Consistent Triplets of Candidate Paralogs by Graph Clustering

HwaSeob Yun[1] **Ilya Muchnik**[1,2] **Casimir Kulikowski**[1]
[1]*Department of Computer Science, Rutgers University, New Jersey, USA*
[2]*DIMACS, Rutgers University, New Jersey, USA*
Email : seabee@cs.rutgers.edu, muchnik@dimacs.rutgers.edu, kulikows@cs.rutgers.edu

ABSTRACT: We introduce a fully automatic clustering method to classify candidate paralog clusters from a set of protein sequences within one genome. A set of protein sequences is represented as a set of nodes, each represented by the amino acid sequence for a protein with the sequence similarities among them constituting a set of edges in a graph of protein relationships. We use graph-based clustering methods to identify structurally consistent sets of nodes which are strongly connected with each other. Our results are consistent with those from current leading systems such as COG/KOG and KEGG based on manual curation. All the results are viewable at http://www.cs.rutgers.edu/~seabee.

1 INTRODUCION

A paralog is one of a set of homologous genes that have diverged from each other as a consequence of gene duplication, in contrast to an ortholog which is a gene connected by vertical evolutionary descent to another. Typically, orthologs perform the same function in different species, while paralogs often acquire new functions even though they are found within the same species [21].

There are many ways of defining similarities between pairs of protein sequences [2][3][5][6][7][8], but only a few systematic procedures to classify groups of paralogous proteins [9][22]. The recent proliferation of full genome scale sequencing data over many organisms makes such homolog searching in complete genomes increasingly more pressing [12][14][15][19]. By clustering families of paralogous proteins in a genome, matching a newly discovered protein sequence to one of the well-annotated and curated families provides information about the unknown protein and helps predict its function [16][18].

This is more accurate than comparing the new sequence to individual proteins in a database because protein families reflect evolutionary relationships where function often follows family lines [17][20]. Automatically identifying candidate paralog clusters, we can help filter them out, which would be valuable when building orthologous groups of proteins across species as in KEGG [9].

The next section introduces our novel method of finding candidate paralog clusters from amino acid sequence information only for either complete or incomplete genomes.

2 METHODS

Our method finds all the possible candidate paralog clusters from a given set of sequences as long as they are from the same genome. There are, however, advantages of knowing the complete genome sequence. When all known annotations have been applied to sequences in a complete set. Those that remain unannotated can be identified and set aside for further analysis and prediction. Since increasingly comprehensive genomic level comparison across different organisms is possible with complete sequencing of many genomes, peculiarities and novelties in each organism can be studied systematically [11].

2.1 Input data and preprocessing

Starting with a set of protein sequences from one complete (or incomplete) genome, we BLAST [1] all-against-all to check similarities. If these are above a given threshold, then a graph is generated between two nodes representing the proteins. The better score is taken from the asymmetric BLAST scores for a pair of proteins so as to obtain an undirected edge. With this set of weighted edges and nodes, an initial graph of gene similarities (G_S) is created. BLAST is used for this preprocessing because of its speed and universal acceptance and availability.

2.2 Consistent triplets (CTs) and their clusters

From the initial graph of gene similarities (G_S) one can then extract a subgraph (G_T) where all nodes and edges are members of at least one triplet comprising three protein sequences closely related to each other. Since gene similarity does not guarantee transitivity, each pair of nodes in a triplet has to have its sequence alignment information from BLAST results checked for consistency. When the three pairwise alignments in a triplet have overlapping segments longer than some threshold, we can say that a consistent triplet has been found. A new subgraph (G_C) can then be constructed from these consistent triplets.

We define a cluster of consistent triplets as a CT-cluster when all nodes in the cluster have more than a certain number of other triplets connected to it, and it is the *largest* possible set of nodes in G_C. CT-clusters can be defined by thresholds and combinatorial conditions, relevant to system decomposition, but definition-based procedures for finding CT-clusters are inefficient, requiring much search, while predefining the thresholds requires an expert intervention.

2.3 Cluster-maximizer

If W is a set of nodes in G_C and $\pi(i, H)$ denotes the number of consistent triplets related to a node i in the subset of nodes H ($i \in H \subseteq W$, $|H| \geq 3$), then a connected component H^* maximizer is defined by the maxmin equation:

$$H^* = \arg \max_{\substack{H \in 2^W \\ |H| \geq 3}} \min_{i \in H} \pi(i, H) \qquad (1)$$

H^* extracts a CT-cluster from a set of genes W. With many

advantages over threshold-based CT-clustering methods, the cluster-maximizer approach starts from a triplet of nodes and always checks the connectivity among the components. Threshold-based clustering requires predefined thresholds which require manual curation [13], where the cluster-maximizer automatically yields a threshold from the maxmin conditions. Every node in a CT-cluster has an equal or larger number of consistent triplets, and the largest CT-cluster found by H^* is also a connected component. The largest subset which is not a connected component but satisfies the given maxmin condition, forms a union of a small number of connected components maximizers, and is unique.

For each connected component in G_C, we find the cluster-maximizer and remove the nodes included in this H^*. This process is iterated over the remaining subset of nodes until no more consistent triplets are found. The known threshold-based procedure to find such clusters has an exponential complexity, but our cluster-maximizer enables a cluster to be found by a polynomial algorithm. The complexity of this polynomial algorithm is $O(m \cdot n^2)$ where m is the number of connected components (each edge of which belongs in at least one consistent triplet) and n is the number of nodes in the largest connected component.

2.4 Quasi-concave set function

There are two major theoretical foundations for the above procedure: one is monotonicity of the π function and the other is the choice of a quasi-concave set function for H^*. The monotonicity of the π function simply means the number of consistent triplets increases or remains the same if larger sets of nodes are considered:

$$\forall i \in H \subseteq H', \pi(i, H) \subseteq \pi(i, H') \qquad (2)$$

Proof of all of the above is based on the property of the chosen criterion:

$$\begin{aligned} &F(H) = \min \pi(i, H) \\ &\forall H_1, H_2 : \\ &F(H_1 \cup H_2) \geq \min \{F(H_1), F(H_2)\}. \end{aligned} \qquad (3)$$

where $F(H)$ is a quasi-concave set function, which can be found in H^* [10].

3 RESULTS

3.1 CT-clustering vs. COG/KOG

We tested our consistent triplet (CT) clustering approach by running it over the same input data sets that COG/KOG [22] used. Organisms in COG have generally short sequences compared to those of KOG, thus procedures for the entire computation were executed in relatively short periods of time with a generous E-value threshold of 10^{-10} for BLAST. KOG has more advanced and complex organisms with longer protein sequences, so CT-clustering has to be done with the stronger E-value threshold such as 10^{-40} to cut down the overall data needing computation. Total size of these data sets (66 organisms from COG and 7 from KOG) involves over 111 millions amino acids.

Three unicellular organisms (Schizosaccharomyces pombe, Saccharomyces cerevisiae, and Encephalitozoon cuniculi) appear in both COG and KOG but CT-clustering was used separately with different thresholds, so they are treated separately as presented in COG/KOG. Table 1 shows the basic statistical results of CT-clustering coverage over COG/KOG.

	COG	KOG
Total number of sequences	168,160	112, 920
Clustering coverage	21.9%	27.7%
Number of CT-clusters	7,177	5,242
Homogeneous CT-clusters	4,843	4,588
Homogeneity	67.5%	87.5%

Table 1: CT-clustering over COG/KOG data sets.

To investigate the relationship between the CT-clustering coverage and the size of a CT-cluster, its coverage is plotted against the average size of CT-clusters for each organism from a COG and a KOG.

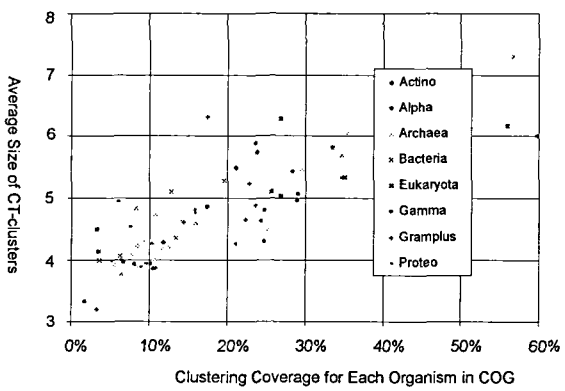

Figure 1: CT-clustering coverage vs. average size of CT-clusters in 66 organisms from COG.

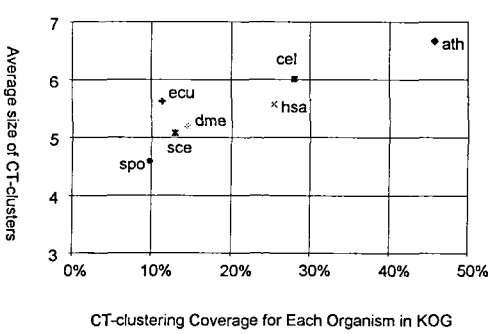

Figure 2: CT-clustering coverage vs. average size of CT-clusters in all 7 organisms from KOG.

ath	=	Arabidopsis thaliana
cel	=	Caenorhabditis elegans
dme	=	Drosophila melanogaster
hsa	=	Homo sapiens
sce	=	Saccharomyces cerevisiae
spo	=	Schizosaccharomyces pombe
ecu	=	Encephalitozoon cuniculi

Figure 1 shows 66 unicellular organisms in COG divided by their 8 predefined groups: Archaea, Eukaryota, Bacteria,

Actino bacteria, Gram plus, gamma, Proteobacteria, alpha. Figure 2 plots the CT-clustering coverage of the 7 eukaryotes in KOG with its 3-letter codes. As these two scatter plots show, the more sequences covered by CT-clusters, the larger the size of CT-clusters.

If there are more sequences in a genome, we expect to have more coverage of clustering than for a genome with fewer sequences, but if we take a look at the results of CT-clustering over KOG, several interesting points arise. First of all, the largest genome in KOG is Homo sapiens with 38,638 sequences and the second is Arabidopsis thaliana with 26,406 sequences, but CT-clustering over Arabidopsis thaliana produced more candidates of paralogous proteins than that of Homo sapiens: 12,097 sequences (45.8% of 26,406) compared to 9,842 sequences (25.5% of 38,638). This coverage over Homo sapiens is surpassed by the third largest genome in KOG (Caenorhabditis elegans) which has 5,829 sequences covered by CT-clustering for 28.1% of 20,751 sequences from its genome. Thus, simply because there are more sequences in a genome, it does not mean it is going to have more paralogs. Plants are known for having many paralogous relations, which shows up in this CT-clustering over KOG.

The larger average size of CT-clusters means more than average CT-clustering coverage. It means there are even stronger paralogous relations in those CT-clusters than the smaller ones because a CT-cluster is not an ordinary cluster of sequences but each sequence in it must have at least two other sequences connected by the triplet consistency. Therefore, as the average size of CT-clusters increases with its CT-clustering coverage in a given genome, we can assume that a higher level of paralogy exists in that genome.

Next, taking a closer look at the size of CT-clusters in various organisms, we need to divide CT-clusters in two groups: one is a group of minimum triplets having only three sequences in them, and the other is a group of CT-clusters with four or more sequences with triplet consistency. In this way, we can clearly see which organisms have small CT-clusters as their majority suggests fewer level of paralogy.

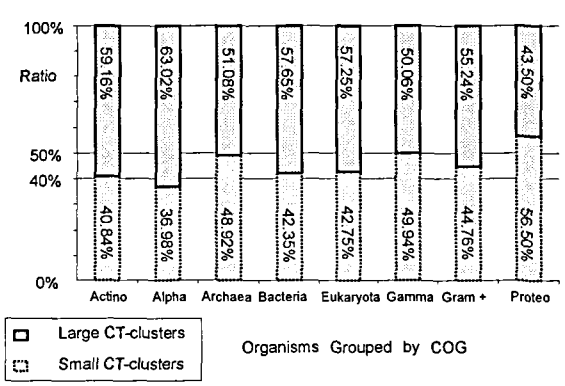

Figure 3: Ratio of 3-node CT-clusters to 4 or larger ones in 66 organisms from COG.

The larger number of small CT-clusters in COG organisms is discovered within the group of Proteobacteria as shown in Figure 3, where 56.5% of CT-clusters within this group have only three protein sequences as paralogs. In other words, sequences in Proteobacteria do not have many paralogs within their genome, which is consistent with the small coverage of overall CT-clustering in this Proteobacteria group as seen in Figure 1. There are 6 organisms in this group, of which five (Neisseria meningitidis MC58, Neisseria meningitidis Z2491, Helicobacter pylori 26695, Helicobacter pylori J99, and Campylobacter jejuni) have less than 10% coverage.

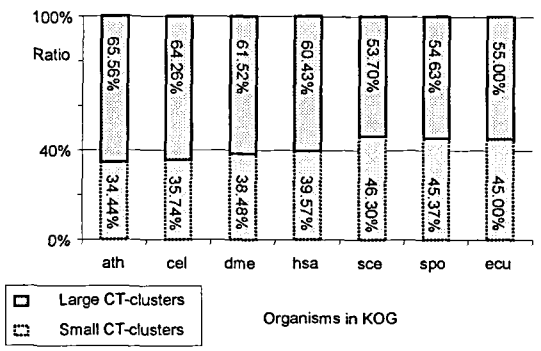

Figure 4: Ratio of 3-node CT-clusters to 4 or larger ones in 7 organisms from KOG.

These results are also consistent with those of CT-clustering over the seven organisms from KOG. Three unicellular eukaryotes (Saccharomyces cerevisiae, Encephalitozoon cuniculi, and Schizosaccharomyces pombe) have a relatively large number of small CT-clusters (more than 45% of all CT-clusters found in them as shown in Figure 4), all with less than 13% of CT-clustering coverage as shown in Figure 2. On the other hand, the other four more complex organisms (Arabidopsis thaliana, Caenorhabditis elegans, Drosophila melanogaster, and Homo sapiens) have less than 40% of small CT-clusters and their coverage is higher than those of the three simple eukaryotes. Arabidopsis thaliana has the least proportion (34.44%) of small CT-clusters and the most CT-clustering coverage (45.8%).

3.2 CT-clustering vs. KEGG

CT-clustering and KEGG share much in common, starting from a graph of gene similarities among nodes. We compared the clustering results of these two different methods over the sets of proteins from the same organism, Homo sapiens. Table 2 summarizes the principle statistical results for the two clustering methods.

Proteins from Homo sapiens	CT-clustering	KEGG
Total number of sequences	12,921	11,667
Total number of clusters	2,048	1,989
Average size of clusters	6.3	5.8
Standard deviation	11.4	17.7
Homogeneous clusters	677	1,028
Homogeneity of clusters	33.1%	51.7%

Table 2: Comparison of results of two clustering methods.

We used 38,638 protein sequences from NCBI as input for CT-clustering while KEGG used 23,504 protein sequences as its input. However, the number of sequences covered by each clustering method is more similar: 12,921 and 11,667

sequences for CT-clustering and KEGG respectively. The total numbers of clusters are also close to each other: 2,048 clusters for CT-clustering and 1,989 clusters for KEGG. Thus the average sizes of clusters contain around 6 sequences per cluster for both methods but the standard deviation over the size of clusters for KEGG is much larger than that of CT-clustering, suggesting that CT-clustering leads to a more homogeneous distribution of clustering in terms of cluster size.

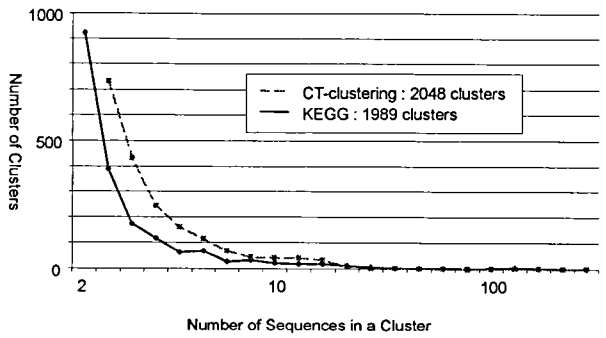

Figure 5: Distribution of the size of a cluster in two graph clustering methods of Homo sapiens.

Figure 5 shows the majority of clusters having two to five sequences in them: 1,408 clusters for CT-clustering (68.8%) and 1,597 clusters for KEGG (80.3%). The last two lines of Table 2 reveal an interesting relationship of these two clustering methods. Each of 677 clusters by CT-clustering has the same KEGG cluster IDs, and each of 1,028 clusters from KEGG is identified by the same CT-clusters. In most cases, KEGG clusters are identified as one CT-cluster or divided into two CT-clusters (90.1%), where CT-clusters are about the same (92.6%).

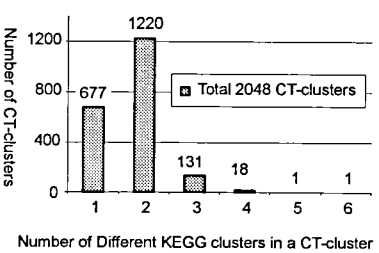

Figure 6: Consistency of CT-clusters verified by KEGG..

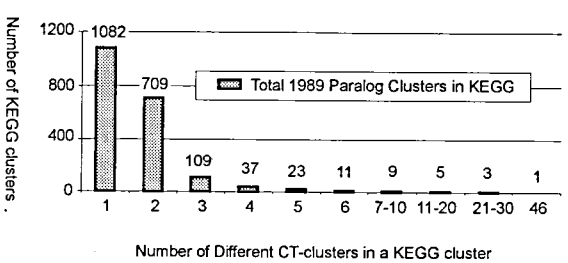

Figure 7: Consistency of KEGG clusters verified by CT-clustering method.

However, there are significant differences in this: as shown in Figures 6 and 7, there are more KEGG clusters having one CT-cluster ID in them than two, but in the case of CT-clusters, the opposite occurs. In CT-clustering, not only there are more clusters having two KEGG IDs in them, but there can also be more, with up to six KEGG cluster IDs found in a single CT-cluster. Only two out of 2,048 clusters have five or six KEGG cluster IDs, and all the other clusters have a diversity of four or less. CT-clusters fit into KEGG better with higher homogeneity.

As shown in Figure 7, there is one case where a KEGG cluster has 46 CT-clusters in it, and as shown in Figure 6, the maximum number of different KEGG clusters in any CT-cluster is six. This suggests that the CT-clustering method has extracted stronger clusters than KEGG does, which is also supported by the fact that CT-clustering starts from consistent triplets as a basic building block of its clusters. This consistent triplet has not only a high sequence similarity score but also strong structural consistency checked by overlapping alignment regions. On the other hand, KEGG takes a quasi-clique as its cluster from a graph of protein-nodes where edges are produced by pairwise sequence similarity scores only.

4 CONCLUSIONS

Consistent triplet clustering (CT-clustering) results have been shown to give fast and robust clustering results. This procedure is fully automatic, not requiring any human intervention during the entire process. It uses a monotone linkage function as its criterion to extract unique consistent triplets from a set of elements which have interrelationships with one another.

To verify its performance, CT-clustering was tested on whole genome data sets: from organisms collected in COG and KOG. To find paralogs from the given sets of sequences, no expensive multialignment was required. Instead, the pairwise sequence alignment software from BLAST was used to find similarities among protein sequences, from which consistent triplets of sequences are found. Using a quasi-concave set function, a core within the group of consistent triplets is extracted as a CT-cluster. All these results are available at http://www.cs.rutgers.edu/~seabee.

CT-clustering is based on transitivity of similarities among three structural objects, thus the members in its cluster share stronger homogeneity than other clustering methods which does not consider structural consistency. This is proven in our comparative studies with COG/KOG and KEGG by higher homogeneity of CT-clusters. Clustering results from KEGG share many similarities with those of CT-clustering since both of them are based on graph-theoretical methods. Nevertheless, CT-clustering showed stronger homogeneity over the same data set (Homo sapiens) because KEGG does not take into account the structural consistency of sequence alignments. CT-clustering was also able to capture the degree of paralogy in organisms by the ratio of large to small CT-clusters in them.

Just as COG extended its genome sets into KOG, more new genome data will be shortly produced and available for extensive study. CT-clustering can adapt to this constantly growing pool of genes and produce new CT-clustering results for them. To efficiently perform this routine work, the data

sets need to be stored and managed in a database. So, web reports can be dynamically generated to users' different needs, with a web interface to submit sets of sequences from new genomes to perform their own analyses. With such a new web interface, a set of sequences, raw results, or even the CT-clustering method itself can be directly downloaded to users, so that they can compute their own CT-clustering locally with their own computing resources.

In parallel, we are working to extend our comparative study with KEGG. Currently, CT-clustering of Homo sapiens is the only one we have compared with KEGG's results. There are far more organisms they cover which can be of interest. COG and KOG clustering results can also be compared with newly calculated CT-clusterings with different thresholds. For example, one could find the point where CT-clustering coverage becomes approximately the same as COG or KOG with more relaxed thresholds.

REFERENCES

[1] S. F. Altschul, W. Gish, W. Miller, E. W. Myers, and D. J. Lipman. Basic local alignment search tool. J. Mol. Biol., 5:403–410, 1990.

[2] T. K. Attwood, M. E. Beck, A. J. Bleasby, K. Degtyarenko, A. D. Michie, D. J. Parry-Smith. Novel Developments with the PRINTS Protein Fingerprint Database, Nucleic Acids Research, 25:212–217, 1997.

[3] A. Bairoch and R. Apweiler. The SWISS-PROT protein sequence data bank and its supplement TrEMBL. Nucleic Acids Research, 27:49–54, 1999.

[4] S. Bornholdt and H. G. Schuster, eds., Handbook of Graphs and Networks: From the Genome to the Internet. Wiley-VCH, Weinheim, 2003.

[5] D. W. Buchan, et al. Structural Assignment for Whole Genes and Genomes Using the CATH Domain Structure Database, Genome Research, 12(3):503–14, 2002.

[6] L. Falquet, et al. The PROSITE database, its status in 2002, Nucleic Acids Res., 30:235–238, 2002.

[7] S. Henikoff, J. G. Henikoff, and S. Pietrokovski. BLOCKS+ : A non-redundant database of protein alignment blocks derived from multiple compilations, Bioinformatics, 15(6):471–479, 1999.

[8] L. Holm and C. Sander. Touring protein fold space with DALI/FSSP. Nucleic Acids Research 26: 316–319, 1998.

[9] M. Kanehisa and S. Goto, KEGG: Kyoto Encyclopedia of Genes and Genomes. Nucleic Acids Res. 28, 27–30, 2000.

[10] Y. Kempner, B. Mirkin, and I. B. Muchnik. Monotone Linkage Clustering and Quasi-Concave Set Functions, Appl. Math. Lett., Vol. 10, No. 4, 19–24, 1997.

[11] E. V. Koonin, Y. I. Wolf, and G. P. Karev. The structure of the protein universe and genome evolution. Nature, 420: 218–223, 2002.

[12] H. W. Mewes, et al. MIPS: A database for genomes and protein sequences Nucleic Acid Research, Vol. 28, No. 1: 37–40, 2000.

[13] B. Mirkin. Mathematical Classification and Clustering. Kluwer Publ. Boston, MA. 1996.

[14] N. Mulder, et al. The InterPro Database, 2003 brings increased coverage and new features, Nucleic Acids Research, Vol. 31, No. 1: 315–318, 2003.

[15] J. Murvai, et al. The SBASE protein domain library, release 7.0: a collection of annotated protein sequence segments. Nucleic Acids Research, 28: 260–262, 2000.

[16] A. Murzin, S. E. Brenner, T. Hubbard, C. Chothia. SCOP: a structural classification of proteins data-base for the investigation of sequences and structures. J Mol Biol 247: 536–540, 1995.

[17] C. A. Orengo, et. al. The CATH protein family database: a resource for structural and functional annotation of genomes. Proteomics, 2(1): 11–21, 2002.

[18] J. Schultz, et al. SMART: a web-based tool for the study of genetically mobile domains, Nucleic Acids Research 28: 231–234, 2000.

[19] F. Servant, et al. ProDom: Automated clustering of homologous domains, Briefings in Bioinformatics, Vol. 3, No. 3: 246–251, 2002.

[20] E. L. Sonnhammer, S. R. Eddy, and R. Durbin. Pfam: a comprehensive database of protein domain families based on seed alignments. Proteins, 28: 405–420, 1997.

[21] R. L. Tatusov, E. V. Koonin, and D. J. Lipman. Genomic perspective on protein families. Science. 278: 631–637, 1997.

[22] R. L. Tatusov, et al. The COG database: new developments in phylogenetic classification of proteins from complete genomes. Nucleic Acids Res. 29 (1): 22–28, 2001.