Automated Construction of Phylogenetic Trees

Curtis Bright
University of Waterloo
cbright@cs.uwaterloo.ca

Nam Pham
University of Waterloo
npham@cs.uwaterloo.ca

December 20, 2010

Abstract

Given a diverse collection of genomes we implement and analyze an automatic method of approximating the evolutionary relationships of the source organisms, using the techniques of *compression* and *clustering*.

1 Introduction

Construction of evolutionary trees in an important task in computational biology. Traditionally this was painstakingly done by extensive analysis of the organisms in question, but recently the widespread growth and availability of computers has lead to techniques to automatically estimate phylogeny trees from, for example, a database of DNA sequences.

In particular, a method of Rudi Cilibrasi and Paul Vitányi [4] is the focus of this report. The method relies on the concept of *normalized information distance* from information theory to determine a 'closeness' between two pieces of information, and in practice relies on *data compression* as a method of computation. Once the distances between each pair of database elements is computed, a *clustering* method is required to group the data into a hierarchical structure; in particular, we implemented the so-called *quartet method*. Both of the main steps in the algorithm (compression and clustering) are completely general, and not limited to phylogenic analysis.

An inspiration for this project was the possibility of reconstructing the mutations which lead to the 'swine flu' pandemic of 2009. However, as the project progressed we were led to consider other virii, including the SARS virus which was responsible for a near-pandemic in 2003. Additionally, as a method of testing our clustering implementation, we repeated the construction of a 24-species phylogeny tree which appeared in [4].

The remainder of this report is organized as follows: Section 2 covers the basic background necessary, and sections 3 and 4 go into detail about the compression and clustering steps. Finally, section 5 contains an analysis of our results, and offers conclusions.

2 Background

2.1 Kolmogorov Complexity and Normalized Information Distance

Kolmogorov complexity was introduced by Solomonoff, Kolmogorov, and Chaitin [8] as a measurement of the information of an object. Any object could be described in different ways for different purposes. For example, the following three strings have the same meaning: *abcabcabcabc*, $(abc)^4$, "*abc* 4 times", but they vary greatly in length. A natural question to ask is: which description is shortest? To answer this question, Kolmogorov complexity is defined as the size of the shortest program that can produce a string x in some fixed universal description language U:

$$K(x) = \min\{\, |p| : U(p) = x \,\}$$

One of the most important uses of Kolmogorov complexity is to estimate the universal information distance between two objects. Ming Li and Paul Vitányi defined the universal normalized information distance, or NID, between x and y to be

$$\mathrm{NID}(x, y) = \frac{\max\{K(x \mid y), K(y \mid x)\}}{\max\{K(x), K(y)\}}$$

in which $K(x \mid y)$ denotes the length of the shortest program to produce x, given y (as in [7]). Unfortunately, since the Kolmogorov complexity is not computable, we have to approximate it via compression. If $C(x)$ denotes the length of compressed x, the normalized compression distance, or NCD, between x and y can approximate the NID and is given by

$$\mathrm{NCD}(x, y) = \frac{C(xy) - \min\{C(x), C(y)\}}{\max\{C(x), C(y)\}}.$$

This measurement provides a useful tool for researchers in several areas, such as bioinformatics [1, 6], software engineering [2] and web search [14].

2.2 Normalized Information Distance and Cluster Analysis

Information retrieval systems first need a measure of similarity between documents before clustering, i.e., organizing objects into similar groupings. Several techniques have been developed for this purpose, including Bayesian and classical inference [10], as well as neural networks [12]. However, those techniques have to rely on the understanding of the document structure. In contrast, the normalized information distance based on Kolmogorov complexity is universal so it can be used to calculate the distance between any two items without relying on parsing or semantic analysis. In fact, it has been used for author identification in online postings [11], contextual information retrieval [9], and spam filtering [13].

Clustering is one of the most successful applications of using NID in information retrieval systems. Cilibrasi and Vitányi made the proposal to use normalized compression distance in hierarchical clustering [4]. It was subsequently successfully applied in clustering languages [7], music [5], and several other applications.

3 Normalized Compression Distance

3.1 Data Collection

The sequences used in our experiments were downloaded from the National Center for Biotechnology Information (NCBI) database. The sequence files from NCBI are in FASTA format, so the first line in each file contains a sequence description. Since this is not actually part of the sequence it was removed to avoid skewing the computed distances.

It should also be noted that in some cases where complete coding sequences were not available, we used partial ones instead.

3.2 Distance Calculation

The normalized compression distance matrix was computed using the compressor bzip2. To calculate the distance between two sequences, we have to run bzip2 three times. First of all, we have to compress the two original files. Then, to calculate $C(xy)$ we have to compress the two files combined together. Running bzip2 on the combined file allow the compression algorithm develop a dictionary on the first file and then apply it to the second. Ideally, if two files x and x' are similar, we will have

$$\text{bzip2}(xx') - \text{bzip2}(x) \approx 0,$$

where bzip2(x) denotes the size of the compressed file containing x. In practice, bzip2 is not optimal compared to Kolmogorov complexity, hence

$$\text{bzip2}(xx') - \text{bzip2}(x) = \epsilon$$

and

$$0 \leq \text{NCD}(x, y) \leq 1 + \epsilon.$$

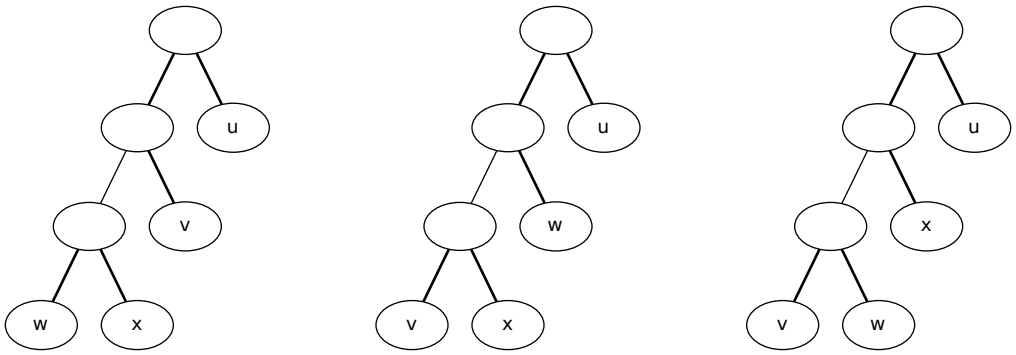
It is very interesting to see that even though individual sequences vary greatly in size, the NCD between each pair still represents their relationship. As seen in Table 1 the distance between two virii of a similar kind is always smaller than the distance between two unrelated virii. For example, AvianIB1 is closer to AvianIB2 than any other virus.

Besides bzip2, we also conducted a few experiments with GenCompress [3]. Even though GenCompress has a better compression ratio, running our clustering algorithm on the GenCompress distance matrix returned identical results. Hence, we decided to use bzip2 as our main compressor in this project.

The process is, unfortunately, very time consuming. To create a distance matrix of n articles, we have to run the compression algorithm $n^2 + n$ times. In an experiment, it took 6 hours to generate a 50×50 matrix.

	AAC	IB1	IB2	BA3	DA1	HA40	HC1	MM	MS	MH11	MH2	PRD1	RSC	SARS	SIRV1	SIRV2
AvianAdenolCELO	0.4261	0.9713	0.9949	0.9940	0.9900	0.9651	0.9903	0.9843	0.9843	0.9955	1.0004	0.9896	0.9910	0.9912	1.0019	1.0013
AvianIB1	0.9713	0.4475	0.9509	1.0014	0.9918	0.9726	0.9857	0.9847	0.9847	0.9903	0.9912	0.9896	0.9906	0.9901	0.9953	0.9996
AvianIB2	0.9949	0.9509	0.4631	1.0017	0.9917	0.9953	0.9910	0.9946	0.9946	0.9917	0.9923	1.0016	0.9915	0.9916	1.0067	1.0018
BovineAdeno3	0.9940	1.0014	1.0017	0.4630	0.9958	0.9958	0.9994	0.9945	0.9945	0.9990	0.9981	0.9993	0.9993	0.9978	1.0139	1.0178
DuckAdeno1	0.9900	0.9918	0.9917	0.9958	0.4644	0.9923	0.9903	0.9907	0.9907	0.9903	0.9915	0.9979	0.9931	0.9903	1.0067	1.0050
HumanAdeno40	0.9651	0.9726	0.9953	0.9958	0.9923	0.3851	0.9879	0.9911	0.9911	0.9922	0.9969	0.9831	0.9930	0.9953	0.9953	0.9986
HumanCorona1	0.9903	0.9857	0.9910	0.9994	0.9903	0.9879	0.4502	0.9934	0.9934	0.9884	0.9877	1.0007	0.9896	0.9868	1.0035	0.9992
MeaselsMora	0.9843	0.9847	0.9946	0.9945	0.9907	0.9911	0.9934	0.4654	0.4654	0.9908	0.9920	1.0040	0.9953	0.9923	1.0097	1.0061
MeaselsSch	0.9843	0.9847	0.9946	0.9945	0.9907	0.9911	0.9934	0.4654	0.4654	0.9908	0.9920	1.0040	0.9953	0.9923	1.0097	1.0061
MurineHep11	0.9955	0.9903	0.9917	0.9990	0.9903	0.9922	0.9884	0.9908	0.9908	0.4695	0.5319	1.0018	0.9164	0.9918	1.0086	1.0149
MurineHep2	1.0004	0.9912	0.9923	0.9981	0.9915	0.9969	0.9877	0.9920	0.9920	0.5319	0.4732	1.0024	0.9192	0.9911	1.0054	1.0075
PRD1	0.9896	0.9896	1.0016	0.9993	0.9979	0.9831	1.0007	1.0040	1.0040	1.0018	1.0024	0.4342	1.0030	1.0009	1.0048	1.0083
RatSialCorona	0.9910	0.9906	0.9915	0.9993	0.9931	0.9930	0.9896	0.9953	0.9953	0.9164	0.9192	1.0030	0.4732	0.9895	1.0087	1.0051
SARS	0.9912	0.9901	0.9916	0.9978	0.9903	0.9953	0.9868	0.9923	0.9923	0.9918	0.9911	1.0009	0.9895	0.4609	1.0052	1.0108
SIRV1	1.0019	0.9953	1.0067	1.0139	1.0067	0.9953	1.0035	1.0097	1.0097	1.0086	1.0054	1.0048	1.0087	1.0052	0.4150	0.9180
SIRV2	1.0013	0.9996	1.0018	1.0178	1.0050	0.9986	0.9992	1.0061	1.0061	1.0149	1.0075	1.0083	1.0051	1.0108	0.9180	0.4184

Table 1: The NCD matrix for 16 virii



(a) u paired with v (b) u paired with w (c) u paired with x

Figure 1: The three topologies of the nodes u, v, w, x

4 Quartet Clustering

Most generally, *cluster analysis* is a method of classifying observations into groups. For our purposes, we require *hierarchical* groups, from which we can infer the evolutionary tree of the organisms in question. There are many proposed schemes for accomplishing this; we follow Cilibrasi and Vitányi and use the so-called *quartet method*. In this method the organisms will be represented by leaf nodes in a full binary tree, and the internal nodes of the tree will represent the clusters; the organisms in the cluster being those descendant of the node.

Given any four organisms, consider the three possible *quartet topologies* shown in Figure 1. The topologies consist of "pairing together" the four nodes into two groups of two, i.e., u can be paired with v, w, or x, and the two remaining nodes are also paired together. Here "pairing together" means that the path in the tree between the two nodes in one pair does not cross the path between the two nodes in the other pair.

For any four leaf nodes in a binary tree, there is exactly one way of pairing together the nodes; the topology which works is said to be *consistent* with the tree. This suggests a way of scoring a quartet in a given tree, namely, adding together the NCD between the first pair of nodes in the consistent topology and the NCD between the second pair of nodes in the consistent topology. Now consider calculating the cost of every set of four data points in the tree and summing the result; this is called the *total cost* of the tree.

When two nodes are close by in the tree they are more likely to be paired together in topologies, so if we can find a tree with a low total cost, that means many close pairs of nodes in the tree will have low NCD. That is, the tree will concisely describe, in a simple hierarchical way, the distance relations given by the NCD matrix.

This raises the question: exactly what is a low total cost for a tree? Since a quartet score can only be one of three values, it is a simple matter to take the minimum of every quartet and sum them to find a lower bound m on the best possible tree score. (However, it may be that no tree actually reaches this minimum, for it might not be possible for a single tree to be consistent with every minimal topology.) Similarly, we can find an upper bound M on the worst possible tree score, and scale any tree score C_T. Indeed, for any tree we have the *normalized tree benefit score*

$$0 \leq S(T) = \frac{M - C_T}{M - m} \leq 1.$$

Since a tree's cost can be computed easily, we could simply score random trees and select the one with highest $S(T)$. Instead, we start off with a random tree and perform a process reminiscent of natural selection; we perform 'mutations' on the tree, and if they increase the tree's score they are kept, otherwise they are discarded. In this way we hopefully progress toward a high scoring tree.

The exact mutations used are swapping leaves, swapping subtrees, and transferring subtrees from one place in the tree to another. Usually a only few mutations are done each iteration, though the number of iterations follows a geometric distribution, so that occasionally many mutations will occur at once; this is to help prevent against getting stuck in only a local optimum.

5 Results

A computer program to automatically construct the NCD matrix from the sequences was written in Java, while a program to automatically perform the quartet clustering from the NCD matrix was written in C.

5.1 Swine Flu

Our project was initially focused on identifying the origin of the 2009 'swine flu' virus. We located 168 sequences of the virus in the NCBI database, spread across 8 gene types (NS, MP, NA, NP, HA, PA, PB1, PB2). Running our distance calculator on the 13 sequences of type PB2 produced Table 2.

	1	2	3	4	5	6	7	8	9	10	11	12	13
1 PB2_A_Arizona_02_2009	0.3460	0.3460	0.3460	0.3608	0.3460	0.3460	0.3460	0.3516	0.3544	0.3460	0.3460	0.3460	0.3460
2 PB2_A_California_04_2009	0.3460	0.3460	0.3460	0.3608	0.3460	0.3460	0.3460	0.3516	0.3544	0.3460	0.3460	0.3460	0.3460
3 PB2_A_California_05_2009	0.3460	0.3460	0.3460	0.3608	0.3460	0.3460	0.3460	0.3516	0.3544	0.3460	0.3460	0.3460	0.3460
4 PB2_A_California_06_2009	0.3608	0.3608	0.3608	0.3503	0.3608	0.3608	0.3608	0.3642	0.3665	0.3608	0.3608	0.3608	0.3608
5 PB2_A_California_07_2009	0.3460	0.3460	0.3460	0.3608	0.3460	0.3460	0.3460	0.3516	0.3544	0.3460	0.3460	0.3460	0.3460
6 PB2_A_California_14_2009	0.3460	0.3460	0.3460	0.3608	0.3460	0.3460	0.3460	0.3516	0.3544	0.3460	0.3460	0.3460	0.3460
7 PB2_A_Minnesota_02_2009	0.3460	0.3460	0.3460	0.3608	0.3460	0.3460	0.3460	0.3516	0.3544	0.3460	0.3460	0.3460	0.3460
8 PB2_A_New_York_1669_2009	0.3516	0.3516	0.3516	0.3642	0.3516	0.3516	0.3516	0.3453	0.3600	0.3516	0.3516	0.3516	0.3516
9 PB2_A_New_York_1682_2009	0.3544	0.3544	0.3544	0.3665	0.3544	0.3544	0.3544	0.3600	0.3517	0.3544	0.3544	0.3544	0.3544
10 PB2_A_New_York_18_2009	0.3460	0.3460	0.3460	0.3608	0.3460	0.3460	0.3460	0.3516	0.3544	0.3460	0.3460	0.3460	0.3460
11 PB2_A_Texas_04_2009	0.3460	0.3460	0.3460	0.3608	0.3460	0.3460	0.3460	0.3516	0.3544	0.3460	0.3460	0.3460	0.3460
12 PB2_A_Texas_07_2009	0.3460	0.3460	0.3460	0.3608	0.3460	0.3460	0.3460	0.3516	0.3544	0.3460	0.3460	0.3460	0.3460
13 PB2_A_Texas_09_2009	0.3460	0.3460	0.3460	0.3608	0.3460	0.3460	0.3460	0.3516	0.3544	0.3460	0.3460	0.3460	0.3460

Table 2: The NCD matrix for the swine flu sequences (PB2 gene)

Running our clustering implementation on Table 2 was found to produce an optimal tree quite quickly, after about 100 iterations. However, if allowed to keep testing new trees it found numerous optimal trees, some of which bore no resemblance to each other. An examination of the NCD matrix showed that many values were equal to each other and all values were approximately 0.35. In fact, examining the sequences revealed they were extremely similar, often differing by only a few characters.

We concluded that a compressor such as bzip2 would not be sensitive enough to produce a NCD matrix of precision sufficient for clustering, and unfortunately could not complete the goal of completing a phylogenic tree for the 2009 swine flu virus.

5.2 Test Run with 24 Species

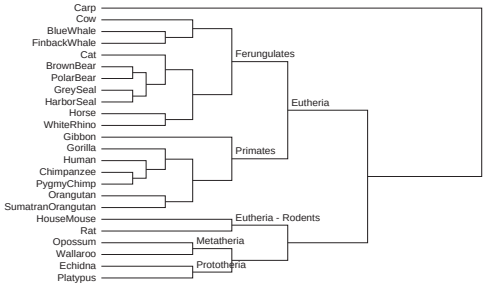
In order to test our clustering algorithm we decided to run our clustering implementation on the 24-species NCD matrix found on page 15 of [4]. They also offer their own phylogeny tree on page 13; for reference this is included in this report as Figure 2a.

The clustering algorithm was run for 1,000,000 iterations, but the best tree was found on iteration 4,078; a graph of the tree score progress is shown in Figure 3a. The best found tree had a normalized tree benefit score of $S(T) = 0.995$, and this tree is shown pictorially in this report as Figure 2b. We have manually labeled the red nodes so as to help comparisons to Figure 2a, which should be noted has the slightly better score of $S(T) = 0.996$. However, as both trees are almost identical to each other, we conclude our implementation passed this test.

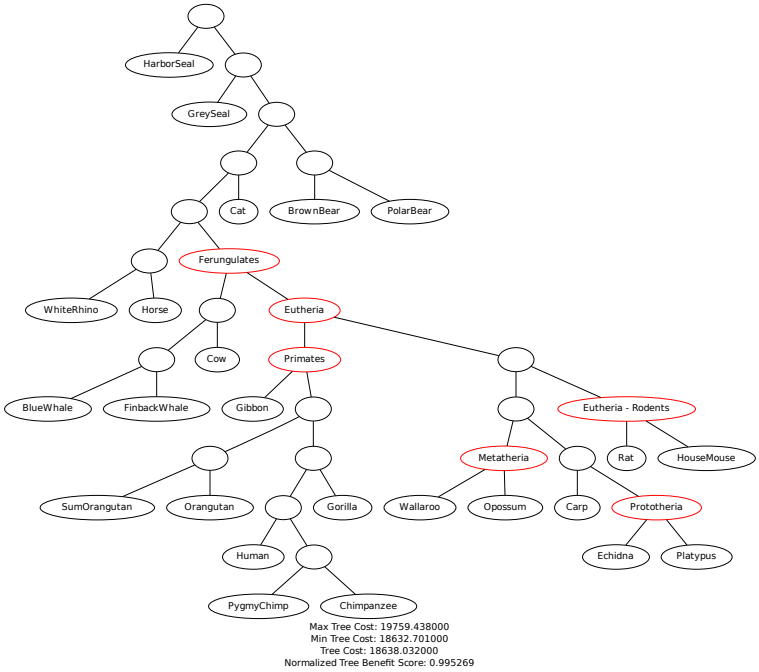
We also compared the quartet clustering method to the *single linkage method* and *Ward's method*, which are available in the R programming environment. The single linkage method worked quite well, discovering another almost identical tree, while Ward's method made some notable errors, for example placing *Gibbon* and *Carp* in close proximity.

Figure 2: Evolutionary tree construction of 24 different species

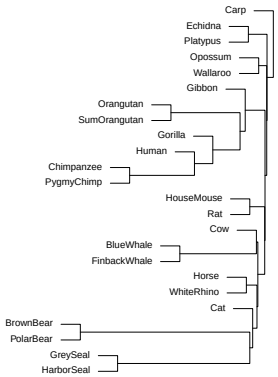
(a) Phylogetic tree construction from [4]



(b) Our constructed tree, quartet method used



(c) Single linkage clustering used



(d) Ward's method clustering used

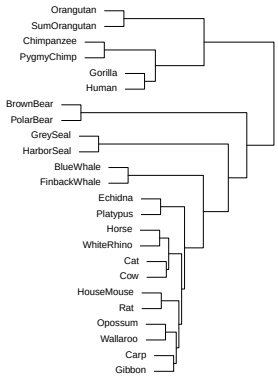
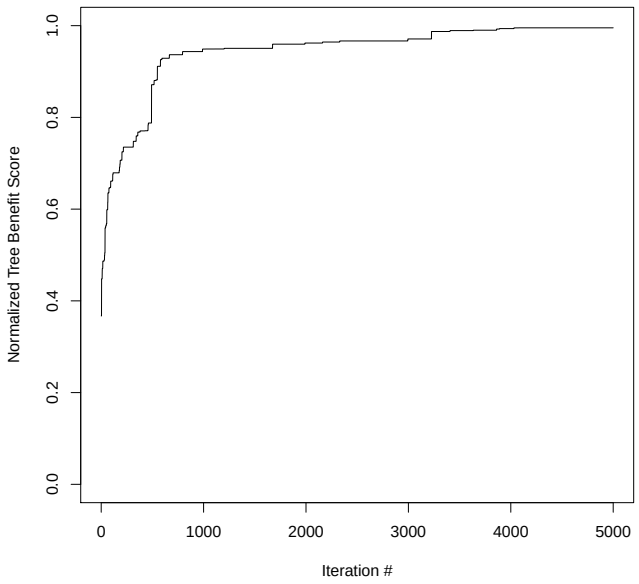
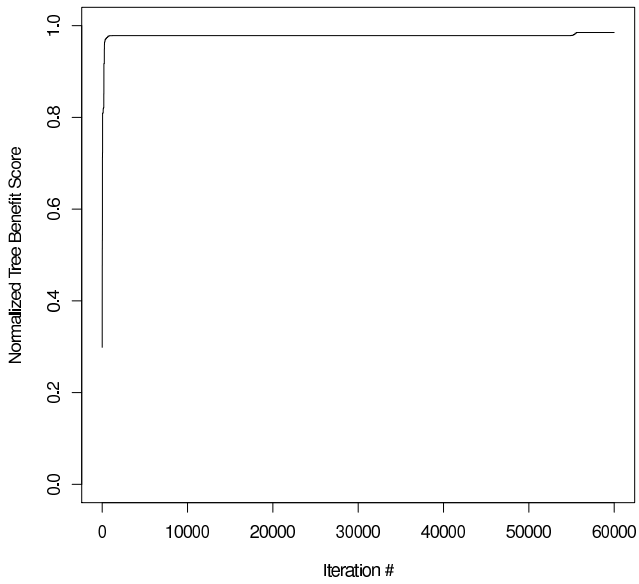


Figure 3: Tree score progress with quartet clustering

(a) 24 species run



(b) 16 virii run



5.3 SARS

In the spirit of our original proposal, we decided on trying to construct a phylogeny tree which would show the evolutionary lineage of the SARS virus, and compare our result to another tree in [4], reproduced here as Figure 4a. Although the exact same virii sequences could not be found, similar ones from the NCBI database were expected to be close enough to enable a comparison between our trees.

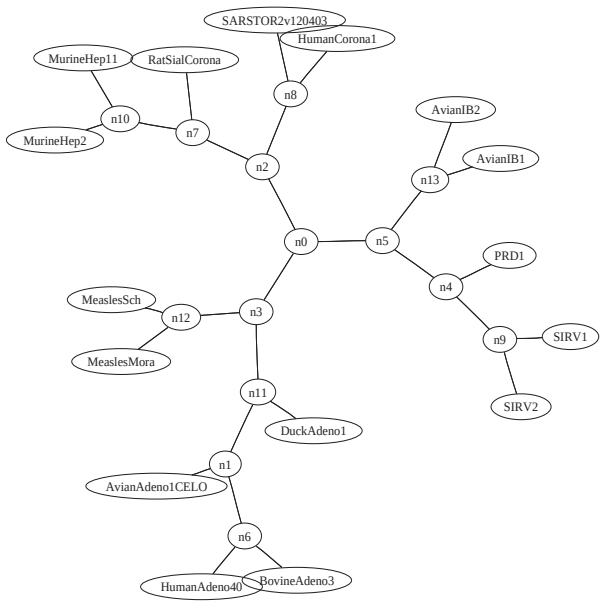
The NCD matrix output has already been given in Table 1. The clustering algorithm was run for 100,000 iterations, and the best tree was found on iteration 55,592, following a long period of inactivity. Probably because of the fewer nodes involved, the tree score progress curve (Figure 3b) was steeper than in the 24 species run; a tree of score better than 0.8 was found after only 50 iterations. The best tree found (Figure 4b) had $S(T) = 0.985$, opposed to $S(T) = 0.988$ in [4], however the trees are fairly similar topologically.

5.4 Final Conclusions

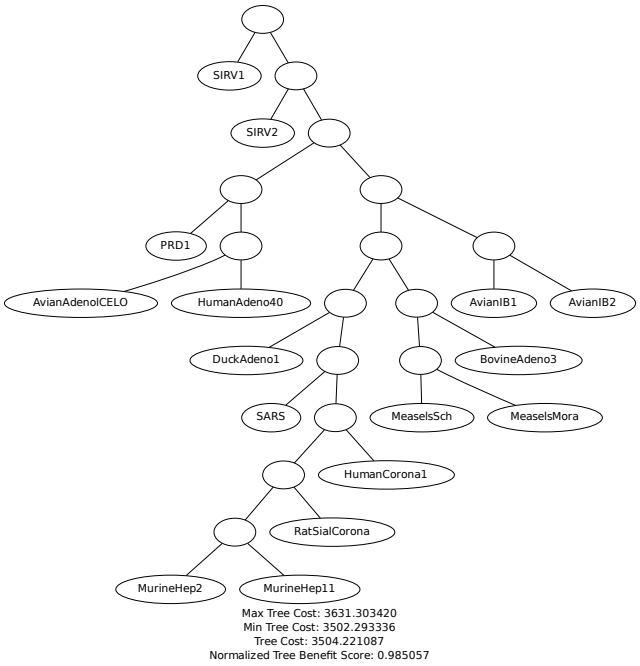
We have demonstrated a very general method for hierarchical clustering which is applicable to many different domains. Assuming the data is available electronically, and of sufficient diversity, one only needs a generic data compressor and our software to automatically organize the data in a hierarchical manner—perhaps revealing previously unseen patterns.

Figure 4: Evolutionary tree construction of 16 different virii

(a) Phylogetic tree construction from [4]



(b) Our constructed tree, quartet method used



References

[1] Cecile Ane and Michael J. Sanderson. Missing the forest for the trees: Phylogenetic compression and its implications for inferring complex evolutionary histories, 2005.

[2] Xin Chen, Brent Francia, Ming Li, Brian Mckinnon, and Amit Seker. Shared information and program plagiarism detection. *IEEE TRANS. INFORM. TH*, 50:1545–1551, 2004.

[3] Xin Chen, Sam Kwong, and Ming Li. A compression algorithm for dna sequences and its applications in genome comparison. pages 52–61, 1999.

[4] R. Cilibrasi and P. M. B. Vitányi. Clustering by compression. *IEEE Transactions on Information Theory*, 51(4):1523–1545, April 2005.

[5] Rudi Cilibrasi, Paul Vitányi, and Ronald De Wolf. Algorithmic clustering of music based on string compression. *Comput. Music J.*, 28:49–67, December 2004.

[6] András Kocsor, Attila Kertész-Farkas, László Kaján, and Sándor Pongor. Application of compression-based distance measures to protein sequence classification: a methodological study. *Bioinformatics*, 22:407–412, February 2006.

[7] Ming Li, Xin Chen, Xin Li, Bin Ma, and Paul Vitányi. The similarity metric. In *Proceedings of the fourteenth annual ACM-SIAM symposium on Discrete algorithms*, SODA '03, pages 863–872, Philadelphia, PA, USA, 2003. Society for Industrial and Applied Mathematics.

[8] Ming Li and Paul Vitányi. *An introduction to Kolmogorov complexity and its applications (2nd ed.)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 1997.

[9] R. Martinez, M. Cebrian, F. de Borja Rodriguez, and D. Camacho. Contextual information retrieval based on algorithmic information theory and statistical outlier detection. In *Information Theory Workshop, 2008. ITW '08. IEEE*, pages 292 –297, May 2008.

[10] F. Mosterller and D. L. Wallice. Applied bayesian and classical inference: the case of the federalist papers, 1964.

[11] David Parry. Use of kolmogorov distance identification of web page authorship, topic and domain, 2005.

[12] S. Singhe and F. J. Tweedie. Neural networks and disputed authorship: new challenges. In *Artificial Neural Networks, 1995., Fourth International Conference on*, pages 24 –28, June 1995.

[13] L. M. Spracklin and L. V. Saxton. Filtering spam using kolmogorov complexity estimates. *Advanced Information Networking and Applications Workshops, International Conference on*, 1:321–328, 2007.

[14] Xian Zhang, Yu Hao, Xiaoyan Zhu, Ming Li, and David R. Cheriton. Information distance from a question to an answer. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '07, pages 874–883, New York, NY, USA, 2007. ACM.