundant data often do not supply more information, for many purposes it is practical to consider only a set of representatives from the complete database. For example, a non-redundant protein database (NR), where proteins with identical sequences are combined to single entries, is compiled at NCBI for BLAST (Altschul *et al.*, 1997) server. The same database clustered at 90% threshold was introduced by Holm and Sander (1998).

Database clustering, which requires a large amount of pairwise alignments, is time consuming. The widely available program for proteins sequence database clustering, used to create nrdb90 (Holm and Sander, 1998), needs about 1 day to process NR database and it is effective at threshold levels around 90% identity. The program we present here is faster, more flexible, and works at lower sequence identity thresholds.

### METHODS, ALGORITHMS AND IMPLEMENTATION

We use the same greedy incremental algorithm as implemented by Holm and Sander (1998). Briefly, sequences are first sorted in order of decreasing length. The longest one becomes the representative of the first cluster. Then each

remaining sequence is compared to the existing representatives. If the similarity with any representative is above a given threshold, it is included into that cluster. Otherwise a new cluster starts with it as representative.

Here, short word filters are applied to reduce the number of pairwise alignments, the most time consuming element of the clustering procedure. Two proteins sharing certain sequence identity should have at least a certain number of identical dipeptides, tripeptides and so on. For example, two sequences having 85% identical residues over a 100-residue window will have at least 70 identical dipeptides, 55 tripeptides and 25 pentapeptides. Therefore, pairs of sequences that don't satisfy these conditions don't have to be aligned, which allows speeding up the clustering of the database.

In nrdb90, the authors mainly use a decapetide filter, because two sequences with over 90% identity must have at least one identical decapetide. However, despite using a lookup table, the nrdb90 still needs more than 1 day to process the current NR database on a workstation. At the same time, the use of decapetide filter limits the threshold to 90% of sequence identity or more.

To address these problems, we introduce a new filtering mechanism based on shorter words (from 2 to 5 residues) and an index table. The advantage of short words is that they can be easily indexed. For pentapeptide, since the total number of them is only  $21^5$ , (each residue has 21 possibilities, 20 amino acids plus 'X'), the index table requires only 4 million entries. So an index table with each entry representing a unique pentapeptide can fit into memory for a PC-based workstation. In our method, each entry in the index table holds numbers of the representative sequences containing this pentapeptide.

When a new sequence is processed, the short word filter works in the following steps. (1) The program finds the types and numbers of pentapeptides the new sequence has. (2) In the index table, the entries of these pentapeptides are scanned. (3) For each old representative sequence, the number of identical pentapeptides with the new sequence is calculated. (4) Only if this number is greater than the required value, an alignment is preformed to confirm

\*To whom correspondence should be addressed.

**Table 1.** Clustering of PDB, SWISS-PROT and NR. Different word lengths were chosen for different thresholds. The processing time was compared with the program nrdb90 at 90% threshold, while the thresholds below 90% were not applicable for nrdb90.

Database	Threshold (%)	Word length	Clusters	Time (minute)	Time of nrdb90
PDB	90	5	4 342	0.25	8.5
8732 sequences	80	4	3 907	0.26	NA
1 850 235 letters	75	3	3 767	0.44	NA
	65	2	3 535	3.96	NA
SWISS-PROT	90	5	71 180	6.4	208
88 780 sequences	80	4	62 272	15.6	NA
31 984 247 letters	75	3	58 651	96.4	NA
NR	90	5	316 436	117	2176
563 276 sequences	80	4	264 495	325	NA
177 028 588 letters	75	3	247 074	1597	NA

their sequence identity. In this way, the filter avoids many unnecessary pairwise alignments.

Processing an index table is much faster than handling a lookup table. Also, using shorter word filters enables the program to cluster databases at relatively lower thresholds. The pentapeptide and tetrapeptide filters are efficient for thresholds above 85 and 80% respectively, tripeptide and dipeptide are valid for thresholds above 75 and 65% respectively.

The program was written in C++, and was developed on a Linux PC. The input to the program is a protein database in FASTA format. The primary output is a FASTA file with representative proteins. The second output file lists the clusters and their members. For a large database, the program can divide it into several parts according to the available memory, and temporary data of each part can be saved on hard disk. This design allows the calculations to be done on computers with smaller memory.

## RESULTS AND CONCLUSIONS

We tested this program on a PC with a 400 MHz PII processor and 512 MB RAM running RedHat Linux 6.1. The target databases were PDB, SWISS-PROT and NR downloaded from the NCBI on 20 September, 2000. The results are shown in Table 1.

Compared with nrdb90, this program is about 20–30 times faster, and it can push the threshold to lower levels, resulting in smaller final databases. The use of the clustered databases saves time in database searching without loss in recognition accuracy. Actually, preliminary data for fold recognition using PSI-BLAST and other profile-based methods suggests improving recognition of distant homologues. The detailed analysis of these results will be presented in a separate work.

The source code is available from <http://bioinformatics.burnham-inst.org/cd-hi>.

## ACKNOWLEDGEMENTS

This research was partly supported by the NIH grant GM60049.

## REFERENCES

- Altschul,S.F., Madden,T.L., Schaffer,A.A., Zhang,J., Zhang,Z., Miller,W. and Lipman,D.J. (1997) Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Res.*, **25**, 3389–3402.
- Holm,L. and Sander,C. (1998) Removing near-neighbour redundancy from large protein sequence collections. *Bioinformatics*, **14**, 423–429.